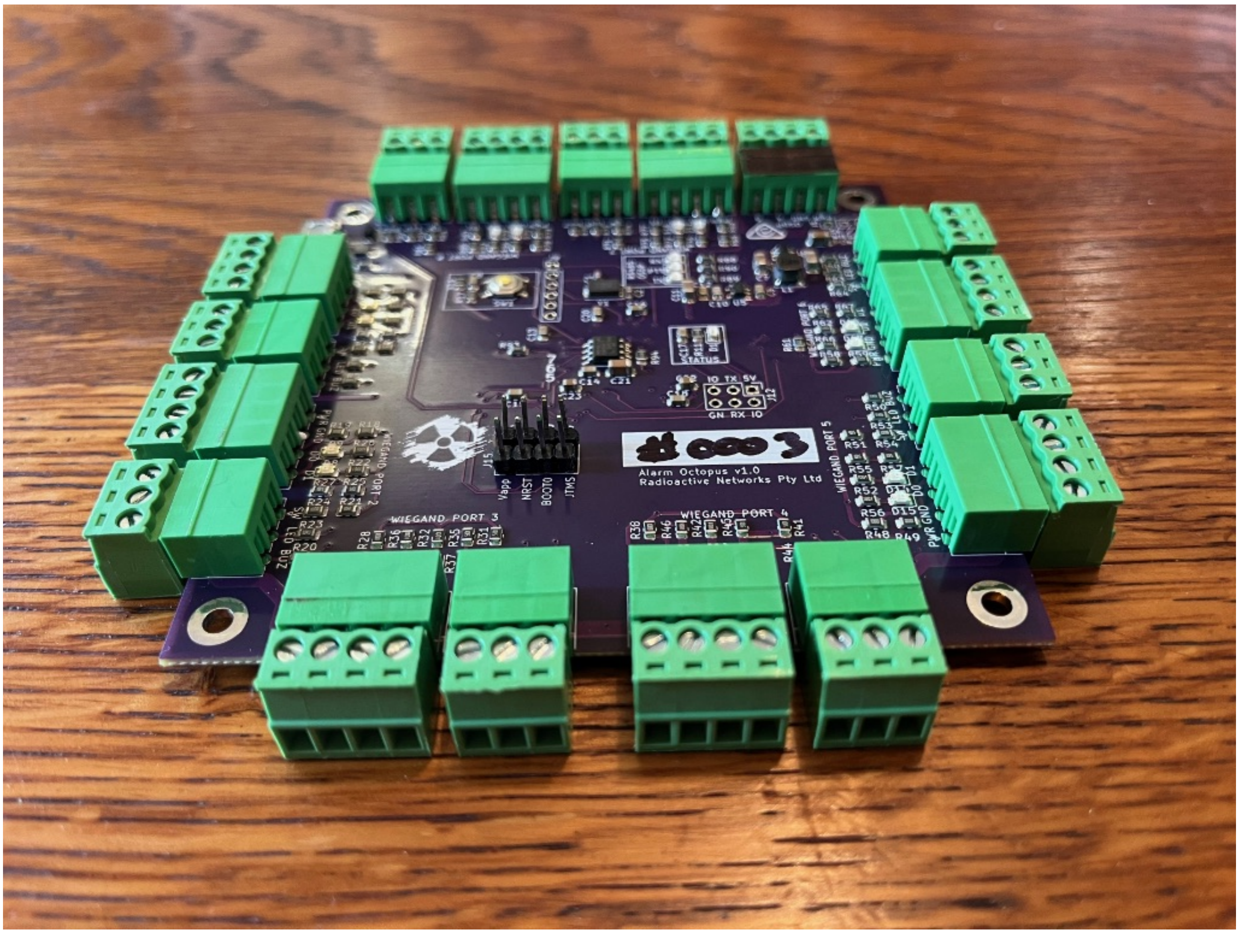
Multi Wiegand to OSDP Converter

Introduction

This documentation describes not only the software running on this device, but the device itself. The product is fairly simple. It takes inputs from one or more Wiegand RFID Readers, and converts this into OSDP on a RS-485 serial bus. We also have a separate getting started document which summarizes much of the content of this document.



Variants

This product line will have some variants:

- Alarm Octopus - eight wiegand RFID inputs with and OSDP output
- Alarm Squid - OSDP input to eight Wiegand outputs

The products will also be able to be used together to extend a number of Wiegand devices over a much larger distance. This document only describes the Alarm Octopus variant.

Hardware

Version one of the product has the following specifications:

- ODSP port operating via RS-485
- USB Serial Port for configuration
- Eight wiegand inputs
- Eight LED and buzzer outputs, normally attached to wiegand readers
- Eight button inputs, normally assigned to wiegand readers
- 12V input. The hardware will support up to 24V, but this will also place the 24V input voltage on the 12V line to the wiegand readers
- The device can also be powered by USB 5V, but any wiegand readers must be configured to operate on 5V or use an alternate power source. The current of the five volt output is limited to about 200 mA.

The hardware also contains a programming connector, should the firmware need to be updated. Future variants of the product are planned to be updatable via USB.

Each wiegand port has a header allowing selecting between using 5V or 12V for wiegand devices.

Communications

A conscious design decision has beem made not to enable configuration of this device remotely. Configuration will need to be done via USB.

On Board

LED and Switch

There are LED's on each Wiegand port to assist with fault finding.

There are also three LED's on the OSDP port, one for received data, one for sent data, and one to indicate when there is a transmission.

The device also has an on-board LED and a pushbutton switch.

The LED will flash periodically. Normally it will toggle whenever there is a transmission made via the OSDP port. With eight OSDP channels operating, this LED should be almost constantly lit. You can use the `led` command to change the functionality of this LED.

In order to reset the device to factory settings, press the button on the main board, and keep it pressed. Then apply power to the board.

If you release the button after five seconds, the board will be reset to factory settings, and the password will be reset to the serial number of the device.

If you release the button after thirty seconds, all encryption keys will be reset, the master card will be removed, any fake cards and offsets removed, and the password will be reset to the serial number of the device.

If you release the button after sixty seconds, the device will start up as normal

Logical Diagram

Pinouts

NOTE: When supplying power via the OSDP port, it is important that the length and gauge of the OSDP cable is sufficient to power any wiegand readers attached to the AlarmOctopus. Alternatively, it is possible to connect a local power supply to the AlarmOctopus INSTEAD of being powered from the control panel.

OSDP Port

1. 12V Input
2. Ground
3. RS-485 A
4. RS-485 B

The OSDP port has three LED's. One for transmit data, one for received data, and one indicating when the RS-485 device is transmitting.

The default OSDP serial parameters are 115200:8:N:1. They can be changed with the `speed` command.

Main Wiegand Port

1. Power - 12V or 5V
2. Ground
3. Wiegand D0
4. Wiegand D1

Each wiegand port also has two LED's. The LED's are connected to D0 and D1. Both should flash when a connected wiegand device is read.

Secondary Wiegand Port

1. Switch Input - Ground input to select
2. LED Output for Wiegand Reader
3. Buzzer Output for Wiegand Reader

USB

1. Mini-USB Port

Pinouts

OSDP - The Open Supervised Device Protocol

OSDP is a serial protocol that communicates predominantly between RFID readers and alarm control panels. Under OSDP, this is between multiple Periheperal Devices (PD's) and a Control Panel (CP). A Control Panel may have multiple Periheperal Devices (PD's) connected to a single Control Panel (CP). There can only be one Control Panel (CP) on a network. The network runs over a RS-485 serial link.

According to the standard, there may be up to 126 PD's on the RS-485 network. In practice many CP's limit the number of PD's.

OSDP Limits On Certain Devices

System	Limit
InnerRange Inception	8
InnerRange Simple Lan Access Module (SLAM)	4
InnerRange Intelligent Lan Access Module (ILAM)	16

Please note that we have only tested this hardware on a limited selection of hardware. If you have hardware that is not listed, please contact us, and we will determine the best way of ensuring that the device works with that hardware.

You may need to disable one or more of the devices OSDP interfaces. If the OSDP comms activity link is not mostly on, this could be an indication that the Control Panel really does not like having so many OSDP devices on the LAN. Of course, it could be an indicator of other issues as well.

Scope of the Device

As mentioned above, this device takes Wiegand inputs and sends them out over OSDP

- Eight OSDP devices sharing a single RS-485 link within the device
- Eight Wiegand RFID Reader inputs, along with optional beep and LED outputs
- An extra pushbutton input for each RFID reader that will send an emulated card read with a user defined RFID card
- Each Wiegand RFID Reader can be assigned to any of the OSDP devices

Custom Functionality

Should you have the need for additional functionality from the device, please conact us and we will see what we can do. Additional software functionality is obviously much easier to implement than new hardware functionality.

Future Devices

We are considering additional devices for the future. They might include devices with some type of relay output or PIR style input. If you have a need for any of those type of devices with an OSDP interface, please contact us.

Physical Security of the Device

This device converts the inherently insecure Wiegand protocol into the significantly more secure OSDP protocol. That is not to say that OSDP is perfectly secure. It is not. As with any technology, there is a tradeoff between ease of use and security. OSDP is fairly secure, but the choice to use it is your choice, not ours.

OSDP can be secured with the use of custom encryption keys on the LAN. Using non-defaulty keys will significantly increase the security of the entire system, and should be considered. Having said that, the weakest part of any system is physical security. This device due to its very natue stores the encryption keys locally. In order to make the entire system as secure as possible, you should use different encryption keys on each OSDP device, and also keep each OSDP device as physically secure as possible. The mounting of this device should be secure, protected by locks and tamper switches.

Updating the Firmware

At the time of writing, there are two methods for updating the firmware. We hope to have further methods in the future.

- ST Link V2 or JTAG using the STM32CubeProgrammer software
- DFU via the USB Port after setting the Boot0 mode

In the future we hope to add the capability to update the firmware via an SD Card, as well as making DFU update easier.

Use Cases

Access Control for Eight Doors

The device by default attaches each Wiegand input to a OSDP device. In this way, the device can control up to eight doors.

Exit Button over OSDP

Each Wiegand port has an additional switch input. This input can be programmed to send a virtual card read. This might be of use if you wanted a Push To Exit button on the inside of a door and wanted to do that via OSDP. The ID of the virtual card is configurable and applies only to an individual reader/door.

Multiple Wiegand Readers in Parallel

The device can be configured so that multiple RFID readers can be sent to the same OSDP port. This would allow, for instance, to have RFID readers on both sides of a door. This would appear to the Control Panel as a single reader, and it could not determine which side of the door the read was on, but it would save an OSDP input in the Control Panel.

Combining Multiple Wiegand Readers Without OSDP

Another way the device can be configured is to combine several RFID readers together to a wiegand output. In this mode you could have seven wiegand inputs and a one wiegand output. Or have two gropus of three inputs being combined into two outputs. Or four inputs and four outputs. The only constraint is that each wiegand port can only be an input or an output.

Wireless Remote with Multiple Wiegand Outputs

We know of one RF Remote Fob reader that has four Wiegand outputs, one for each button on the remote control FOB. Each button on the remote outputs the same Wiegand ID for the same remote, but each remote sends a different ID. This device can be configured so that all four Wiegand inputs are sent to the Control Panel over a single RS-485 cable without needing to purchase individual converters.

Configuration Interface

Only a small number of the internal settings of the devices may be configured remotely, via OSDP. Most settings must be configured when connected to the device locally via a serial connection. Settings are saved automatically onto the device. They can also be downloaded as a text file as a backup.

The serial connection works best if the terminal emulator handles `bs` and `del`. This is often a setting within the emulation package.

The serial parameters are 115200:8:N:1.

Hardware

The device works with most Wiegand devices that use either 26 or 34 bit formats. Contact us if you have devices using other formats and we will see what we can do to add them. Other capabilities such as a PIN pad may be possible, but this would likely require more work.

Commands

Configuration Cookbook

One OSDP Port Active

All eight OSDP Ports Active

Four OSDP Ports Active with two Wiegand Readers per Port

Operations

The device may take a few moments to start up fully. This is because multiple OSDP devices need to synchronise with the Control Panel. Depending on the situation it may take five seconds. Other times we have seen it take a minute. If the Control Panel and the device are both confused because the unit has been restarted, it may take five minutes before both devices synchronise. This is a function of the Control Panel and not this device.

Command Interface

The command interface is used by using a serial terminal. The valid commands are:

- reset
- FACTORY
- osdp
- route
- encrypt
- enable
- fake
- serial
- speed
- mask
- info
- help

Intermediate Commands

- card
- debugosdp
- fake
- basic
- pause
- save
- info
- stats
- hardware
- address
- help
- mask

Web Site

There is a web page to assist with configuring the device. The link is `HERE`. The page is self contained with only HTML and JAVASCRIPT and no external libraries so can be downloaded and saved on your computer for use without internet access. You can do anything with the commands below - it is just that the web page makes it a bit easier to navigate the settings.

The web page does not actually connect to the hardware - it requires you to connect to the hardware with a serial terminal program. Once you have connected and logged into the hardware, as described elsewhere in this document, type the `info` command. Copy the output of the command and paste it into the text box at the end of the page. Then click the `parse commands` button on the page. Your settings will now be shown in the GUI on the page

Make whatever changes you need to. Once complete, press the `generate commands` button. Copy all the commands from the large text box and then paste them into the serial terminal. Depending on the settings changed, you may need to pwoer cycle the board.

The only settings that need to be done manually are restarting the device and changing the password.

We are working on a version of this page that operates using WebSerial, automating managing settings. Please let us know if you would like to be informed when this is available.

reset - restarts the device

This command simply causes the device to restart as if it was power cycled.

FACTORY - reset to factory settings

This command is in upper case. Sending this command will cause the device to be set back to factory settings and then restarted.

osdp - set the OSDP Address for each port.

This device contains a number of OSDP interfaces. This command without any parameters will print the OSDP address for each port. Port numbers start at 0. OSDP Addresses are between 0 and 126.

```
osdp [osdp port number] [osdp address]
```

```
osdp 2 26 // Set the address of OSDP port #2 to 26.
```

route - set the OSDP Port for each wiegand reader.

Unlike most Wiegand to OSDP interfaces, this device lets you assign multiple wiegand readers to a single OSDP port. There is no limit to the number of wiegand devices that can be assigned to an OSDP port. Wiegand interfaces start at 1. OSDP port numbers also start at 1.

```
route [wiegand port number] [osdp port number]
```

```
route 4 3 // Assign wiegand port 4 to OSDP port 3
```

encrypt - set the encryption key for each OSDP port

OSDP can be reasonably secure, ensuring the security of both ends. OSPD permits equipment to operate, depending on configuration, unencrypted, encrypted using a well know encryption key, or using a private encryption key.

OSDP encryption keys are 16 characters long, making up 128 bits. Each character may be between '0' and '9', as well as between 'A' and 'F'. Certain encryption keys are better than others. Having a key where the characters in the key are repeated a lot is seen as bad, as are encryption keys where each character or character pair is just one higher or lower than the one before it.

Absolutely the worst is using the default encryption key of '303132333435363738393A3B3C3D3E3F'. This key is often used to initially lock the device before the Control Panel sends a new encryption key to the device. Some Control Panels will provide the encryption key to the user, and it will be their responsibility to transfer it onto a Periheperal Device. This command assists in that regard.

Typing the 'encrypt' command by itself will display the encryption keys for each OSDP port. The OSDP port number starts at 0. The encryption key, as noted above, is 16 characters long consisting of the numbers '0' to '9' and letters 'A' to 'F'.

```
encrypt [osdp port number] [16 character encryption key]
```

```
encrypt 2 303132333435363738393A3B3C3D3E3F
```

enable - enable and disable wiegand, LED, Buzzer, Relay and OSDP ports

Depending on the product, this product might have sixteen physical wiegand ports and eight logical OSDP ports. The product only has one physical OSDP port, which is needed for the product to work and cannot be turned off. Any of the physical wiegand and logical OSDP ports can be enabled or disabled.

This command also gives the ability to enable and disable LED, Buzzer and Relay connections connected to each wiegand port.

The state may be '1', '0', 'true' or 'false'.

This `enable install` command sets or removes install mode on an OSDP port. Install mode will allow the control panel to set an encryption key for that OSDP port. To do this, the encryption key for the port is temporarily set to a default value. The Control Panel can then send a command to set the encryption key. The device will restart when the encryption key has been saved, and the port will exit install mode. It should be noted that this functionality has not been extensively tested.

When an encryption key has been set by the Control Panel, this will be saved, and the install mode will be removed for that port. It should be noted that the key will not be enacted until the device is restarted.

The secure mode in OSDP enforces a number of constraints for security. It requires that the channel be encrypted and prohibits the standard install encryption key from being used. Operating with it enabled on each port is good practice. Enabling `install` mode on a port will disable secure mode as the two cannot be active at the same time.

```
enable {wiegand|osdp|led|buzzer|relay|install|secure} [port] [state]
```

```
enable wiegand 3 1 # enable wiegand port 3
```

```
enable wiegand 1 1 # enable wiegand port 1
```

```
enable wiegand 1 0 # disable wiegand port 1
enable osdp 6 1 # enable OSDP port 6
```

```
enable osdp 2 false # disable OSDP port 2
```

```
enable install 1 true # Enable install mode for OSDP port 1
```

```
enable secure 3 false # Disable secure mode for OSDP port 3
```

fake - set a 'fake' card for each wiegand port to be sent on button press

This is an easy feature that is hard to describe. Each wiegand interface has an additional pushbutton input that can be used if required. When this button is pressed, the device sends the details of a pre-set RFID card to the Control Pane;, as if the person who had pressed the button actually had that card. This feature is there to be used if needed. The card ID is set by this command. Any card ID can be set, but it is probably best if the card ID is used only in accordance with pressing the pushbutton and not issued to a real user.

26 and 34 bit RFID Card ID's are currently supported. More can be added as needed. 26 bit cards are stored as 8 HEX characters. 34 bit cards are stored as 10 HEX characters. In both cases, the numbers are sent LSB first, and are 0 padded at the end. In practice, read a real card and use the information displayed to determine what to enter. On the InnerRange Inception, a 34 bit card might read as '22015993D5C1'. In this case, drop the leading '22' and enter the card ID as '015993D5C1'

The wiegand port number starts at 1'

```
fake [port] [card number]
```

```
fake 2 015993D5C1 # send the card ID of 015993D5C1 whenever wiegand port 2's button is pressed.
```

master - use a card to log in to the console

Rather than using a password to access the console, it is possible to use a specific RFID card. This will unlock the console as if the password had been entered. It is possible to deactivate this featre by entering a card number with eight or ten 0's.

```
master [card number]
```

```
master 0123456789 # unlock the console whenever the card ID of 0123456789 is used
```

serial - Set the device serial number

This command displays the hardware serial number of the device. It cannot be changed. The serial number may be between 0 and FFFFFFFF.

```
serial
```

led - change the functionality of the on-board LED

The on board LED has a number of modes. Mode 0 has teh led changing state as the main loop executes. In mode 2 the LED toggles in response to OSDP transmissions. Mode 3 momentarily flashes the LED in line with Wiegand output

```
led [mode]
```

mask - display the status of the OSDP connections

Most of the comands deal with settings. This command deals with actual status. It reports the status of each OSDP logical port, if it is active and if encryption is active. It only displays OSDP status for ports that are not disabled. The first status returned indicats if there has been a sesssion successfully made on each port. The second indicates if the connection has been secured with encryption. It should be noted that the session may report as being offline between when the connection is initially made and when it is secured.

```
mask
```

card - send false card code

As noted by the 'fake' command, it is possible to send a preset card read to the Control Panel when a button is pressed. For testing, the 'card' command has been created to emulate that button being pressed. It can be used on any of the Wiegand ports.

```
card [port]
```

```
card 3 # Emulate Wiegand button 3 being pressed
```

offset - Modify the Wiegand code sent from a reader

This is another unusual command that has a rather interesting use case. Many Control Panels are limited in the number of OSDP ports that they support. They may only support four OSDP readers for expansion board. There are valid technical reasons why a manufacturer may wish to do this.

Some manufacturers for equally as valid reasons make devices with multiple wiegand outputs. One device that we have come across is a reasonably priced RF FOB reader which has four wiegand outputs. FOBs that work with this reader have four buttons and will send a code out a different wiegand port depending on what button was pressed. The situation though is that the only difference between any button press for a particular FOB is the wiegand output the code comes out. Different FOBs have different codes of course.

This means that ultimately by using this device that you have just used up four wiegand inputs or OSDP devices.

The offset command can help. It is not an ideal solution, but it might be just what is needed. This command modifes any wiegand reads on a port by applying the offset to the code that is sent. This works best when you have multiple wiegand ports going to a single OSDP device. In the situation above you could send the four wiegand ports into a single OSDP device. But in this case, you wouldn't know which button was pressed.

So you then apply the offset command to each wiegand port - for instance applying an offset of 100 to port 1, 200 to port 2 and do on. Then when a button was pressed on a remote, the code would be different for each button. So instead of enrolling a single FOB, you need to enroll each button. This is not elegant but might be just what the customer needs.

The actual offset is a bit more complex than adding a nunber to the value on each port. It actually performs an 'Exclusive Or' to the card value. Because of that, the offsets in hex are generally best to be a binary sequence. That is, good offsets for the four ports might be something like 00000000, 00010000, 00020000 and 00030000.

As the product matures, the description of this command will likely improve. Right now, if you can't get this command to do what you need, please contact us and we would be happy to help out.

The value for the offset is sent in the same was as the 'fake' command with either eight or ten hex digits.

The wiegand port number starts at 1'

```
offset [port] [card number]
```

```
offset 2 0001000000 # Apply this offset to any card reads on wiegand port 2
```

output

This is an experimental feature which works on OSDP ports, regardless if that OSDP port is active or not. When `output` is active for an OSDP port, any Wiegand reads on that OSDP port are also output on the selected Wiegand port as an output. This can be useful when you want to combine two or more Wiegand readers to a single wiegand output, regardless of if OSDP is needed. Enter a wiegand port of 0 to disable.

A restart is required when disabling a wiegand output when you want to use that port for wiegand input. A restart is suggested needed when enabling a wiegand input

```
output [osdp port] [wiegand port]
```

```
output 2 4 # Send all reader data on OSDP port 2 to Wiegand port 4
```

protect

By default, very little information is shown on the serial console unless the user has entered a password. By setting `protect 0`, all status information will be displayed even if the console is locked. A warning, however, is that the console has the potential to leak information that could be used to bypass security.

```
protect [level]
```

```
protect 1 # Protect card data from being examined without a password
```

stats

The stats page collects a number of statistics for the device. The statistics are colelcted both since the last reboot, and since first power up.

hardware

This command displays the status of all the hardware inputs and set the state of the hardware outputs.

When using Wiegand, the command is of the form `hardware wiegand [port] [pin] [state]`. Port would normally be between 1 and 8. Pin would be from 0 to 4 - with 0 = D0; 1 = D1; 2 = LED; 3 = Buzzer; 4 = Switch. At the moment, only LED (2) and Buzzer (3) are implemented. State may be 1, 'T' or 't' for On, and 0, 'F' or 'f' for Off.

This command will remain in place until the unit is restarted or the Control Panel sends a new command. Importantly, the hardware will flash the LED output when OSDP is not connected, which will affect this command. This is a debugging function.

```
hardware wiegand [port] [pin] [state]
```

```
hardware wiegand 1 2 1
```

address

The address command displays the OSDP serial number for each OSDP port, to allow each port to be uniquely identified. The OSDP serial number will be of the form xxx0 for the first OSDP port, xxx1 for the next, etc; where xxx are HEX digits. The OSDP serial number is based on the hadware serial number, and cannot be changed.

```
address
```

debugosdp

This is a command that hopefully never needs to be used. It provides low level debugging of the communications between the control panel and the board. Debugging is on a per OSDP port basis. The port number is just the index of the OSDP port. Flags are a bit more complex. Flags is a number. To determine the value of the flag, add up all the desired options.

1 = Data Trace. 2 = Packet Trace. 4 = Monitor POLL packets.

If you want to do a Data Trace (1) and Monitor Poll Packets (4), add up the values for both, and in this case you will get a flag value of 5. If you want everything, select a flag value of 7. It can be disabled on a per port basis with the value of zero.

Just a warning. If you start using this feature and want to turn it off, either send the commands to turn it off without being able to see what you are typing, or disconnect the OSDP port.

```
debugosdp [port] [flags]
debugosdp 2 7
```

basic

The `basic` command very quickly sets up a few basic settings. It removes any OSDP debugging on all ports. It also only enables a single OSDP port. This is enough to be able to type more commands without being bamboozled.

```
basic
```

pause

The `pause` command pauses saving settings until the command is issued again or the device is restarted. This can be useful when saving a lot of settings as it will reduce the number of writes to memory. This command should not be needed.

```
pause
```

save

The `save` command saves settings, when the `pause` command is active.

```
save
```

password

The password command is used to unlock the console, lock the console and change the password. This functionality is designed to stop casual tampering. It is not designed to stop someone who has the means and resources to bypass the password. It should be reasonably secure. Obviously, the password used should be unique and not used on any other system. We do not use the password directly, but store a transform of the password using a one way function.

To lock the console, type 'password'.

To unlock the console, type 'password' followed by your password. The default password is the serial number of the unit. It is recommended that you change this.

To change the password, type 'password set' followed by your desired password. The password must be eight characters or more. There are two special 'passwords'. The first is 'blank' which will remove the password. The second is 'serial' which sets the passowrd to the serial number of the device. This is the default password.

If a password is set, the console will lock automatically about five minutes after the last command is entered, or when 'password' is typed by itself.

```
password
```

```
password [password]
password set [password|blank|serial]
```

speed

This command changes the speed of the OSDP port. You must perform a restart of the board to enable these settings. The board cannot have the OSDP serial speed changed with the osdp_COMSET command under the OSDP protocol. Valid speeds are 9600, 19200, 38400, 57600 or 115200.

```
speed
```

```
speed [speed]
```

info

Display information on all parameters

```
info
```

help

Display bacic help. Type help byitself to list all known commands, and help followed by the name of the function for specific help on that function

```
help
```

Show

Display the last few wiegand card reads, along with the port and the age.

```
show
```

Inject

Inject RFID from the command line as if it was sent by an actual card reader. It works on an OSDP port. The Card number needs to be eight or ten hex digits long. This command can be used to integrate other hardware to the alarm system.

```
inject [port] [card number]
```

```
inject 2 ABCD1234EF
```

Open Source

As with most modern software, this product uses a number of Open Source libraries. We are more than happy to provide you with copies of the libraries we use. Alternately, you can download the software on the links below. Importantly, we have modified the LibOSDP package, and have made our modifications available for free. Please be aware however that our own software and hardware are both convered by copyright. Likewise, we do not offer support on any of these libraries, including our modifications to the LibOSDP package.

LibOSDP

- https://libosdp.sidcha.dev/license - Apache
- Modifications available on https://github.com/vk2tds/libosdp_arduino

Automaton

- https://github.com/tinkerspy/Automaton/blob/master/LICENSE - MIT
- https://github.com/tinkerspy/Automaton/

FlashStorage-STM32

- https://github.com/khoih-prog/FlashStorage_STM32/blob/main/LICENSE - MIT
- https://github.com/khoih-prog/FlashStorage_STM32

ParseCommands

- https://github.com/Gfy63/ParseCommands/blob/main/LICENSE - GPL
- https://github.com/Gfy63/ParseCommands

Yet Another Arduino Wiegand Library

- https://github.com/paulo-raca/YetAnotherArduinoWiegandLibrary/blob/master/LICENSE - Public Domain
- https://github.com/paulo-raca/YetAnotherArduinoWiegandLibrary

ArduinoUniqueID

- https://github.com/ricaun/ArduinoUniqueID/blob/master/LICENSE - MIT
- https://github.com/ricaun/ArduinoUniqueID