AN-276

Configuring Credential Types in Protege GX

Application Note

Last Published: 18-Oct-23 2:15 PM

Contents

Introduction 4
Prerequisites 4
Programming Overview 6
Supported Formats 7
Credential Type Settings 7
Scenario 1: Legacy Card Readers 9
Creating the Credential Type 9
Configuring the Reader Expander 10
Adding the Door Type 10
Assigning Access to Users 11
Scenario 2: License Plate Recognition 12
Creating the Credential Type 12
Configuring the Onboard Reader Expander 13
Adding the Smart Readers 13
Adding the Door Type 14
Assigning Access to Users 14
Scenario 3: Serial PIN Pad 15
Creating the Credential Type 15
Configuring the Onboard Reader Expander 16
Adding the Smart Reader 16
Adding the Door Type 16
Assigning Access to Users 17
Appendix: Configuring Custom Wiegand Credential Types 18
Configuring a Descramble Format 19

Introduction

Credential types are used by the Protege GX system to recognize alternative forms of user identification for the purpose of access control. They enable a controller or reader expander to interpret custom Wiegand card formats from card readers connected to the reader ports. The controller can also recognize credential data in a wide variety of other formats, received through either the ethernet port or RS-485 reader port (often via an RS-232 to RS-485 converter). This provides a flexible method of integrating with third-party systems and card readers.

This feature has a broad range of applications, including:

- The use of uncommon or legacy card formats in Protege GX.
- License plate recognition via cameras or DVRs/NVRs connected over IP.
- Integration with third-party modules which send data such as PINs, passwords or biometric credentials over serial or ethernet interfaces.
- Controlling user access based on compliance or certification requirements. For more information, see Application Note 286: Programming Compliance Types in Protege GX.

This application note provides instructions on programming Protege GX for operation with customized credential types. It includes in-depth programming examples which you can adapt to your own requirements.

This document only covers the programming that is relevant to Protege GX. Third-party applications and devices may have their own programming requirements.

Prerequisites

Protege GX Components

The following components must be installed and operational.

Component	Version	Notes
Protege GX	4.2.180 or higher	• To use compliance types, version 4.3.285 or higher is required.
Protege GX Controller	2.08.766 or higher	• The stated firmware version is required even when the credential data is received by connected reader expanders. • For earlier firmware versions, a single custom Wiegand format can be programmed in **Sites | Controllers | Custom reader format**. • Each controller can support up to 248 third-party reading devices connected over ethernet, and up to 2 card readers or 1 third-party serial device. • Compliance types require firmware version 2.08.1000 or higher.
Reader Expanders		
Protege Reader Expander	-	• Reader expanders only support Wiegand and Compliance credential types. They do not support other credential type formats. • Compliance types require firmware version 1.12.582 or higher.

Licensing

License	Order Code	Notes
Third-party Reader Interface License	PRT-GX-TPR-IF	1 license per third-party smart reader record programmed. These licenses are required to use credential type formats other than Wiegand or Compliance.
Door License	PRT-GX-DOR-1	1 license per door record.
	PRT-GX-DOR-10	
	PRT-GX-DOR-50	

Programming Overview

Because credential types have a variety of applications, the specific programming steps required can be quite different for each situation. This section outlines the general programming steps involved in configuring and using credential types, while the programming scenarios go into more detail about specific situations.

This app note does not include instructions for programming compliance types. For programming instructions, see Application Note 286: Programming Compliance Types in Protege GX.

The basic steps for programming credential types are:

1. In **Programming | Doors**, add any doors that will be used with this credential type. Add the doors to any access levels that will be using the credential types.
2. Add the new credential type in **Sites | Credential types**. Configure the format of the credential type based on the requirements for your specific credentials.

   The supported formats and other settings for defining a custom credential type are listed below. Additional information about configuring Wiegand format credential types is included in the appendix (see page 18).
3. If you are configuring a Wiegand format credential type for a controller or expander's **reader port**:
   - If the readers are connected to the controller, enable the controller's onboard reader expander in **Sites | Controllers | Configuration**.
   - In **Expanders | Reader expanders**, set the **Port 1/2 network type** to match the type of reader connected (Wiegand or ICT RS485).
   - In the **Reader 1/2** tab, set the **Reader 1/2 format** to Custom credential.

     When Custom credential is selected, the **Reader 1/2 secondary format** is ignored. To provide an alternative credential option, you can set a **Fallback door type** in **Programming | Door types**.
   - Save the settings and module update the reader expanders you have configured.
4. If you are configuring a credential type for the controller's **ethernet port**:
   - Enable the controller's onboard reader expander in **Sites | Controllers | Configuration** (if readers are connected to the controller).
   - Select the onboard reader expander in **Expanders | Reader expanders** and set the **Ethernet network type** to Third party generic. Enter the **Ethernet port** that the controller will listen on.
   - If multiple readers will be sending data over ethernet, enter the command:

     `SmartReaderPortOffset = true`

     With this command enabled, the controller will listen for each smart reader on a different ethernet port. The port used by each smart reader is equal to the **Ethernet port** plus the **Configured address** of the smart reader record.
   - Save the settings and module update the onboard reader expander.
   - In **Expanders | Smart readers**, add one smart reader for each reading device connected over ethernet. Assign each smart reader a unique address on the controller's ethernet port.
   - Configure the settings for each smart reader, including the door, direction and valid credential types.
5. If you are configuring a credential type for the controller's **serial interface** (RS-485):
   - Enable the controller's onboard reader expander in **Sites | Controllers | Configuration** (if readers are connected to the controller).
   - Select the onboard reader expander in **Expanders | Reader expanders** and set the **Port 1/2 network type** to Third party generic.
   - In the **Reader 1/2** tab, enter the communication settings for the third-party module, such as the **Baud rate** and **Parity**.
   - Save the settings and module update the onboard reader expander.

- In **Expanders | Smart readers**, add a smart reader.
- Configure the settings for the smart reader, including the door, direction and valid credential types.

6. Create a new door type in **Programming | Door types**. Set the **Entry/Exit reading mode** to Custom and add all required credential types to the **Entry/Exit credential types** field.
7. Assign the new door type to all of the doors which require the assigned credential types.
8. In **Users | Users**, select each user who needs to use the new credential type. Scroll down to the **Credentials** section and enter the credential against the user.
   - For Wiegand card formats, you can enter the facility and card numbers separated by a colon (e.g. 1234:567890).
   - For other credential formats, enter a number or alphanumeric string (such as a license plate).
   - Like normal cards, you can badge or enter unassigned credentials and right click on the 'Read raw credential' event to assign them to a user. Alternatively, you can use the ICT Data Sync Service to synchronize user credentials with another system.
9. Use the **Disabled** checkbox and **Inactivity period** to disable the credential when it is no longer needed.

   The credential expiry settings (**Start** and **End**) are only available for compliance types.

Supported Formats

The following credential data formats are supported by this feature:

- **Unicode**: The credential data sent to the controller uses two bytes to represent each character as per the Unicode standard.
- **UTF8**: The credential data sent to the controller uses a variable number of bytes to represent each character as per the UTF-8 standard.
- **ASCII**: The credential data sent to the controller uses a single byte to represent each character as per the ASCII standard.
- **Numeric**: The credential data sent to the controller is a binary number composed of up to 8 bytes. The bytes are ordered using little endian.
- **Hexadecimal**: The credential data is sent to the controller as an array of binary numbers. When the specific credential is entered into the user programming for each user, the format used is hexadecimal with the numbers 0-9 and letters A-F representing each nibble of the credential.
- **Wiegand**: The credential data sent to the controller or reader expander is composed of a Wiegand bit stream. This bit stream can be encoded in numerous different ways and a format descriptor must be included in the **Wiegand or TLV format** field. For the Wiegand format the preceding, trailing and prefix character settings and case sensitive setting are ignored.
- **TLV**: This option is reserved for future development.
- **Compliance**: A special credential type that allows you to use custom requirements such as health and safety certificates, driver's licenses and industry qualifications as credentials. This credential type can be used with controllers and reader expanders. Compliance types have different settings from regular credential types.

Controllers support all credential type formats via either RS-485 or ethernet. Reader expanders only support Wiegand credential types.

This document does not cover programming for compliance types. For information on this feature, see Application Note 286: Programming Compliance Types in Protege GX.

Credential Type Settings

The following settings are available for programming credential types. Many of these settings depend on the specific format of the data sent by the third-party system, and must be programmed correctly to allow the controller to recognize this data.

- **Preceding characters**: The maximum number of characters that may be ignored at the start of the data packet received by the controller. If the credential is found before this number of characters is counted it will still be accepted.

  This setting is determined by the third-party device/application.

- **Trailing characters**: The maximum number of characters that may be ignored at the end of the data packet received by the controller. If there are fewer than this number of trailing characters after the credential is found the credential will still be accepted.

  For example, this field may be set to 1 to ensure that credentials will be accepted even if they are followed by a carriage return character.

  This setting is determined by the third-party device/application.

- **Prefix**: The characters that are required at the start of the credential data packet sent to the controller.

  This setting is determined by the third-party device/application.

- **Case sensitive**: Defines whether or not the data is case sensitive.

  This setting is determined by the third-party device/application.

- **Unique value**: When this option is enabled, duplicate credentials are not allowed (i.e. two users may not have the same credential). When disabled, duplicate credentials are permitted.

  If non-unique credential values are permitted, ensure that any door type using this credential type also requires a unique credential (e.g. card) for two-factor authentication so that Protege GX can accurately identify the user requesting access.

- **Credential limit per user**: This option allows you to restrict the number of credentials that can be added to each user through the Protege GX software. The credential limit is set to Unlimited by default, and can be restricted to a number from 1-10.

  Compliance types are always unlimited.

  It is not possible to set a credential limit if one or more users already have more than the maximum number of credentials. You can run a user search (**Users | User search**) with the Credential type column to see which users have excess credentials assigned.

  The SOAP service ignores the credential number restriction.

- **Disable inactive user credentials**: When this option is enabled, new users added to the system will automatically have a default **Inactivity period** applied to this credential type (**Users | Users | General**). If the user does not use the credential within that period it will be disabled.

  When you save a change to this setting you will be prompted to apply the change to all users. Select **Yes** to override the settings programmed in individual user records with the new default value. This may take some time for sites with a large number of users. If you select **No**, the default setting will only be applied to users added after the change.

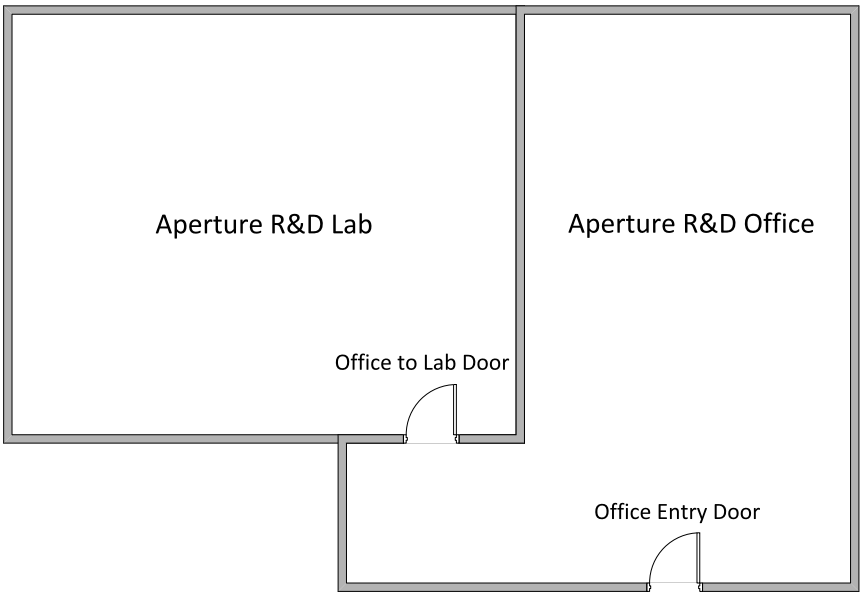
- **Default credential inactivity period**: Set the number of days, hours or minutes that the credential must be inactive before it is disabled.

  The maximum inactivity period is 365 days. If you enter a longer period the field will reset to the default period of 30 days.

Scenario 1: Legacy Card Readers

In the following programming scenarios we will be considering the R&D facility for Aperture Appliance, who are upgrading their existing security system to Protege GX.

The external and internal doors in the Aperture Appliance facility use legacy card readers which read a 36 bit Wiegand format. To save costs, the company has decided to keep the existing readers and cards as they upgrade to Protege GX. Therefore, we must program the reader expanders to recognize this card format.



Before you begin, program the following records:

Reader Expander	Door	Reader Connection
Aperture RD1	Office Entry Door	Wiegand (multiplexed)
	Office to Lab Door	Wiegand (multiplexed)

Creating the Credential Type

To begin, we must program a credential type which will enable the reader expander to recognize the 36 bit card data received from the readers. For this we need knowledge of the Wiegand format of the card data. More details about how the Wiegand format is defined are outlined in the appendix (see page 18). For this example, we will use the following hypothetical format:

```
#36bit
A,FACILITY,10,MSB,BIN
B,CARD,24,MSB,BIN
PPAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBB
E.XXXXXXXXX.........................
.O.........XXXXXXXXXXXXXXXXXXXXXXXX
```

To program a credential type with this format:

1. Navigate to **Sites | Credential types**.
2. Add a new credential type with the name 36 bit format.
3. Set the **Format** to Wiegand.

4. Enter the card format in the **Wiegand or TLV format** field. Each line in the example above must be separated by 1-2 underscores, as follows:

```
#36bit__A,FACILITY,10,MSB,BIN__B,CARD,24,MSB,BIN__
PPAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBB__
E.XXXXXXXXX.......................__.O..........XXXXXXXXXXXXXXXXXXXXXXXX
```

5. This format has no **Preceding characters**, **Trailing characters** or **Prefix**.
6. Tick the **Unique value** box to ensure that multiple users cannot have the same card number.
7. The client would like unused access cards to expire after a period of 90 days. To achieve this we can set a default expiry period for all new user records:
   - Enable the **Disable inactive user credentials** option.
   - Set the **Default credential inactivity period** to 90 days.
8. Click **Save**.

Configuring the Reader Expander

The reader expander which the legacy card readers are connected to must be configured to support them and correctly interpret the card data. Because the credentials are standard Wiegand data provided by card readers connected to the reader port, there is no need to program smart reader records (as in the other scenarios below).

1. Navigate to **Expanders | Reader Expanders** and select Aperture RD1.
2. If they are not already configured:
   - Set the **Port 1 network type** and **Port 2 network type** to Wiegand.
   - Enable the **Multiple reader input port 1/2** options.
3. In the **Reader 1** tab, set the **Reader 1 format** to Custom credential. This allows the reader expander to use the credential types programmed in the door type to interpret the card data (see below).
4. Repeat in the **Reader 2** tab.
5. Click **Save**.
6. Wait for the changes to be downloaded, then right click on the reader expander and click **Update Module**.

Adding the Door Type

We need to program a door type to allow users to gain access using their 36 bit format cards. For added security, the company requires users to enter both card and PIN to enter these doors. Only a card is required to exit.

1. Navigate to **Programming | Door types**.
2. Add a new door type with the name Card and PIN Entry, Card Exit (36 bit).
3. Scroll down to the **Entry** section and set the **Entry reading mode** to Custom.
4. Under the **Entry credential types** heading, click **Add**. Select both the 36 bit format and PIN credential types and click **OK**.

   When multiple credential types are added to this field, users must enter all of the credentials to gain access to the door. To specify an alternative credential type, assign a **Fallback door type**.

5. Enable the **Sequence** option and position the 36 bit format above the PIN credential type. This will ensure that users badge the card before entering their PIN.
6. Set the **Exit reading mode** to Custom.
7. Under the **Exit credential types** heading, click **Add**. Select the 36 bit format credential type only and click **OK**.
8. Click **Save**.
9. Navigate to **Programming | Doors** and multi-select (Shift + click) the Office Entry Door and Office to Lab Door.

10. Set the **Door type** to Card and PIN Entry, Card Exit (36 bit).
11. Click **Save**.

Assigning Access to Users

To gain access to the doors, users must have the appropriate access levels and card and PIN credentials assigned.

To assign door access to users:

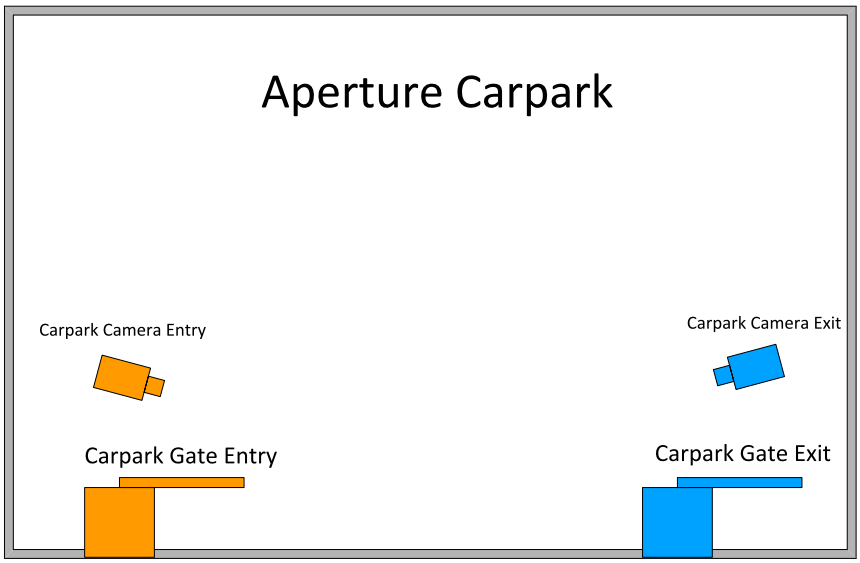
1. Navigate to **Groups | Door Groups** and create a new door group called R&D Doors.
2. Add both the Office Entry Door and Office to Lab Door to the door group. Click **Save**.
3. Navigate to **Users | Access Levels** and assign the new door group to all access levels that require access to the facility.

To assign 36 bit credentials to users:

1. Navigate to **Users | Users**.
2. Select or add a user record and enter a **PIN** code for the user.
3. Scroll down to the **Credentials** section and observe that the 36 bit format credential has been added.
4. Enter the user's facility and card number, separated by a colon, e.g. 123:4567890.
5. In the **Access Levels** tab, ensure that the user has access to the facility doors.
6. Click **Save**.
7. Repeat for all users who require access to the facility.

Scenario 2: License Plate Recognition

To prevent unauthorized access to their sensitive facility, Aperture has set up a license plate recognition (LPR) system in their carpark. Cameras in the carpark read license plates and send the resulting ASCII data to the controller over ethernet.



Each gate in the carpark is associated with a camera, which sends data to the controller on a specific ethernet port. Before you begin, create the door and camera records listed below.

Door	Camera	Ethernet Port
Carpark Gate Entry	Carpark Camera Entry	9001
Carpark Gate Exit	Carpark Camera Exit	9002

On a real site we would need to ensure that the ethernet port for each camera is open in the firewall, so that the cameras can send credential data to the controller.

Creating the Credential Type

In this scenario, the cameras send license plate data to the controller in ASCII format with no additional characters. We need to create a credential type that will recognize this input.

1. Navigate to **Sites | Credential types**.
2. Add a new credential type with the name License Plate.
3. Set the **Format** to ASCII.
4. For this format, there is no need to configure the **Preceding characters**, **Trailing characters** or **Prefix**.
5. Enable the **Unique value** checkbox. This ensures that two users cannot have the same license plate credential.

   If **Unique value** were disabled, we would need to include a second credential such as card or PIN for carpark access.

6. Aperture would like to disable any license plates that have not been used to enter the carpark after a period of 90 days. Enable **Disable inactive user credentials** and set the **Default credential inactivity period** to 90 days.
7. Click **Save**.

Configuring the Onboard Reader Expander

We must configure the controller's onboard reader expander to receive data from the cameras over the ethernet port.

First, enable the controller's onboard reader expander if you have not already done so:

1. Navigate to **Sites | Controllers** and select the controller used for this scenario.
2. In the **Configuration** tab, set **Register as reader expander** to an available address (i.e. not used by a physical reader expander).
3. Click **Save**.
4. Navigate to **Expander | Reader expanders** and add a new reader expander with the same **Physical address** as configured above.

Next, configure the ethernet port:

1. In the **General** tab, set the **Ethernet network type** to Third party generic.
2. Enter the first **Ethernet port** that the controller will listen on for credential data. In this scenario, the port is 9000.
3. Because there is more than one camera sending data over the ethernet port, we must use a port offset to differentiate between the data from different cameras. In the **Commands** section, add the following:

   ```
   SmartReaderPortOffset = true
   ```

   With this command enabled, the controller will listen for each camera on a different ethernet port. The port used by each camera is equal to the **Ethernet port** plus the **Configured address** of the smart reader record (which we will program below).
4. Click **Save**.
5. Wait for the changes to be downloaded, then right click on the reader expander and click **Update module**.

Adding the Smart Readers

Each camera will be represented by a smart reader, which allows the controller to associate incoming credential data with a specific door.

1. Navigate to **Expanders | Smart readers** and select the relevant **Controller** in the toolbar.
2. Add a new smart reader called Carpark Camera Entry.
3. Set the **Expander address** to match the **Physical address** of the controller's onboard reader expander.
4. Set the **Expander port** to Ethernet.
5. Set the **Configured address** to 1. This address and the **Ethernet port** configured above determine the TCP/IP port that the controller will listen on for credentials from this camera. In this case, the TCP/IP port is 9001.

   The controller will listen for the smart reader with address 1 on both the base **Ethernet port**, and the **Ethernet port** + 1. In this case, it will listen for Carpark Camera Entry on both ports 9000 and 9001.
6. In the **Reader** tab, set the **Reader one location** to Entry and the **Reader one door** to Carpark Gate Entry.
7. Scroll down to the **Reader credential match types** section. Click **Add** and select the License Plate credential type. Click **OK**.
8. Save the smart reader record.
9. Repeat the above steps to configure a second smart reader with the following settings:
   - **Name**: Carpark Camera Exit
   - **Expander address**: The **Physical address** of the onboard reader expander
   - **Expander port**: Ethernet

- **Configured address**: 2

  The controller will listen for this camera over port 9002.
- **Reader one direction**: Exit
- **Reader one door**: Carpark Gate Exit
- **Reader credential match types**: License Plate

10. Save the second smart reader record.

Adding the Door Type

We need to program a door type to allow the carpark gates to accept license plate credentials.

1. Navigate to **Programming | Door types**.
2. Add a new door type with the name Carpark Gate.
3. Scroll down to the **Entry** section and set the **Entry reading mode** to Custom.
4. Under the **Entry credential types** heading, click **Add**. Select the License Plate credential type and click **OK**.
5. Repeat the above to set the **Exit reading mode** and **Exit credential types**.
6. The carpark gates will also have card readers installed to allow users to open the gate without a vehicle. Set the **Fallback door type** to Card.
7. Click **Save**.
8. Navigate to **Programming | Doors** and multi-select (Shift + click) the Carpark Gate Entry and Carpark Gate Exit.
9. Set the **Door type** to Carpark Gate.
10. Click **Save**.

Assigning Access to Users

For users to access the carpark gates the doors must be assigned to their access levels. In this case, it is also necessary to assign a license plate credential to each user, just as you would a card number or PIN.

To assign door access to users:

1. Navigate to **Groups | Door Groups** and create a new door group called Carpark Gates.
2. Add both Carpark Gate Entry and Carpark Gate Exit to the door group. Click **Save**.
3. Navigate to **Users | Access Levels** and assign the new door group to all access levels that require access to the carpark.

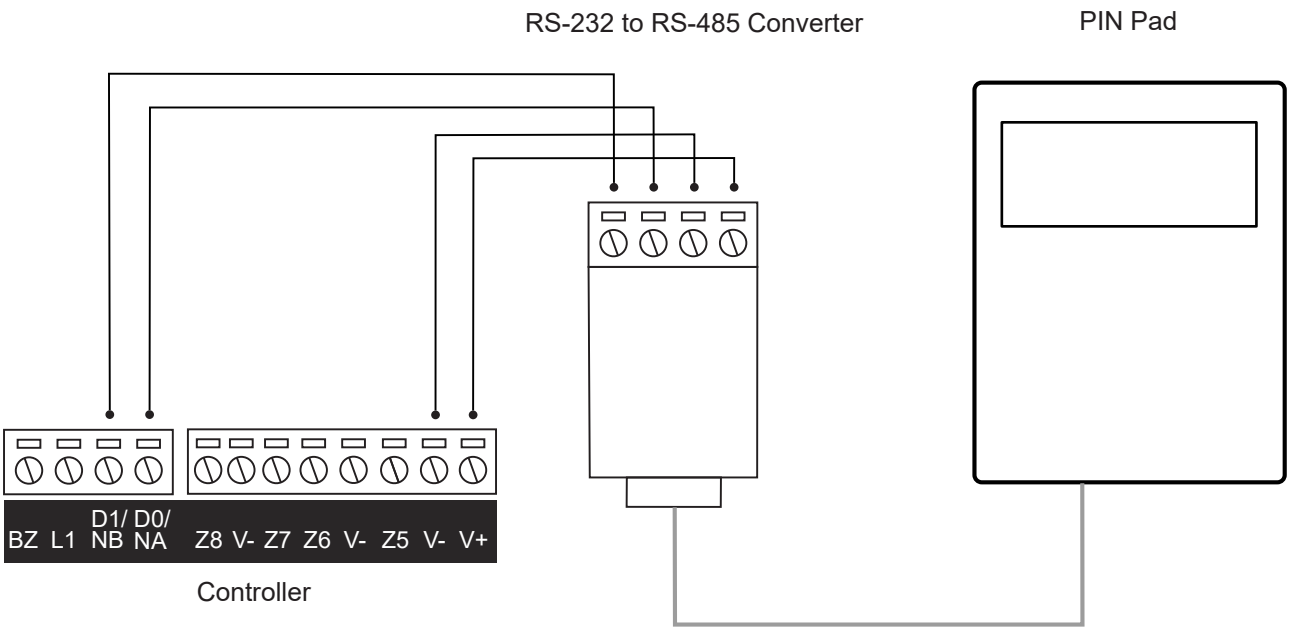
To assign license plate credentials to users:

1. Navigate to **Users | Users**.
2. Select or add a user record and scroll down to the **Credentials** section. Observe that the License Plate credential has been added to the user and has an **Inactivity period** of 90 days as we configured above.
3. Enter a license plate, such as ABC123.
4. In the **Access Levels** tab, ensure that the user has access to the carpark gates.
5. Click **Save**.
6. Repeat for all users who require access to the carpark.

Scenario 3: Serial PIN Pad

The Aperture Appliance R&D lab contains sensitive assembler machinery that can only be controlled by authorized operators. The assembler is activated using a PIN pad installed by the machine. Part of the security system upgrade includes integrating this machinery with Protege GX, so that authorization can be determined by the user's access level.

The PIN pad's serial interface will be connected to the Protege GX controller's RS-485 reader port, via an RS-232 to RS-485 converter:



Note: Serial modules must be connected to the reader ports on the controller. Reader expanders can only interpret data from card readers.

The initiation circuit for the machinery will be connected to one of the controller's relay outputs. We want the assembler's initiation switch to turn on when an authorized user enters a PIN, and off when they enter the PIN again. The assembler can be programmed as a Protege GX door with the following settings:

Door	Lock output	Lock activation time
Assembler (Unlock to Initiate)	Assembler Initiation Switch (Controller 1.3)	0

Creating the Credential Type

The PIN pad sends PIN data to the controller in an ASCII format, with some preceding and trailing characters. Each PIN code sent by the PIN pad has the following structure: `PIN1234#`

To create the credential type:

1. Navigate to **Sites | Credential Types**.
2. Add a new credential type with the name Assembler PIN Code.
3. Set the **Format** to ASCII.
4. Based on the structure described above, we need to enter the following information:
   - **Preceding characters**: 3
   - **Trailing characters**: 1
   - **Prefix**: PIN

5. Enable the **Unique value** option.
6. Click **Save**.

Configuring the Onboard Reader Expander

We need to enable and configure the controller's onboard reader expander so that it can receive the serial data over reader port 1.

First, enable the controller's onboard reader expander if you have not done so already:

1. Navigate to **Sites | Controllers** and select the controller used for this scenario.
2. In the **Configuration** tab, set **Register as reader expander** to an available address (i.e. not used by a physical reader expander).
3. Click **Save**.
4. Navigate to **Expander | Reader expanders** and add a new reader expander with the same **Physical address** as configured above.

Next, configure reader port 1 to communicate with the PIN pad:

1. In the **General** tab, set the **Port 1 network type** to Third party generic.
2. Open the **Reader 1** tab and scroll down to the **Third party generic** section.
3. Enter the following communication details for the PIN pad to allow the controller to interpret the incoming data. These details would be supplied by the PIN pad manufacturer.
   - **Reader 1 baud rate**: 4800
   - **Reader 1 parity**: None
   - **Reader 1 stop bits**: 1 Stop
   - **Reader 1 inter-byte time out**: 5 ms
4. Click **Save**.
5. Wait for the changes to be downloaded, then right click on the reader expander and click **Update Module**.

Adding the Smart Reader

The PIN pad will be represented by a smart reader addressed to the controller's reader port 1.

1. Navigate to **Expanders | Smart readers** and select the relevant **Controller** in the toolbar.
2. Add a new smart reader called Assembler PIN Pad.
3. Set the **Expander address** to match the **Physical address** of the controller's onboard reader expander.
4. Set the **Expander port** to Port 1.
5. Leave the **Configured address** as <not set>. As the controller can only support one reading device over the serial interface, the configured address is ignored.
6. In the **Reader** tab, set the **Reader one door** to Assembler (Unlock to Initiate).
7. Scroll down to the **Reader credential match types** section. Click **Add** and select the Assembler PIN Code credential type. Click **OK**.
8. Save the smart reader record.

Adding the Door Type

A door type must be programmed to allow users to 'unlock' the assembler with their PIN codes.

1. Navigate to **Programming | Door types**.
2. Add a new door type with the name Assembler PIN Code.
3. Set the **Entry reading mode** to Custom.
4. Under the **Entry credential types** heading, click **Add**. Select the Assembler PIN Code credential type and click **OK**.
5. Click **Save**.
6. Navigate to **Programming | Doors** and select Assembler (Unlock to Initiate).
7. Set the **Door type** to Assembler PIN Code.
8. Click **Save**.

Assigning Access to Users

To activate the assembler, users must have access granted in their access levels, and a unique PIN code credential.

To grant access to the assembler, navigate to **Users | Access Levels** and assign the Assembler (Unlock to Initiate) door to all access levels that require access.

To assign user credentials:

1. Navigate to **Users | Users**.
2. Select or add a user and scroll down to the **Credentials** section.
3. In the Assembler PIN Code row, enter a PIN code for the user (e.g. 1234).
4. Click **Save**.

Appendix: Configuring Custom Wiegand Credential Types

The Wiegand credential type allows you to program custom card formats, enabling the controller or reader expander to recognize card formats which are not included in the standard programming.

The **Custom reader format** tab in the controller programming also allows you to program custom card formats, however this only allows one custom format to be configured per controller. The credential types feature allows you to program multiple custom card formats.

To create this credential type, you must enter a **Wiegand or TLV format** to define how incoming data should be interpreted. The example below shows how to define the standard HID 34 bit format.

```
# 34bit
#
# Variables
A,FACILITY,16,MSB,BIN
B,CARD,16,MSB,BIN
# Format
PAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBP
# Parity
EXXXXXXXXXXXXXXX.................
.................XXXXXXXXXXXXXXXO
```

Notes on the format definition:

- Each line in the format definition should be separated by a single or double underscore. For this example, you would enter:

  ```
  #34bit__A,FACILITY,16,MSB,BIN__B,CARD,16,MSB,BIN__
  PAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBP__EXXXXXXXXXXXXXXX.................._
  _.................XXXXXXXXXXXXXXXO
  ```

- Lines beginning with **#** are for human readability only and are ignored by the controller.
- The **Preceding characters**, **Trailing characters**, **Prefix** and **Case sensitive** fields are not applicable for this format.

Variables

Variables must be defined for both the facility and card codes. In this example:

```
A,FACILITY,16,MSB,BIN
```

- The first value (**A**) is the symbol that is used in the format definition to describe the relevant bit positions. The symbol can be anything other than **#**, **P**, **X**, **.** or **,** but would typically be an early letter of the alphabet.
- The second value (**FACILITY**) defines whether the symbol refers to a facility or card code. Valid values are **FACILITY** or **CARD**.
- The third value (**16**) defines the number of bits that make up the code.
- The fourth value (**MSB**) defines whether the run of bits will be interpreted most significant bit first or least significant bit first. Valid values are **MSB** (most significant bit first) or **LSB** (least significant bit first).
- The fifth value (**BIN**) defines the structure of the bit pattern and whether it should be interpreted as a binary number or as a binary coded decimal number. Valid values are **BIN** (binary number) or **BCD** (binary coded decimal).

Format

The format defines where bits are located in the Wiegand bit stream. In this example:

```
PAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBP
```

- Each letter corresponds to a single bit that is received in the Wiegand bit stream.
- **P** denotes a parity bit.
- **A** or **B** denotes a bit corresponding to one of the previously defined **variables**. The letter used will depend on the symbol defined for each variable.
- In the example shown above, the bit stream is defined as having one parity bit, followed by 16 facility code bits, followed by 16 card code bits, followed by one parity bit.

Parity

Finally, the parity bits are defined. For example:

```
EXXXXXXXXXXXXXXXX.................
```

- Each letter/symbol corresponds to a single bit that is received in the Wiegand bit stream.
- **O** or **E** defines the position of an **odd** or **even** parity bit.
- **X** defines the position of a bit to be included in the parity calculation.
- **.** defines the position of a bit that is not included in the parity calculation.
- Multiple lines may be required to cover multiple parity bits.

Including more than one **O** or **E** in a single line of parity calculation definition will result in incorrect interpretation of the bit stream.

Fixed Bits

- If the card contains fixed (always present) bits then these can be defined in a similar way to the card and facility bits using a third entry:

  ```
  C,FIXED,4,MSB,BIN
  ```

  The number, **4** in this case, is the decimal value of the fixed bit stream.

  The following example defines the **Infinity 37 bit** Wiegand format which contains the fixed bits **010** in the second, third and fourth bit positions. The binary value **010** equals the value **4** in decimal notation.

  ```
  #Infinity37bit__A,FACILITY,12,MSB,BIN__B,CARD,19,MSB,BIN__
  C,FIXED,4,MSB,BIN__PCCCCAAAAAAAAAAAABBBBBBBBBBBBBBBBBP_
  _.X.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.E__
  O.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.X.
  ```

- Alternatively, fixed bits can be represented directly in the format string. An alternate representation of the Infinity 37 bit Wiegand format would be:

  ```
  #Infinity37bit__A,FACILITY,12,MSB,BIN__B,CARD,19,MSB,BIN__
  P0100AAAAAAAAAAAABBBBBBBBBBBBBBBBBP_
  _.X.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.E__
  O.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.XX.X.
  ```

Configuring a Descramble Format

Some proprietary 125kHz credentials send card data in a scrambled order. This is common with PSK (phase-shift keying) card formats and makes it difficult to transition legacy sites which use these credentials to new card readers. However, you can program a custom Wiegand credential type which allows the controller to interpret these credentials even if the card reader cannot.

Contact ICT Technical Support for assistance with this feature.

Designers & manufacturers of integrated electronic access control, security and automation products.
Designed & manufactured by Integrated Control Technology Ltd.