AN-281

Deploying Protege GX on Microsoft Azure

Application Note

The specifications and descriptions of products and services contained in this document were correct at the time of printing. Integrated Control Technology Limited reserves the right to change specifications or withdraw products without notice. No part of this document may be reproduced, photocopied, or transmitted in any form or by any means (electronic or mechanical), for any purpose, without the express written permission of Integrated Control Technology Limited. Designed and manufactured by Integrated Control Technology Limited, Protege® and the Protege® Logo are registered trademarks of Integrated Control Technology Limited. All other brand or product names are trademarks or registered trademarks of their respective holders.

Last Published: 27-May-22 03:31 PM

Contents

Introduction

This application note describes the additional configuration required to deploy Protege GX on the Microsoft Azure cloud platform. The Protege GX installer currently only supports local deployment; therefore, manual configuration is required to complete deployment to an Azure server.

Several parts of the system can be installed and run on Azure:

- The services can be run on an Azure-hosted virtual machine.
- The ProtegeGX and ProtegeGXEvents databases can be uploaded as Azure SQL databases.
- The Protege GX SOAP Service and Web Client can be run as Azure app services.

This application note describes the process for deploying both new and existing Protege GX installations on Azure. It also includes the additional configuration steps required to deploy the Protege GX SOAP Service and Web Client as Azure app services.

Assumed Knowledge

Deploying Protege GX to Azure is a highly technical process. This guide assumes:

- Good familiarity with deploying Protege GX, including the Protege GX SOAP Service and Web Client.
- Basic familiarity with the Azure cloud platform.
- A high level of technical competence.

Requirements

Software Versions

You may deploy either an existing or a new Protege GX installation to Azure. This application note assumes that the Azure-hosted installation will utilize TLS 1.2 to encrypt the connection between the Protege GX services (hosted on a virtual machine) and client connections. Therefore, the following software versions are required for any new or existing installations:

Software Component	Version	Notes
Protege GX	4.3.264.9 or higher	It is recommended that the most recent version of Protege GX is used for this deployment, as it becomes more difficult to update the databases once they are hosted on Azure (see page 10).
Protege GX SOAP Service	1.6.0.1 or higher	
Microsoft SQL Server	**32-bit installations:** 2012 SP4, 2014 SP2, or a later edition that supports TLS 1.2 **64-bit installations:** 2016 SP2 or a later edition that supports TLS 1.2	For the SQL server editions that support TLS 1.2, see the Microsoft Documentation. For the SQL server editions supported by Protege GX, see the Protege GX Server Installation Manual.

Licensing

For this deployment, you will require an active Microsoft Azure Cloud subscription with the following resources available:

- Virtual machines
- SQL databases

- Storage accounts
- Resource groups

  It is recommended that you create a single resource group dedicated to the Protege GX installation.

- App services (if the SOAP service and web client are to be hosted on Azure)

Deploying Protege GX to Azure

Before you begin, the following resources will be required in the Azure portal:

- A **resource group** to contain all of the resources used for the Protege GX deployment.
- A **virtual machine** on which Protege GX can be installed, assigned to the Protege GX resource group.
- A **storage account** with an **account kind** of BlobStorage, assigned to the Protege GX resource group.
- An **Azure SQL server** for the Protege GX databases, assigned to the Protege GX resource group.

Setting up the Virtual Machine for TLS

For this deployment, the Protege GX services will be installed on a virtual machine (VM) hosted on Azure. However, before installing Protege GX some setup is required to ensure that TLS is correctly configured on this VM.

The self-signed TLS certificate that is automatically generated by the Protege GX installer will only include hostnames and IP addresses that the machine knows about. However, when the machine is a VM running on a cloud platform it may be accessible via DNS hostnames or IP addresses that are not exposed to the VM itself. If the name on the certificate is not the same as the one a client is attempting to connect to, the client will abort the connection.

The easiest solution is to set the name of the VM to the VM's public DNS name **before** installing Protege GX with TLS. This will ensure that the installer will generate a TLS certificate using the name that clients will most often use to connect.

1. On the Azure-hosted virtual machine, open the **Windows Control Panel** and navigate to the **System** section.
2. Under the **Computer name, domain and workgroup settings**, click **Change settings**. Under the **Computer Name** tab, click **Change...**.
3. Set the **Computer name** to the first part of the public DNS name.
4. Click **More...** and set the **Primary DNS suffix** to the rest of the public DNS name (e.g. centralus.cloudapp.azure.com).
5. Click **Ok**, then **Ok** to save.

Installing Protege GX on a Virtual Machine

Regardless of whether there is an existing installation of Protege GX, it is necessary to install the software on a virtual machine. Client workstations will connect to the Protege GX services running on this VM, which in turn will be connected to the Azure-hosted databases.

Installing Protege GX on a virtual machine is much the same as installing on a standard computer. See the Protege GX Server Installation Manual for more detailed instructions. Ensure that the installation meets the following requirements:

- All software meets the version prerequisites defined above (see page 4).
- A local SQL server instance is installed on the VM.
- The Protege GX server is **included** in the installation.
- TLS 1.2 is **enabled**.
- The Protege GX ports are **open** to allow client connection. These must be configured in the Azure virtual machine's **Network Security Group** as well as the **Windows Firewall** on the VM itself.

After Protege GX is set up on the VM, it is recommended that you test connecting a client workstation to the Protege GX services to ensure there are no issues with the installation or the TLS configuration.

Uploading the Databases to Azure

The following procedures will guide you through the process of preparing the ProtegeGX and ProtegeGXEvents databases and uploading them from the local server to Azure SQL databases. It is possible to migrate either existing databases containing programming and events, or new, empty ones.

Note: These procedures do not need to be carried out on the VM used above.

If working on an existing installation, the following should be considered:

- The Protege GX services should be stopped to prevent new programming and events from entering the databases as they are being migrated. If not, the databases uploaded to Azure will be out of date.

  If you are stopping the services, ensure that a downtime period is scheduled.
- It may be preferable to restore a backup of the ProtegeGX database to another SQL instance and work from that rather than from the original database.
- Sometimes the ProtegeGXEvents is very large and creating or restoring a full backup is impractical. In these cases it is recommended that you use a blank events database for the initial transition to Azure, then migrate any required event data separately.

Decrypting the Databases

The current Protege GX installer process is designed to support local deployments. Since the Protege GX databases contain stored procedures using `WITH ENCRYPTION`, they cannot be directly exported to Azure. Therefore, the 'encrypted' procedures must first be decrypted using a tool such as dbForge SQL Decryptor, which can be downloaded for free from this link.

1. If you are starting with a backup, first restore it to a local SQL server instance. If starting with a live database, take a backup.
2. Stop the Protege GX services by stopping the update service.
3. Download and install the most recent version of dbForge SQL Decryptor. Run the application and connect to the local SQL instance.
4. Expand the **Database** node, right-click on the **ProtegeGX** database and select **Decryption Wizard**.
5. Set the **Output Type** to Decrypt in-place (alter objects).
6. Click **Execute**.
7. Repeat the above procedure for the **ProtegeGXEvents** database.

Exporting and Uploading the Databases

Once the stored procedures have been decrypted, the databases are ready to export to a BACPAC file.

Some procedures may differ slightly depending on your version of SQL Server.

1. Open SQL Server Management Studio and connect to the local instance.
2. Right click on the ProtegeGX database and select **Tasks > Export data-tier application**. Follow the prompts to export the database to a known location on your local machine.

   The **Deploy Database to Microsoft Azure SQL Database...** option may be used to automatically upload the databases to Azure. However, the manual method allows you to save a copy of the BACPAC files.
3. Repeat the above step for the ProtegeGXEvents database.
4. In the Azure portal, navigate to the **Storage Account** created above. Open the **Storage Explorer** page from the left menu bar.
5. Right click on the **BLOB CONTAINERS** node and click **Create blob container**. A single blob container may be used for both Protege GX databases.

IMPORTANT: This blob container **must not** be publicly accessible, as it contains the unencrypted databases.

6. Open the new blob container. Click **Upload** and browse to the ProtegeGX BACPAC file to upload it to the Container. Repeat this process for the ProtegeGXEvents BACPAC file.
7. Navigate to **SQL Databases**, select the Protege GX Azure SQL Server and click **Import Database** in the toolbar. Follow the prompts to import the BACPAC blob file into the database. Repeat for ProtegeGXEvents.
8. Once the databases have been successfully loaded, it is recommended that the uploaded blobs be deleted from the blob container to ensure that the unencrypted data is not available.

Adding a Service User

The Protege GX services should not access the Azure SQL server as the administrator account. Instead, they should use a dedicated account with minimal access rights. This is achieved by creating a contained database user which is limited to only accessing the relevant databases. See this page in the SQL help for more information.

1. In the Azure portal, navigate to **SQL Databases > ProtegeGX > Query Editor**.
2. Log in using the administrator account.
3. Run the following queries:

```
CREATE USER <username> WITH PASSWORD='<password>';
ALTER ROLE db_datareader ADD MEMBER <username>;
ALTER ROLE db_datawriter ADD MEMBER <username>;
GRANT EXECUTE TO <username>;
GRANT CONNECT TO <username>;
DBCC FLUSHAUTHCACHE;
```

   Replace the placeholders as follows:

   - `<username>`: The login name of the account, for example `application`.
   - `<password>`: A randomly-generated strong password (record this for use in the next stage). Avoid passwords containing single or double quotes.

4. Verify that the new login works by using it to connect to the Azure SQL Database via Microsoft SQL Server Management Studio.
5. Repeat the above procedure to add the user to the ProtegeGXEvents database.

Connecting the Protege GX Services to the Azure SQL Instance

Before starting the Protege GX services, the connection strings must be updated to point to the Azure SQL databases instead of the local ones created by the installer.

Navigate to the installation directory (by default **C:\Program Files (x86)\Integrated Control Technology\Protege GX**) and edit these config files: **GXSV.exe.config**, **GXSV2.exe.config** and **GXEvtSvr.exe.config**.

In each file, locate the `<connectionStrings>` node and replace its contents with the examples below. Replace `SQLSERVERNAME` with the name of your Azure server (can be found on the **Overview** page under **Server name**) and `SQLUSERNAME` and `PASSWORD` with the service credentials created above.

Stop the Protege GX services before editing the config files.

For GXSV.exe.config:

```
<add name ="MainConnection"
connectionString="Server=tcp:SQLSERVERNAME.database.windows.net,1433; Initial
Catalog=ProtegeGX; Persist Security Info=True; User ID=SQLUSERNAME;
```

```
Password=PASSWORD; MultipleActiveResultSets=False; Encrypt=True;
TrustServerCertificate=False; Connection Timeout=30;"></add>
```

```
<add name ="EventConnection"
connectionString="Server=tcp:SQLSERVERNAME.database.windows.net,1433; Initial
Catalog=ProtegeGXEvents; Persist Security Info=True; User ID=SQLUSERNAME;
Password=PASSWORD; MultipleActiveResultSets=False; Encrypt=True;
TrustServerCertificate=False; Connection Timeout=30;"></add>
```

For GXSV2.exe.config and GXEvtSvr.exe.config:

> The connection strings below are relevant for Protege GX software version 4.3.289.1 and later. For earlier versions, use the alternative connection strings provided below (see below).

```
<add name ="MainConnection" connectionString="Provider=MSDASQL; Extended
Properties=&quot; Driver={ODBC Driver 17 for SQL Server};
Server=tcp:SQLSERVERNAME.database.windows.net,1433; Database=ProtegeGX;
Persist Security Info=True; UID=SQLUSERNAME; PWD=PASSWORD;
MultipleActiveResultSets=False; Encrypt=True; TrustServerCertificate=False;
Connection Timeout=30;&quot;"></add>
```

```
<add name ="EventConnection" connectionString=" Provider=MSDASQL; Extended
Properties=&quot; Driver={ODBC Driver 17 for SQL Server};
Server=tcp:SQLSERVERNAME.database.windows.net,1433; Database=ProtegeGXEvents;
Persist Security Info=True; UID=SQLUSERNAME; PWD=PASSWORD;
MultipleActiveResultSets=False; Encrypt=True; TrustServerCertificate=False;
Connection Timeout=30;&quot;"></add>
```

In addition, update the following connection string in GXEvtSvr.exe.config only:

```
<add name ="MainConnectionCLI"
connectionString="Server=tcp:SQLSERVERNAME.database.windows.net,1433; Initial
Catalog=ProtegeGX; Persist Security Info=True; User ID=SQLUSERNAME;
Password=PASSWORD; MultipleActiveResultSets=False; Encrypt=True;
TrustServerCertificate=False; Connection Timeout=30;"></add>
```

Updating the Server Names

As usual after restoring a database backup to a different location, the computer name must be updated in the client UI.

1. Start the Protege GX services and log in to the client.
2. Navigate to **Global | Download Server** and update the **Computer Name** to the VM machine name.

   > This should be the first part of the fully qualified domain name.

3. Repeat the above in **Global | Event Server**.

Connection Strings prior to 4.3.289.1

In Protege GX versions prior to 4.3.289.1, the following connection strings should be used for GXSV2.exe.config and GXEvtSvr.exe.config:

```
<add name ="MainConnection" connectionString="Provider=MSOLEDBSQL;
Server=tcp:SQLSERVERNAME.database.windows.net,1433; Initial
Catalog=ProtegeGX; Persist Security Info=True; User ID=SQLUSERNAME;
Password=PASSWORD; MultipleActiveResultSets=False; Encrypt=True;
TrustServerCertificate=False; Connection Timeout=30;"></add>
```

```
<add name ="EventConnection" connectionString=" Provider=MSOLEDBSQL;
Server=tcp:SQLSERVERNAME.database.windows.net,1433; Initial
```

```
Catalog=ProtegeGXEvents; Persist Security Info=True; User ID=SQLUSERNAME;
Password=PASSWORD; MultipleActiveResultSets=False; Encrypt=True;
TrustServerCertificate=False; Connection Timeout=30;"></add>
```

Software Updates

For the most part, the update process for Protege GX is the same as for a normal deployment. However, since the Protege GX installer currently does not natively support deployment to Azure, database upgrades must be applied to the Azure-hosted databases manually. The main database must be updated frequently, and the events database very rarely.

SQL script(s) for upgrading the database will be supplied by ICT on request.

> The installer still requires a local database instance in order to function.

1. Since database upgrades cannot be downgraded/rolled back, take a backup of the databases before applying any updates in case a rollback is required. Once the software update is confirmed to be working correctly, these backups can be deleted.
2. The Protege GX config files may be overwritten by the installer. Therefore, make a copy of the existing config files before beginning the update process. This will allow you to replicate the `<connectionStrings>` nodes should they be deleted (see page 8).
3. Stop the Protege GX services by stopping the update service.
4. Run the installer for the new Protege GX version. Complete the installation as normal.
5. Open the Azure-hosted Protege GX server in SQL Server Management Studio. Apply the upgrade scripts to the database(s) by selecting **File > Open** and opening the script file(s).
6. Update the `System.GXMigratedVersionTo` column to the current version that has been installed. In the Azure Portal, navigate to the **Query editor** for each database and enter the following query, where `<versionnumber>` is the new version number in full (e.g. 4.3.264.9):

   ```
   UPDATE System SET GXMigratedVersionTo='<versionnumber>'
   ```

   > If this step is not completed, the download and event services will fail to start correctly and the controllers will remain offline.

7. Restart the Protege GX services.

Additional Configuration for TLS

This section touches on some common issues that may be encountered when establishing TLS on a cloud-hosted installation.

Custom Certificates

If desired, you may replace the auto-generated TLS certificate with a custom certificate. This must be installed to the local machine's Personal store.

1. Open the Microsoft Management Console and add the Certificates snap-in for the local machine and add the Certificates snap-in:
   - Press **Windows + R**, type **mmc.exe** and press Enter to open the Microsoft Management Console.
   - Select **File > Add/Remove Snap-In...**.
   - Select **Certificates** and click **Add**.
   - Select **Computer account** and click Next, then select **Local computer** and click Finish.
   - Click Ok.
2. Browse to **Personal\Certificates**.
3. Add the new certificate by right-clicking and selecting **All Tasks > Import**, then completing the Certificate Import Wizard.
4. Open the new certificate (by double-clicking on it), navigate to the **Details** tab and take note of the certificate's **Thumbprint**.
5. Once the new certificate is available, the data service configuration must be updated to point to the new certificate. Open **GXSV.exe.config** and update the Certificate **Thumbprint** at the following node (near the bottom of the file):

   ```
   configuration/system.serviceModel/behaviors/serviceBehaviors/behaviour
   [@name="md"]/serviceCredentials/serviceCertificate@findValue
   ```

Example:

```
<serviceCredentials>
    <serviceCertificate storeLocation="LocalMachine" storeName="My"
findValue="057ac578cf3804d0f6eac366088a3dd51246e1c4"
x509FindType="FindByThumbprint" />
</serviceCredentials>
```

Custom Wildcard Certificates

It is possible to install custom wildcard TLS certificates in the same way as the standard custom certificates above. However, due to bugs in the Windows Communication Foundation (the service framework used by the Protege GX Data Service), some additional configuration is required to install a wildcard certificate.

In particular, it is necessary to change the hostname in **GXPI.exe.config** and **GXRpt.exe.config** from localhost to the hostname you will actually be connecting to.

This should be completed for both config files for **each client installation**. However, the same configuration files can simply be copied to other machines as necessary.

The following sections need to be updated in each file:

- `configuration/system.serviceModel/client/endpoint@address` - this should be the full hostname that you are actually connecting to.

- configuration/system.serviceModel/client/endpoint/identity/dns@value - this should be the first entry listed in the certificate's Subject Alternative Names section.

Example:

```
<endpoint address="net.tcp://security.other.domain.bits:8000/GXSV/GXService"
behaviorConfiguration="md0" binding="netTcpBinding"
bindingConfiguration="Binding1" contract="ServiceReference2.IGXService">
    <identity>
        <dns value="*.other.domain.bits" />
    </identity>
</endpoint>
<endpoint address="net.tcp://security.other.domain.bits:8010/GXSV/GXService2"
behaviorConfiguration="md0" binding="netTcpBinding"
bindingConfiguration="Binding1" contract="GXServiceRef2.IGXService2">
    <identity>
        <dns value="*.other.domain.bits" />
    </identity>
</endpoint>
```

After this change has been made, when connecting from the client it is necessary to leave the **Server** field on the login page blank. If this field is filled the client will fail to connect correctly.

Configuring the Protege GX SOAP Service

This section describes the additional configuration required to deploy the Protege GX SOAP Service as an Azure App Service (recommended) rather than hosting it on the Server Virtual Machine itself.

The following procedure requires an FTP Client to transfer Protege GX SOAP Service files to the Azure app service web-root. A good free FTP application is Filezilla, which can be downloaded from this link.

1. Install the Protege GX SOAP Service locally. Ensure that you install with **TLS enabled**.

   On the **Customize WCF TCP/IP Port** page, point the SOAP service to the Azure-hosted data service:

   - **Protege GX Data Server installed PC name**: your-vm.region.cloudapp.azure.com (Public DNS of the Server VM)
   - **Data Server Port**: 8000 (or as configured)
   - **Report Server Port**: 8010 (or as configured)

   For instructions on installing the SOAP Service, see the Protege GX SOAP Service Installation Manual.

2. Locate and edit the following file: **C:\inetpub\wwwrootProtegeGXSOAPService\Web.config**.
   - Under `/configuration/system.serviceModel/`, comment out or remove this line: `<serviceHostingEnvironment multipleSiteBindingsEnabled="true" />`.
   - If you are using TLS security (recommended) on the data service:
     - Under `/configuration/system.serviceModel/client/endpoint@address`, set the endpoint hostname to the data service VM's DNS-accessible name.
     - Under `/configuration/system.serviceModel/client/endpoint/identity/dns@value`, set the endpoint DNS-identity to one of the 'Subject Alternative Names' in the data service's TLS Certificate.
     - Check that the following node exists: `/configuration/system.serviceModel/behaviors/endpointBehaviors/behavior[@name=md0]/clientCredentials/serviceCertificate/authentication`. This will be required to allow connection unless a custom trusted SSL certificate is being used by the data service.

       If this node does not exist, reinstall the SOAP service and ensure that you enable TLS.

3. In the Azure portal, create a new **App Service** to host the SOAP service.
4. Navigate to **App Services > SOAP Service > Deployment Center**. Locate the **FTP(S) endpoint** for the app service:
   - If the deployment center has not been fully configured, select the **FTP** option and click the **Dashboard** button at the bottom of the page.
   - If it has been configured, click **Deployment Credentials** in the top toolbar and select the **FTP/FTPS** tab.
5. Use an FTP client to connect to the provided FTP endpoint, using either the **App Credentials** or your personal **User Credentials**. The latter can be configured when viewing the deployment credentials, but may take a few minutes to sync after saving.
6. Upload the files from C:\inetpub\wwwrootProtegeGXSOAPService to the root of the app service.

SOAP App Service Configuration

In the Azure portal, navigate to **App Services > SOAP Service > Configuration**. In the **General settings** tab, set the following:

- **Stack**: .NET
- **.NET Framework Version**: V4.7

- **Platform**: 32-bit
- **HTTP Version**: 1.1
- **Always on**: On

In the **Path mappings** tab, add the following mappings under **Virtual applications and directories**:

Virtual path	Physical path	Type
/	site\wwwroot	Application
/ProtegeGXSoapService	site\wwwroot\ProtegeGXSoapService	Application

To set the **Type** to Application, uncheck the **Directory** checkbox.

From the sidebar, select the **TLS/SSL Settings** page. It is recommended that you implement the following settings:

- **HTTPS Only**: On
- **Minimum TLS Version**: 1.2

Troubleshooting

Access is Denied

- When attempting to log in to the web client, the error 'Access is denied' is displayed.
- checklogin.php returns 'ERROR Access is denied.'

Troubleshooting:

- Ensure that the Protege GX services are running.
- The SOAP DNS-identity or hostname may not match the data service certificate (may have been left as 'localhost'). Return to step 2 above (see previous page).

Configuring the Protege GX Web Client

The following procedure requires an FTP client to transfer Protege GX Web Client files to the Azure App Service web-root. A good free FTP application is Filezilla, which can be downloaded from this link.

For instructions on installing the web client, see the Protege GX Web Client Installation Manual.

1. Install the Protege GX Web Client locally. On the **Customize WCF TCP/IP Port** page, set the SOAP hostname and port to point to the Azure SOAP instance:
   - **Protege GX SOAP Interface installed PC**: your.azure-soap.host
   - **Protege GX SOAP Interface running port number**: 443
2. In the Azure portal, navigate to **App Services > Web Client > Deployment Center**. Locate the FTP(S) endpoint for the web client:
   - If the deployment center has not been fully configured, select the **FTP** option and click the **Dashboard** button at the bottom of the page.
   - If it has been configured, click **Deployment Credentials** in the top toolbar and select the **FTP/FTPS** tab.
3. Use an FTP client to connect to the provided FTP endpoint, using either the **App Credentials** or your personal **User Credentials**. The latter can be configured when viewing the deployment credentials, but may take a few minutes to sync after saving.
4. Upload the files from C:\inetpub\wwwrootProtegeGXWebClient to the root of the app service.

Web Client App Service Configuration

In the Azure portal, navigate to **App Services > Web Client > Configuration**. In the **General settings** tab, set the following:

- **Stack**: .PHP
- **PHP Version**: 7.3
- **Platform**: 32-bit
- **HTTP Version**: 1.1
- **Always on**: On

In the **Default documents** tab, set **index.php** as a default document.

Troubleshooting

Error - SOAP Connection Failed

- Arises when attempting to connect to the SOAP service using the web client.
- Error message contains the following:

```
SOAP-ERROR: Parsing WSDL: Couldn't load from 'https://your.azure-
soap.host/ProtegeGXSOAPService/service.svc?wsdl': failed to load external
entity "https://your.azure-
soap.host/ProtegeGXSOAPService/service.svc?wsdl"
```

Troubleshooting:

- Ensure that the SOAP service is not stopped.
- Verify that the SOAP service is working by connecting to its URL.
- Navigate to the web client installation directory and check the file include/soap.connect.php. Ensure that the SOAP service URL is defined correctly (this should end with `/ProtegeGXSOAPService/service.svc`).

Configuring the Single Record Download Service

If you are using the single record download service in an Azure-hosted installation, you also need to adjust the connection string for this service.

In the installation directory for Protege GX, open GXSV2B.exe.config. Locate the `<connectionStrings>` node and replace the `"Main"` connection string with the example below:

```
<add name="Main" connectionString="Trusted_Connection=no; User
ID=SQLUSERNAME; Password=PASSWORD; TrustServerCertificate=no; Encrypt=yes;
Server=tcp:SQLSERVERNAME.database.windows.net,1433; Database=ProtegeGX; max
pool size=2000;" />
```

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

Disclaimer: Whilst every effort has been made to ensure accuracy in the representation of this product, neither Integrated Control Technology Ltd nor its employees shall be liable under any circumstances to any party in respect of decisions or actions they may make as a result of using this information. In accordance with the ICT policy of enhanced development, design and specifications are subject to change without notice.