Skkynet Embedded Toolkit for Renesas Synergy

Quick Start

How to quickly get up and running with the Skkynet ETK for Renesas Synergy.

Overview

The Skkynet Embedded Toolkit (ETK) is a C library that allows the developer to quickly create applications that can send and receive data in real time to both to the Cogent DataHub® industrial middleware application and to the SkkyHub™ cloud service.

Data Data is identified as (name, value, quality, timestamp) tuples, allowing your application and all cooperating applications to interact with your data by name rather than by hardware address.

Cogent DataHub Data from the ETK can be transmitted on your local LAN to the Cogent DataHub industrial middleware, which can automatically convert it to OPC, DDE, ODBC, E-mail, TCP, Modbus or custom formats. The Cogent DataHub can also trigger scripts and actions based on data changes from your application, and update information on any industrial HMI. In effect, with the DataHub your application immediately becomes a first-class participant in any industrial control system.

SkkyHub In addition, your data can be transmitted via the Internet to the SkkyHub cloud service, where it can be accessed remotely by any authorized user. This allows you to monitor and control your embedded device without presenting an attack surface to the Internet. The SkkyHub service provides everything you need to not only connect your device, but also to create web-accessible graphical HMIs for your service engineers, analysts and end users.

Developer-friendly The Skkynet ETK provides a developer-friendly method to establish and maintain a TCP socket connection and set of data points, distributing changes to these data points among connected client and server applications. Developers using the ETK can also use a thread-safe API within the ETK to create their own processing threads that can write data and subscribe to data point changes from the ETK engine.

More Help If questions come up that are not covered by this documentation, please feel free to contact Skkynet at our website, by email: info@skkynet.com, or by phone: +1 905 702 7851.

Features include:

- Full-time connectivity to the server for minimum latency
- Transfer latencies only microseconds above network ping time
- Event-driven communication - only data changes are transmitted
- Bi-directional communication, allowing both monitoring and control
- Publish/subscribe data model
- Server-side data discovery - no server configuration necessary
- Efficient structured text data format for low bandwidth usage
- Multiple ingoing and outgoing data sockets on a single thread
- Integrated timers with round-robin sharing with socket data
- Automatic resynchronization when connection is lost and recovered
- Automatic connection retries
- Runs on architectures with no floating point support
- Small footprint
- Thread-safe API for developer threads to emit and consume data
- Optional built-in WebSocket support for traversing proxies

- Optional built-in scripting for powerful local processing
- Optional support for SSL
- Optional support for IPV6
- Optional support for Modbus master to multiple slaves

Supports the following targets:

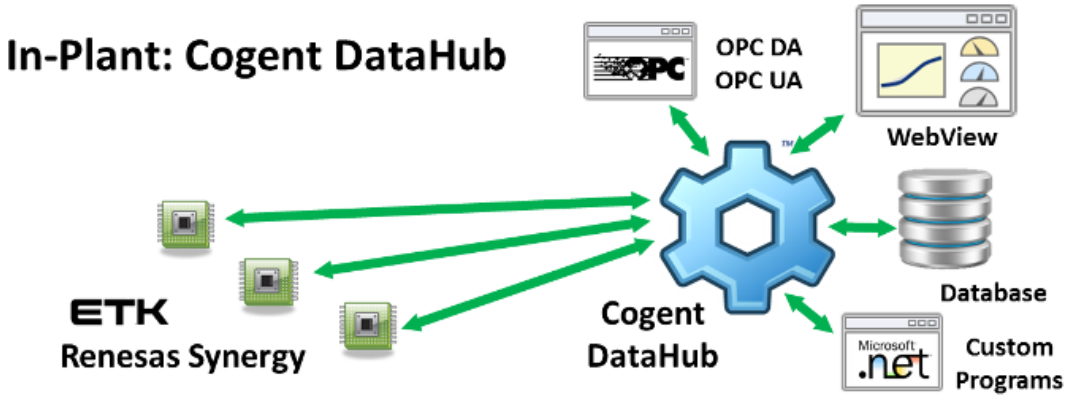
- Linux (ARM, x86)
- uClinux
- ThreadX (for Renesas Synergy)
- Windows (Cygwin)
- Windows (Visual Studio)
- QNX
- Portable to most platforms offering a BSD socket API

DataHub and SkkyHub

The Skkynet ETK is designed to connect to the Cogent DataHub® software and the SkkyHub™ cloud service.

Using Cogent DataHub

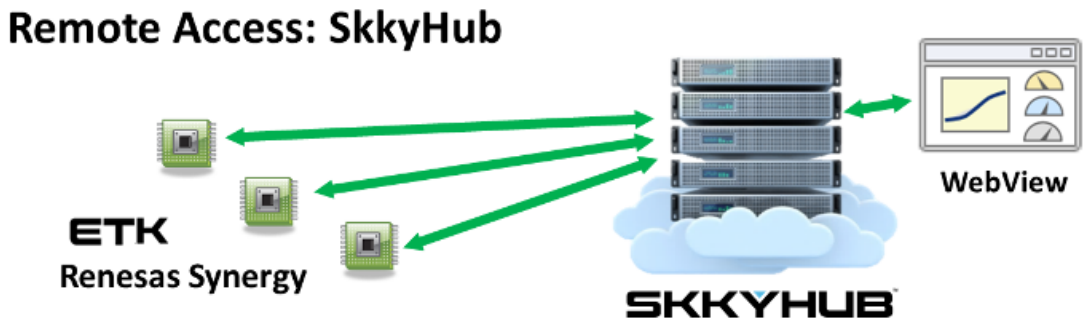
The Cogent DataHub is industrial middleware that accepts connections from the Skkynet ETK and integrate its data into any industrial control system. The DataHub provides bidirectional, real-time communication between OPC servers and clients, ODBC-compliant databases, Excel spreadsheets, Modbus devices, custom programs, and more. It can also trigger scripts and actions based on data changes from your application, and update information on any industrial HMI.



The Cogent DataHub runs on any modern Windows platform. You can download a free demo version from the Cogent DataHub website. The product documentation provides complete instructions on how to use the DataHub, and a series of how-to videos help you get quickly up to speed.

Using SkkyHub

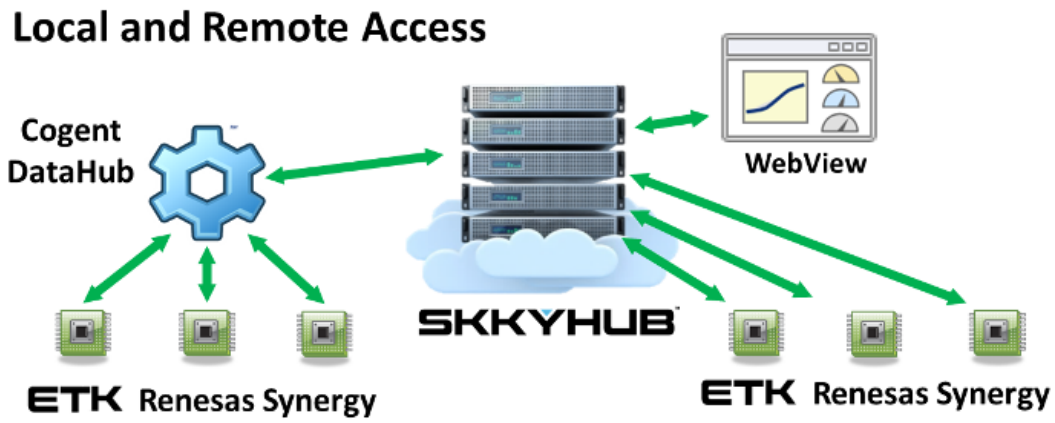
The SkkyHub cloud service allows you to securely monitor and control your embedded device from anywhere in the world in real time. The ETK's outbound-only connection architecture allows it to connect securely to SkkyHub, while presenting zero attack surface to the Internet. In this way, SkkyHub supports real-time M2M connectivity, as well as a web-based HMI that allows authorized access to application data.



You can test a connection with SkkyHub in demo mode, as demonstrated in our sample application test. To use SkkyHub with your application on an on-going basis, you will need a SkkyHub account. There are several different service types available to accommodate your needs, described in detail on the Skkynet website. The SkkyHub documentation provides the information you will need to access the service, administer your account, use Skkynet WebView, and more.

Using Them Together

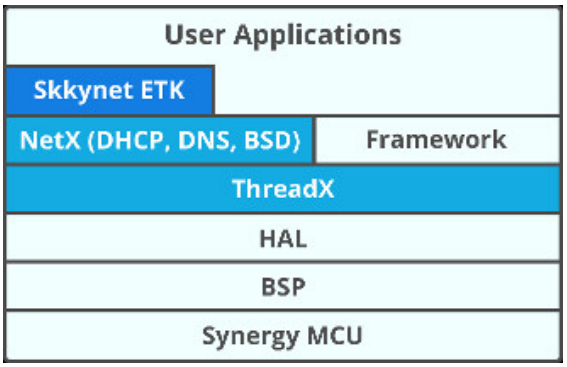
The Cogent DataHub and SkkyHub are both fully compatible with each other, and with the Skkynet ETK. Therefore you can use all of them in one fully integrated system.



For example, the Cogent DataHub can be used to collect data from a number of devices running the ETK, and then send the consolidated data to SkkyHub. That data can be integrated with other data collected directly from the ETK, and any combination of it made available in WebView.

Getting Started

The Skkynet ETK is an application-level library that depends only on the ThreadX operating system, the NetX networking layer and an Ethernet network driver. Information about memory requirements is available in the section ThreadX Memory Usage.



The Skkynet ETK is available for the Renesas Synergy embedded development platform:

http://am.renesas.com/products/embedded_systems_platform/synergy/index.jsp

To get started:

1. Install the e2 Studio development environment from the Renesas Gallery web site.
2. Follow the instructions there to create a membership and to download the following:
   a. e2 Studio (the development environment based on Eclipse, including an GNU ARM cross-compiler)
   b. SSP (The Synergy Software Package, including support for a variety of hardware, ThreadX operating system, NetX network stack, etc.)
3. Once installed, download the Skkynet ETK installer from the Renesas gallery **Software Addons** section and run the installer on your computer. When asked for an installation path, please indicate the top-level directory where you have installed e2 Studio.

Now you are ready to create a new project.

Creating a New Project

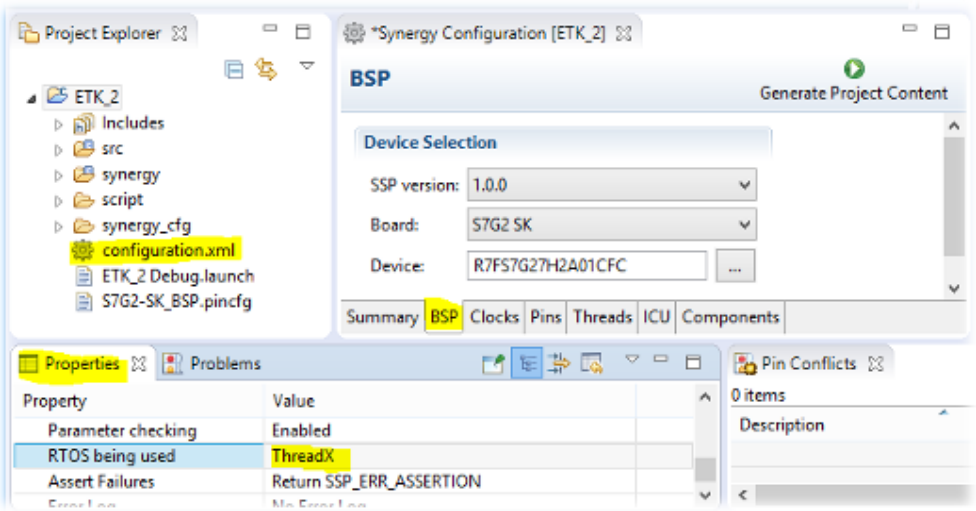
To create a Synergy project with the Skkynet ETK, follow these steps within e2 Studio:

1) Create a Project

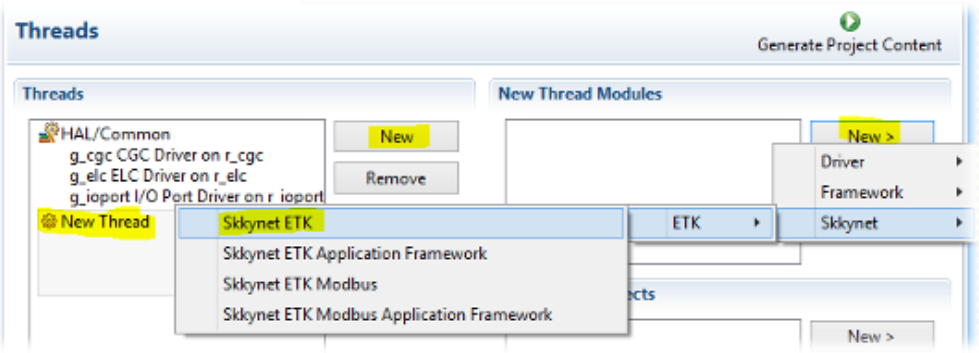
1. Ensure that the Skkynet ETK is installed on your system.
2. Select **File # New # Synergy Project**.
   a. Give your project a name.
   b. Select your License file.
   c. Press **Next**.
3. Select the target board:
   a. Select the board (like **S7G2 SK**).
   b. Press **Next**.
4. Select a project template:
   a. Select a project that includes your BSP (like **S7G2-SK BSP**).
   b. Press **Finish**.
5. Wait for the project to be created. You may get a message saying: "This kind of project is associated with the Synergy Configuration perspective. Do you want to open this perspective now?" Choose **YES** to go straight to the configuration, explained in the next step.

2) Configure the SSP

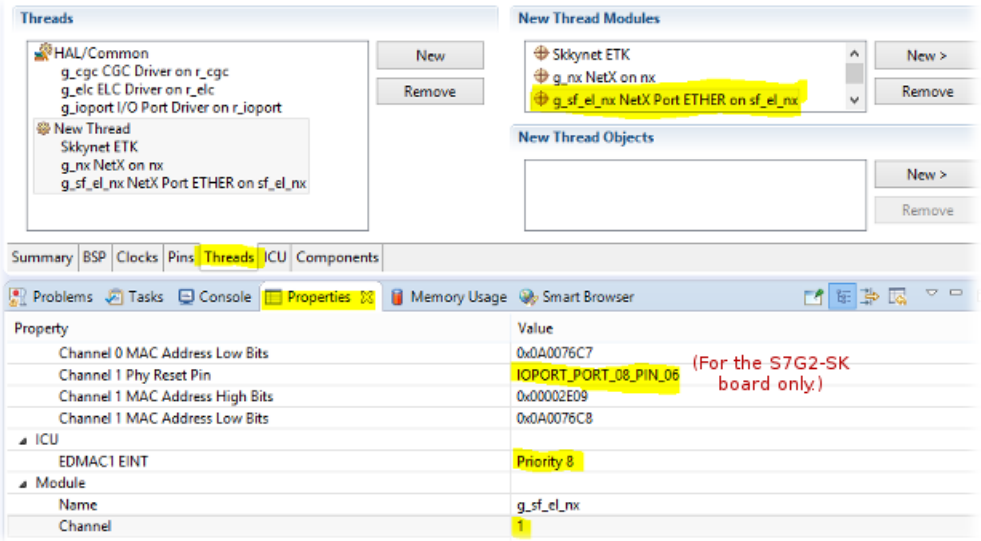
1. Double-click **configuration.xml** in the project source tree.
2. Configure the RTOS:
   a. Select the **BSP** tab at the bottom of the **Synergy Configuration** pane.
   b. Select the **Properties** tab at the top of the bottom pane of the e2 Studio window (beneath the Synergy Configuration pane).
   c. Scroll to the bottom of the Properties tab and change the setting:
      - **RTOS being used** to **ThreadX**



3. Create a thread:
   a. Select the **Threads** tab at the bottom of the Synergy Configuration pane.
   b. Press the **New** button in the center of the Threads pane.
   c. Press the **New >** button to the right of the New Thread Modules list:
      - Choose **Skkynet # ETK # Skkynet ETK**.

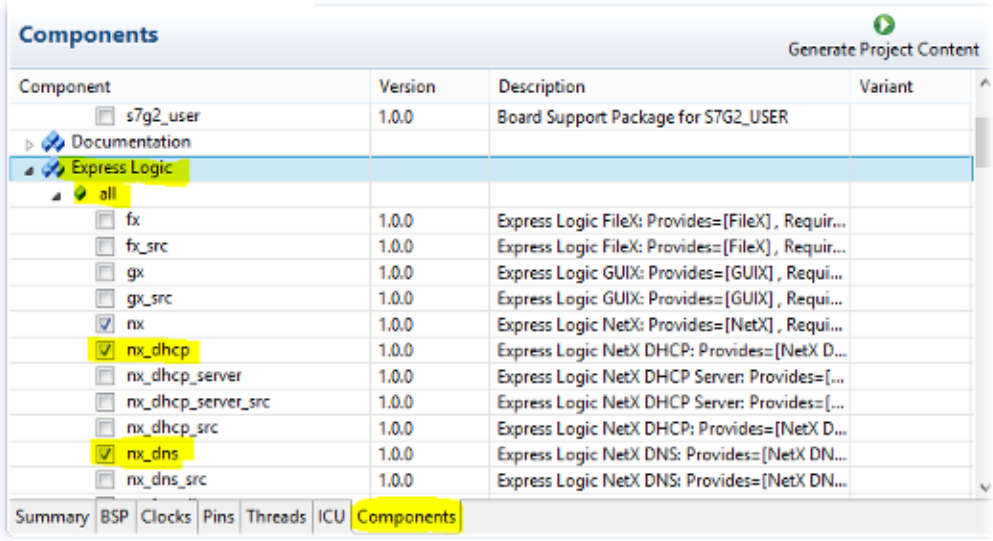


   d. Repeat the previous step for each of these, as needed:
      - **SkkynetApp** (optional - contains application template source)
      - **SkkynetModbus** (if you are using Modbus)
      - **SkkynetAppModbus** (optional - contains application template source for Modbus)
   e. Select the **SkkynetETK** item in the **New Thread Modules** list.
      - Select the **Properties** tab at the top of the bottom pane of the e2 Studio window
      - If you do not wish to use DHCP in your application, set "Use DHCP" to "No"
      - If you do not wish to use DNS in your application, set "Use DNS" to "No"
      - If you wish to change the size of the Skkynet ETK heap, set it here. The size of the heap will depend on the number of data points that your application uses. You will need at least 32K. You can find more information about memory usage here: ThreadX Memory Usage.
   f. In a similar way, press the **New >** button to the right of the New Thread Modules list, and choose:
      - **Framework # Networking # NetX on nx** (required)
      - **Framework # Networking # NetX Port ETHER on sf_el_nx**. (required)
   g. Select the **g_sf_el_nx NetX Port ETHER on sf_el_nx** item in the **New Thread Modules** list.
   h. Select the **Properties** tab at the top of the bottom pane of the e2 Studio window and change the settings:
      - **EDMAC1 EINT** to **Priority 8** (any number should do)
      - **Channel** to **1**
   i. If you are using the S7G2-SK board, select the **Properties** tab at the top of the bottom pane of the e2 Studio window and change the settings:
      - **Channel 1 Phy Reset Pin** to **IOPORT_PORT_08_PIN_06**



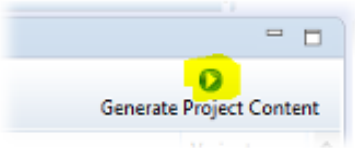
3) Configure the Skkynet ETK

1. Select the **Threads** tab at the bottom of the Synergy Configuration pane.
2. Select your newly created thread from the previous step.
3. Select **Skkynet ETK** in the New Thread Modules list:
   - Select the **Properties** tab at the top of the bottom pane of the e2 Studio window and change the setting:
     - **Modbus Support** to **0** or **1**, if you are using Modbus.
4. Select the **Components** tab at the bottom of the Synergy Configuration pane:
   - Ensure that the following components are selected:
     - **Express Logic # all # nx_dhcp**
     - **Express Logic # all # nx_dns**



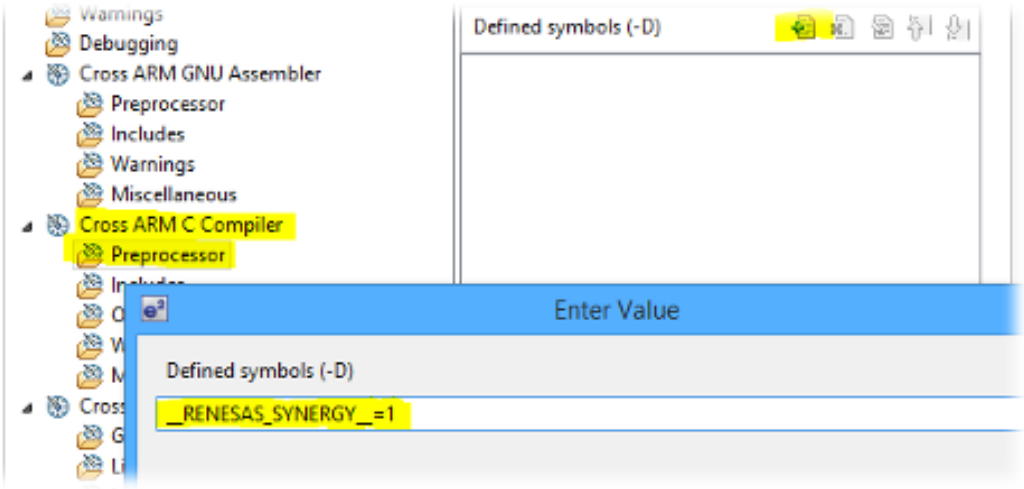
4) Generate the project content

- Press **Generate Project Content** in the top-right corner of the Synergy Configuration pane.

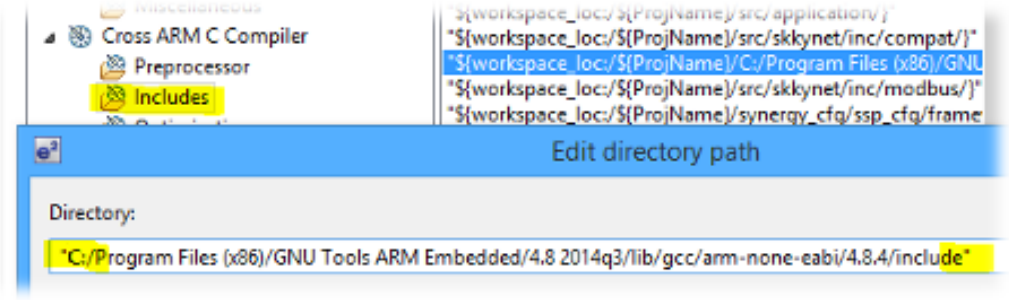


5) Configure the build environment

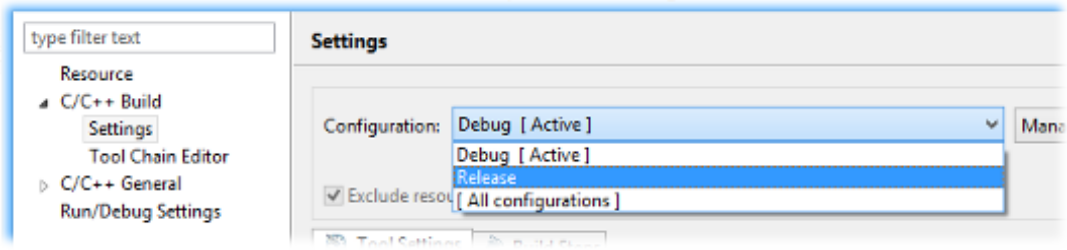
1. Find your project in the e2 stdio Project Explorer pane.
2. Right-click the project title and select **Properties**.
3. Expand **C/C++ Build** in the left-hand pane of the Properties dialog.
4. Select **Settings** within **C/C++ Build**.
5. Select **Cross ARM C Compiler** # **Preprocessor** in the right-hand pane (not Assembler or Linker).
6. In the **Defined Symbols** section, add this symbol (starting and ending with two _ characters):
   - **`__RENESAS_SYNERGY__=1`**



7. Select **Cross ARM C Compiler** # **Includes**.
8. Edit the include path beginning with **`${workspace_loc:/${ProjName}/C:/Program Files (x86)/...`** as follows:
   - Remove the string **`${workspace_loc:/${ProjName}/`** from the beginning of the path, leaving the opening double-quote.
   - Remove the character **`}`** from the end of the path, leaving the closing double-quote.



9. Repeat the modifications to the **Preprocessor** and **Includes** for the **Release** configuration, so that both **Debug** and **Release** configurations contain them.



10. Press OK to close the Properties dialog.

6) Build your project

- Right-click on your project title in the Project Explorer and select **Build Project**. If the Console output contains three lines similar to the following near the end, then your build was successful:

```
arm-none-eabi-size --format=berkeley "ETK_2.elf"
  text  data    bss    dec     hex filename
136704  2336 123540 262580   401b4 ETK_2.elf
```

You should now have a project that implements a connection to a DataHub or SkkyHub server. You will need to configure the IP address or domain name, TCP port and data points for your application. If you have included Modbus/TCP support then you will need to configure the Modbus slave IP and the mapping between I/O addresses and point names. The sample files contain some examples of both single and multi-threaded operation, along with simple interaction with the LEDs on the target board. Please refer to the Template Files documentation for details.

Now you can test the sample application.

Testing the Sample Application

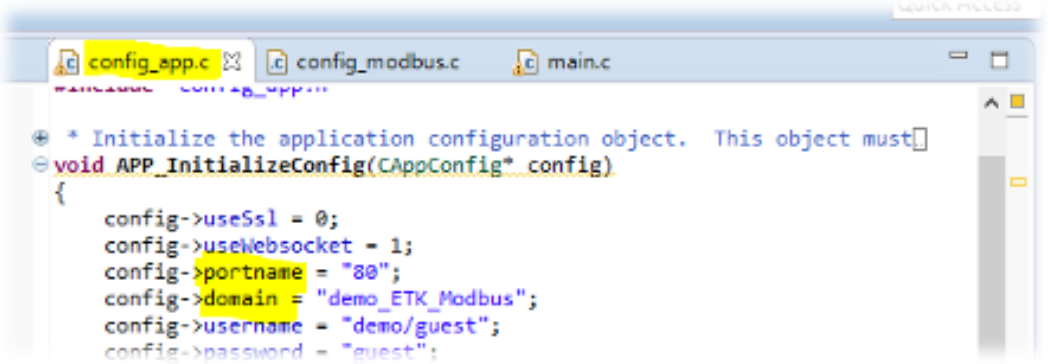
The sample application contains all of the required code to create a connection between your board and either a Cogent DataHub ® or a SkkyHub™ server. You will need to modify the configuration to match your application requirements. The simplest test is to install the Cogent DataHub on a PC on your local network, and to transmit data from your Synergy application to the DataHub. Once you are satisfied with your Synergy application you can sign up for an account on the SkkyHub cloud service and modify the target IP address and data domain in your application to send data to your cloud account.

1) Install and Configure Cogent DataHub

1. Download the Cogent DataHub from the Cogent DataHub home page and install it.
2. Configure the DataHub Web Server as documented. Mainly, you need to ensure that the **Act as a web server** option is checked. You may also need to change the port number if your PC is running software that uses port 80 (e.g., Skype or IIS).

2) Configure a DataHub Connection

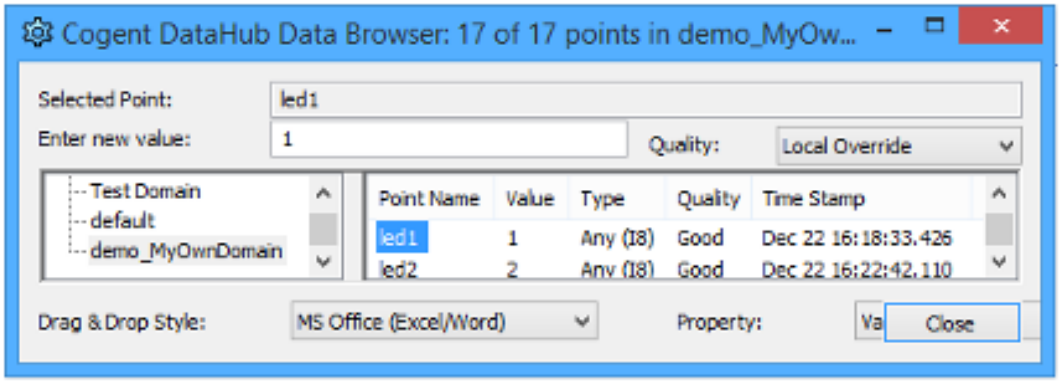
1. Open the file src/application/config_app.c in your e2 studio project.
2. Find the function APP_InitializeConfig, which sets the values of members of a CAppConfig. structure.



3. Modify these values to match your network:
   - hostname - the IP address or host name of the computer running the DataHub
   - portname - the HTTP server port setting of the DataHub (normally 80)
   - modbusHost - the IP address or host name of the Modbus slave device
   - modbusPort - the port number of the Modbus slave device (normally 502)
4. In addition, you should modify the domain to create a data domain (essentially a namespace) for your data. Use the default prefix, `demo_` and change the string `ETK_Modbus` to a name of your choice, like this: `demo_Charles_at_Acme` or similar. This name may be used later for testing on a public system, and you will get the best results if it is unique.
5. Open the file src/application/config_app.h in your e2 studio project.
   - If you are not using DHCP to assign an IP address, modify the definition for STATIC_SERVER_IP_ADDRESS to assign an IP address to this appliction.

- If you are not using DHCP, and you are using DNS, modify the definition for STATIC_DNS_SERVER_ADDRESS to the IP address of the DNS server. The address 8.8.8.8 is Google's public DNS.
- If you are not using DHCP, and you are on a network with a router, modify the definition for STATIC_IP_GATEWAY_ADDRESS to the address of your router.

6. Rebuild and run your Synergy application
7. Ensure that your Windows firewall settings allow an incoming connection on the DataHub HTTP server port (portname above).
8. Start the Cogent DataHub, and click the **View Data** button to open the Data Browser window.



9. In the left pane, click the data domain for your application. Among the points listed, you should see at least `led1` and `led2`.
   - The point `led1` corresponds to LED 1 on your board. Depending on the board you are using this value will affect the LED differently. For example, on the SK-S7G2 a value of `0` will turn the light on, and non-zero will turn it off. On the DK-S7G2 a value of `0` will turn the LED off and values of 1 through 3 will select the LED color. You can change the value of the point by clicking on the name `led1`, and typing in a `1` in the **Enter a new value** field above, then pressing Enter.
   - The point `led2` corresponds to LED 2 on your test board. Again, its behaviour depends on the board you are using.
   - Note: If you are using Modbus, you should see the default Modbus values appear in the Data Browser window as well.

If you are able to view and interact with your data in the Data Browser window, then you have successfully connected to the Cogent DataHub. Now you can continue, and configure a connection to SkkyHub.

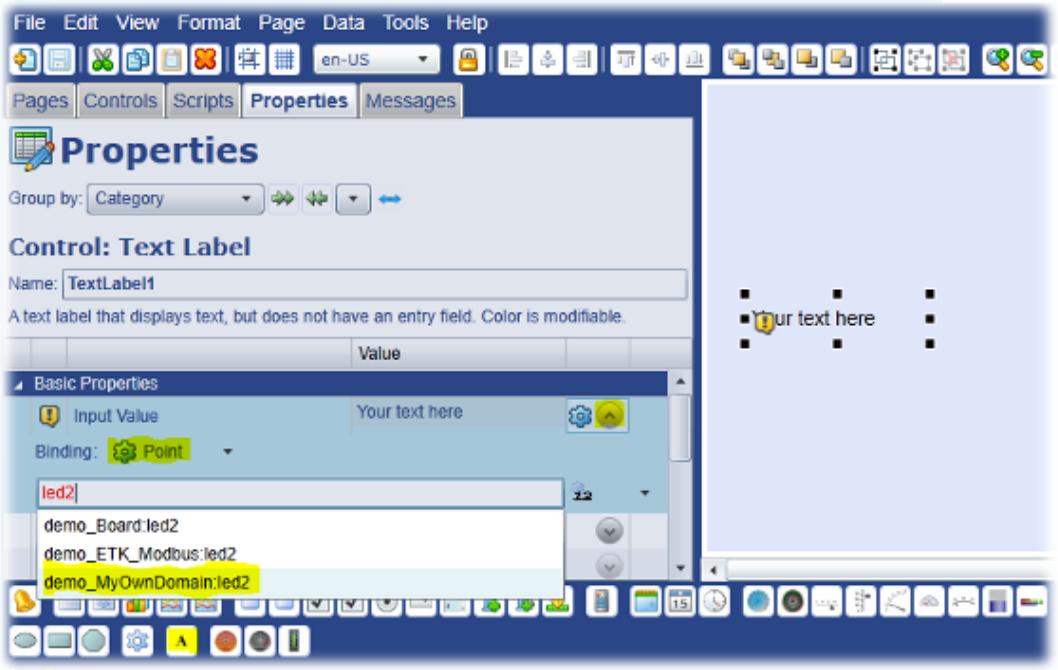
3) Configure a SkkyHub Connection

1. To connect to the SkkyHub service, open the file src/application/config_app.c in your e2 studio project.
2. Find the function APP_InitializeConfig, to set the values of the members of the CAppConfig structure.
3. Make the following changes:
   - hostname # `demo.skkynet.com`
   - portname # `80`

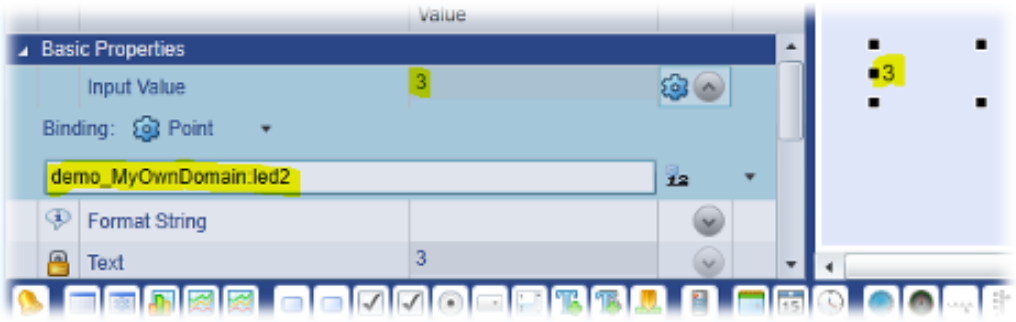
- domain # demo_*MyOwnDomain* (your domain name as specified previously)
- username # demo/guest
- password # guest

Note: If you already have a SkkyHub account, you can use your own user name, password, and data domain here. Otherwise, you need to use the guest account and demo_*your_domain* as described.

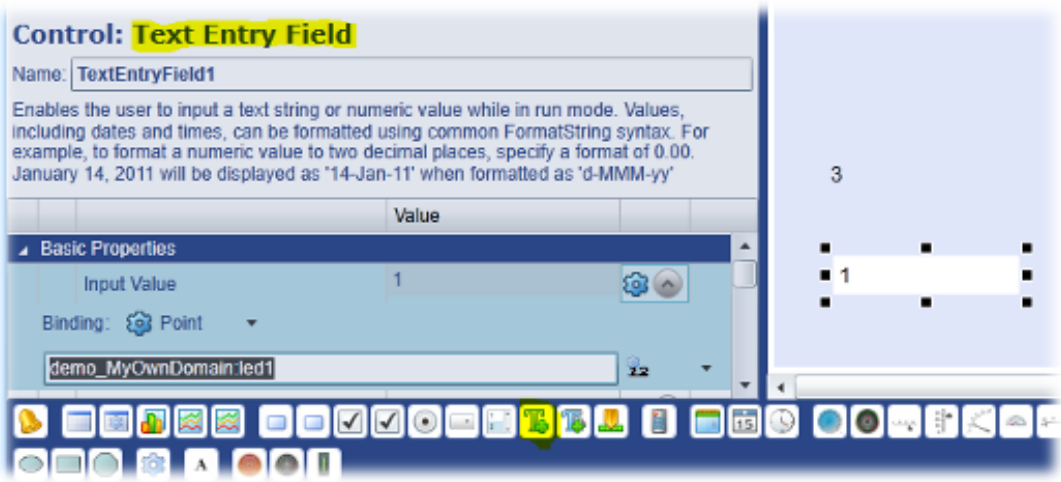
4. To check your data connection, open a browser and go to demo.skkynet.com.
5. Log in with the username Guest and password guest. This will open the Skkynet WebView interface.
6. Select **File** # **New** to open a new page.
7. Add a new Text Label control by clicking the "A" button in the controls list at the bottom of the window.
8. In **Basic Properties** # **Input Value**, select the arrow button to open the Binding entry field.
9. Select **Point**, and in the entry field, type led2. This will search the domain for all points containing the string led2.
10. Choose the string demo_*yourdomainname*:led2.



The value of led2 should appear in the control.

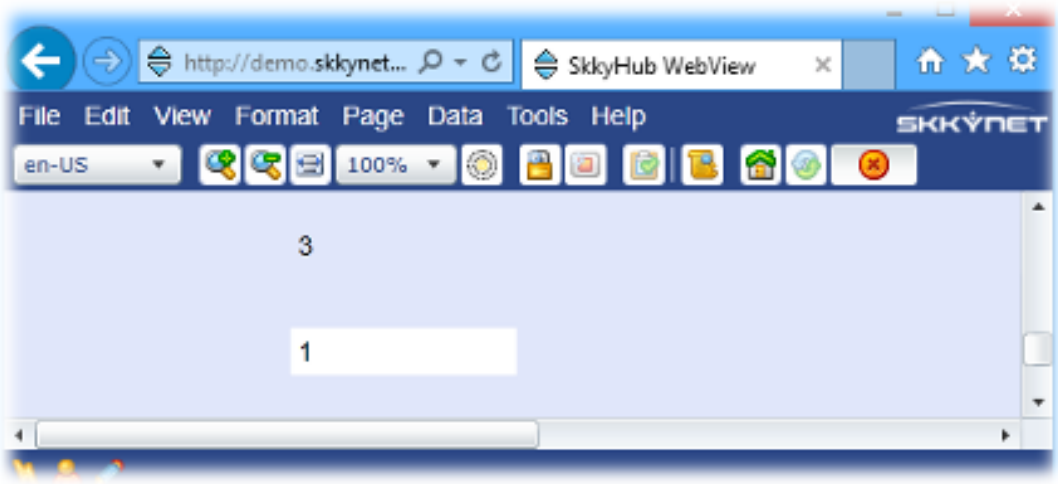


11. In the same way, add a new Text Entry Field control (a blue "T" with green plus sign) to the page, and link it to the data point `led1` in your data domain.



12. Click the green Run arrow at the top of the window.

This will put the page into Run mode, allowing you to change the value in the `led1` text entry field to demonstrate 2-way connectivity from SkkyHub to your device.



13. If you are able to see and interact with your data, then you have successfully connected to SkkyHub.

4) Next Steps

Now that you have tested your application with the DataHub and SkkyHub, you are ready to do any or all of the following, as explained in the relevant documentation:

- Configure more data points in config_points.c.
- Configure Modbus connections and I/O mappings in config_modbus.c.
- Configure user threads in config_threads.c.
- Configure timers in config_timers.c.
- Customize your application.
- Open a SkkyHub account. The demo account you have used for this test does not allow you to save pages or build an application. To do that, you will need a Skkynet account.