

# Logix Designer SDK Getting Results Guide



# Contents

- Overview.....7**
  - Considerations when using the Logix Designer SDK .....7
- Download projects to controllers.....8**
  - Download one project to multiple controllers..... 8
    - Example: Download one project to multiple controllers..... 8
  - Download multiple projects to multiple controllers.....10
    - Example: Multiple ACDs to multiple controllers.....11
  - Deploy a project and save it to an SD Card.....13
    - Example: Download a project and save to an SD card.....13
  - Download a project for automated testing.....14
    - Example: Provision and validate..... 15

# Important User Information

Read this document and the documents listed in the additional resources section about installation, configuration, and operation of this equipment before you install, configure, operate, or maintain this product. Users are required to familiarize themselves with installation and wiring instructions in addition to requirements of all applicable codes, laws, and standards.

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice.

If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.



**WARNING:** Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.

---



**ATTENTION:** Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

---

**IMPORTANT:** Identifies information that is critical for successful application and understanding of the product.

---

These labels may also be on or inside the equipment to provide specific precautions.



**SHOCK HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

---



**BURN HAZARD:** Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

---



**ARC FLASH HAZARD:** Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

---

The following icon may appear in the text of this document.



**Tip:** Identifies information that is useful and can help to make a process easier to do or easier to understand.

---

Rockwell Automation recognizes that some of the terms that are currently used in our industry and in this publication are not in alignment with the movement toward inclusive language in technology. We are proactively collaborating with industry peers to find alternatives to such terms and making changes to our products and content. Please excuse the use of such terms in our content while we implement these changes.

# Additional resources

These documents contain additional information concerning related Rockwell Automation products.

Resource	Description
Industrial Automation Wiring and Grounding Guidelines, publication, <a href="#">1770-4.1</a>	Provides general guidelines for installing a Rockwell Automation industrial system.
<a href="#">Rockwell Automation product certifications</a>	Provides declarations of conformity, certificates, and other certification details.

View or download publications at <https://www.rockwellautomation.com/en-us/support/documentation/literature-library.html>. To order paper copies of technical documentation, contact a local Rockwell Automation distributor or sales representative.

# Legal notices

Rockwell Automation publishes legal notices, such as privacy policies, license agreements, trademark disclosures, and other terms and conditions on the [Legal Notices](#) page of the Rockwell Automation website.

## Software and Cloud Services Agreement

Review the Rockwell Automation Software and Cloud Services Agreement [here](#).

## Open Source Software Licenses

The software included in this product contains copyrighted software that is licensed under one or more open source licenses.

You can view a full list of all open source software used in this product and their corresponding licenses by opening the index.html file located your product's OPENSOURCE folder on your hard drive.

The default location of this file is:

C:\Program Files (x86)\Rockwell Software\Studio 5000\Logix Designer SDK\ReleaseNotes\OPENSOURCE\index.htm

You may obtain Corresponding Source code for open source packages included in this product from their respective project web site(s). Alternatively, you may obtain complete Corresponding Source code by contacting Rockwell Automation via the **Contact** form on the Rockwell Automation website: <http://www.rockwellautomation.com/global/about-us/contact/contact.page>. Please include "Open Source" as part of the request text.

## Overview

The Logix Designer Software Developer's Kit (SDK) provides programmatic access for 3rd party applications to manage and interact with the Logix Designer application. Use the `IServiceApiClientV2` and `IControllerApiClient` interfaces to set up the SDK and manage Logix Designer programs with programmable tools and functionality. For a complete description of the SDK functions and requirements, see the Logix Designer SDK online help.

## Considerations when using the Logix Designer SDK

Keep these considerations in mind when using the Logix Designer SDK.

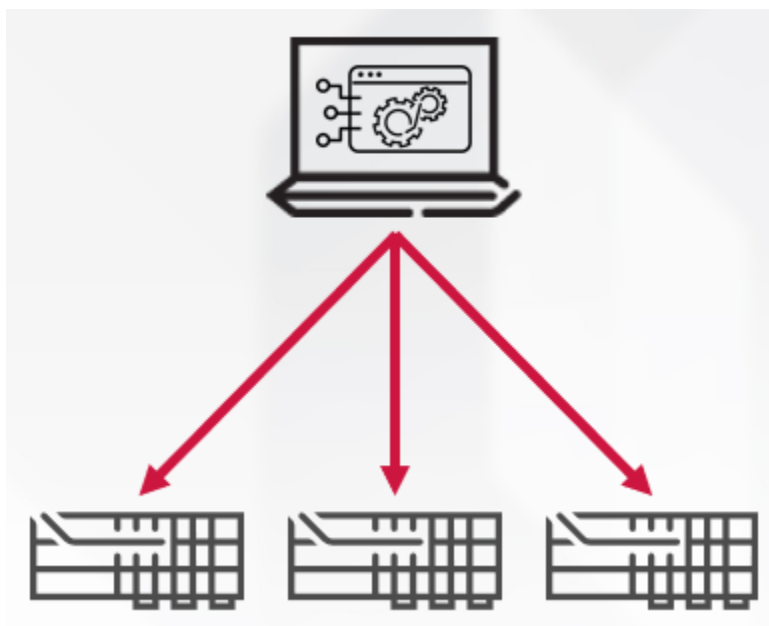
- The SDK supports projects for Logix Designer version 31 and later. Projects for versions 30 and earlier are not supported.
- The SDK supports FactoryTalk Linx as its communication software. FactoryTalk RSLinx is not supported.
- The controller that you configure to be used by the communication path must be previously configured or visible over the FactoryTalk Linx Network Browser.
- The Logix Designer SDK supports only a 64-bit OS.
- The maximum size for Logix Designer projects is 500MB.
- The maximum size for a single Logix Designer SDK operation data file is 30kB.
- The Logix Designer SDK server uses this specific TCP port: 53204. It is not configurable. Make sure this port is free.

## Download projects to controllers

Use the Logix SDK to build programs that download projects to controllers.

### Download one project to multiple controllers

Use the Logix Designer SDK to create a program that downloads one project to multiple controllers. For example, if you have several identical machines set up and you need one project to run on all of them, you can create a program that completes the download steps.



The download program carries out these steps:

1. Opens the ACD project in Logix Services.
2. Sets the communication path to the controller.
3. Switches the controller to Remote Program mode.
4. Downloads the project to the controller.
5. Without closing the project, repeats steps 2, 3, and 4 for each controller.

### Example: Download one project to multiple controllers

This example code shows how to construct an application that downloads one project to multiple controllers.

```
using RockwellAutomation.LogixDesigner;  
  
namespace SingleAcidManyControllerDownload  
{  
    /// <summary>  
    /// This class shows how to create a static class which could be referenced by a program.  
}
```



```

///
/// This class associates multiple controller comm paths with a single ACD file,
/// then exposes a method which will open the projects and initiate downloads in series.
///
/// It then saves those projects into a dictionary, using the comm path as the key, and a handle
on the <see cref="LogixProject"/>
///
///
/// If you instead wanted to download in parallel, see the ManyAcdManyControllerDownload example.
///
/// If you want to see how to reference a class in your main program,
/// see the examples CreateDeploymentSdCard or ProvisionAndValidate.
/// </summary>
public class DownloadCtrl
{
    /// <summary>
    /// This is a hard coded list of controller comm paths.
    /// This information could come from anywhere. You could expand this hard-coded list,
    /// or load the information from an XML or CSV file if you wish, or passed as variables from
a CI system.
    /// </summary>
    private const string CTRL1 = @"AB_ETH-1\10.116.38.143\Backplane\8";
    private const string CTRL2 = @"AB_ETH-1\10.116.38.144\Backplane\8";
    private const string CTRL3 = @"AB_ETH-1\10.116.38.145\Backplane\8";
    private const string CTRL4 = @"AB_ETH-1\10.116.38.146\Backplane\8";
    private const string CTRL5 = @"AB_ETH-1\10.116.38.147\Backplane\8";

    /// <summary>
    /// The file path to the ACD.
    /// </summary>
    private static string _filePath = @"C:\Path\To\Project.acd";

    /// <summary>
    /// The controller comm path list.
    /// </summary>
    private static List<string> _controllerCommPaths = new List<string>()
    {
        CTRL1,
        CTRL2,
        CTRL3,
        CTRL4,
        CTRL5,
    };

    /// <summary>

```

```

    /// This dictionary will be populated so that multiple instances of <see
    cref="LogixProject"/>

    /// can be identified and retrieved by their comm path for use elsewhere in a program.

    /// </summary>

    private static Dictionary<string, LogixProject> logixSdkProjects = new Dictionary<string,
    LogixProject>();

    public static async Task DownloadToMultipleControllers()
    {
        var programMode = LogixProject.RequestedControllerMode.Program;
        var runMode = LogixProject.RequestedControllerMode.Run;

        /*! Download ACD file to all controllers and set to run mode. !*/
        foreach (var controller in _controllerCommPaths)
        {
            LogixProject logixProject = await LogixProject.OpenLogixProjectAsync(_filePath);
            await logixProject.SetCommunicationsPathAsync(controller);
            await logixProject.ChangeControllerModeAsync(programMode);
            await logixProject.DownloadAsync();
            await logixProject.ChangeControllerModeAsync(runMode);

            /*! hold project reference for later use !*/
            logixSdkProjects.Add(controller, logixProject);
        }

        // you could read from the controller later.
        var tagPath = "Controller/Tags/Tag[@Name='someBool']";
        var ctrl1Project = logixSdkProjects.Single(x => x.Key == CTRL1).Value; /*! A reference
        to the LogixProject that matches the same commPath as CTRL1 !*/

        //if SOME CONDITION
        if(await ctrl1Project.GetTagValueBOOLAsync(tagPath,
        LogixProject.TagOperationMode.Online))
        {
            //Do something
        }
    }
}

```

## Download multiple projects to multiple controllers

Use the Logix Designer SDK to create a program that downloads multiple projects to multiple controllers. In this use case, each controller has its own ACD project file to be downloaded.

The download program runs this sequence as parallel threads for many controllers:

1. Open the ACD project.
2. Set the comm path to the controller.
3. Set the controller to Remote Program mode.
4. Download the project.

## Example: Multiple ACDs to multiple controllers

This example code shows how to construct an application that downloads multiple projects to multiple controllers.

```
using RockwellAutomation.LogixDesigner;

namespace ManyAcidManyControllerDownload
{
    /// <summary>
    /// This class shows how to create a static class which could be referenced by a program.
    ///
    /// This class associates multiple controller comm paths with their specified ACD file,
    /// then exposes a method which will open the projects and initiate downloads in parallel.
    ///
    /// If you instead wanted to download in series, see the SingleAcidManyControllerDownload example.
    ///
    /// If you want to see how to reference a class in your main program,
    /// see the examples CreateDeploymentSdCard or ProvisionAndValidate.
    /// </summary>
    public class DownloadManyAcids
    {
        /// <summary>
        /// This is a hard coded list of controller comm paths and file-paths to ACD files.
        /// This information could come from anywhere. You could expand this hard-coded list,
        /// or load the information from an XML or CSV file if you wish, or passed as variables from
        a CI system.
        /// </summary>
        private const string CTRL1 = @"AB_ETH-1\10.116.38.143\Backplane\8";
        private const string CTRL2 = @"AB_ETH-1\10.116.38.144\Backplane\8";
        private const string CTRL3 = @"AB_ETH-1\10.116.38.145\Backplane\8";
        private const string CTRL4 = @"AB_ETH-1\10.116.38.146\Backplane\8";
        private const string CTRL5 = @"AB_ETH-1\10.116.38.147\Backplane\8";
        private const string FILE1 = @"C:\Path\To\Project.acd";
        private const string FILE2 = @"C:\Path\To\Project2.acd";
        private const string FILE3 = @"C:\Path\To\Project3.acd";
        private const string FILE4 = @"C:\Path\To\Project4.acd";
        private const string FILE5 = @"C:\Path\To\Project5.acd";
    }
}
```

```

    /// <summary>
    /// This dictionary associates a comm path with an ACD file.
    /// This assigns uniquely, but multiple controllers could have the same ACD file if you
chose.
    /// </summary>
    private static Dictionary<string, string> _controllers = new Dictionary<string, string>()
    {
        { CTRL1, FILE1 },
        { CTRL2, FILE2 },
        { CTRL3, FILE3 },
        { CTRL4, FILE4 },
        { CTRL5, FILE5 },
    };

    /// <summary>
    /// This function loops over each item in <see cref="_controllers"/> above queues all the
necessary steps to download to a controller
    /// for each item.
    /// </summary>
    /// <returns>A task which resolves when all downloads are complete.</returns>
    public static Task Download()
    {
        var taskList = new List<Task>();
        foreach( var (commPath, acdFile) in _controllers){
            /*! Each comm path/acdFile pair has a task to run the following lambda function added
to the list of tasks created above. !*/
            taskList.Add(Task.Run(async ()=>
            {
                /*! Open project, set comm path, change mode to program, download, change mode to
run. !*/
                var logixProject = await LogixProject.OpenLogixProjectAsync(acdFile);
                await logixProject.SetCommunicationsPathAsync(commPath);
                await
logixProject.ChangeControllerModeAsync(LogixProject.RequestedControllerMode.Program);
                await logixProject.DownloadAsync();
                await
logixProject.ChangeControllerModeAsync(LogixProject.RequestedControllerMode.Run);
                Console.WriteLine($"Finished downloading {acdFile} to {commPath}");
            }));
        }
        return Task.WhenAll(taskList); /*! Return when all threads are complete !*/
    }

```

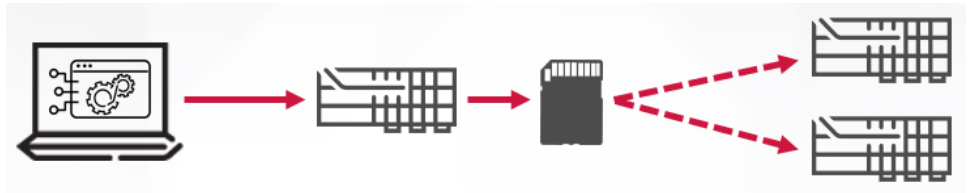
```

}
}

```

## Deploy a project and save it to an SD Card

Use the Logix Designer SDK to create a program that downloads a project to a controller and then saves the project to an SD card. Users can carry the SD card to other controllers and run the program on startup.



The program carries out these steps:

1. Opens the ACD project in Logix Services.
2. Sets the communication path to the controller.
3. Switches the controller to Remote Program mode.
4. Downloads the project to the controller.
5. Saves the project to the SD card.

## Example: Download a project and save to an SD card

This example code shows how to construct an application that downloads a project to a controller and saves the project to an SD card.

```

using RockwellAutomation.LogixDesigner;

namespace CreateDeploymentSdCard
{
    /// >summary>
    /// Class which opens a >see cref="LogixProject"/>, goes online with a given controller,
    downloads the program,
    /// and then exposes the >see cref="LogixProject.StoreImageOnSDCardAsync"/> to the user. Uses the
    initializer pattern
    /// because the initial steps are asynchronous.
    /// >/summary>
    public class SDCardFactory
    {
        /// >summary>
        /// Reference to the >see cref="LogixProject"/> to be referenced later.
        /// >/summary>
        private LogixProject _project;
    }
}

```

```

    /// >summary>
    /// Instantiates the SDCardFactory class.
    /// >/summary>

    /// >param name="project">The >see cref="LogixProject"/> to load to SD card.>/param>
    private SDCardFactory(LogixProject project)
    {
        _project = project;
    }

    /// >summary>
    /// A pass through method to >see cref="LogixProject.StoreImageOnSDCardAsync"/>
    /// >/summary>
    /// >returns>A task which resolves when the SD card write is complete.>/returns>
    public async Task LoadToSdCard()
    {
        await _project.StoreImageOnSDCardAsync();
    }

    /// >summary>
    /// Static initializer which opens the project and does all the one time
    /// setup tasks, then returns a new instance of >see cref="SDCardFactory"/>.
    /// >/summary>
    /// >param name="acdPath">The path to the ACD file.>/param>
    /// >param name="commPath">The comm path to the controller.>/param>
    /// >returns>An instance of >see cref="SDCardFactory"/>>/returns>
    public static async Task<SDCardFactory> Init(string acdPath, string commPath)
    {
        var project = await LogixProject.OpenLogixProjectAsync(acdPath); /*! Open the project !*/
        await project.SetCommunicationsPathAsync(commPath); /*! Set the comm path in the project
to the one given by the user. !*/
        await
project.ChangeControllerModeAsync(LogixProject.RequestedControllerMode.Program); /*! Set controller
to program mode !*/
        await project.DownloadAsync(); /*! Download the program. !*/

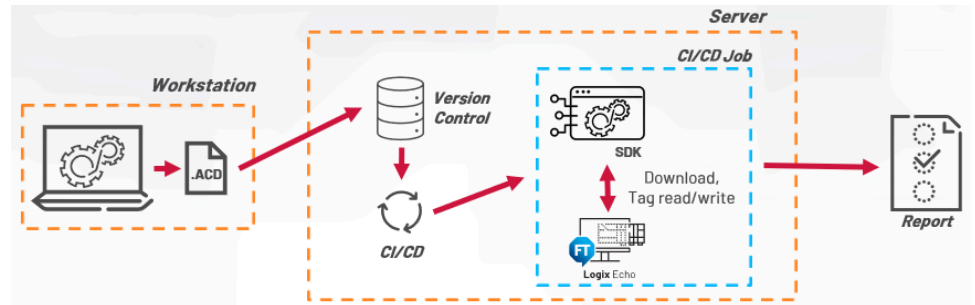
        return new SDCardFactory(project);
    }
}
}

```

## Download a project for automated testing

Use the Logix Designer SDK to create a program that downloads a project and runs it on an emulated controller.

- Commit an ACD file to version control.
- A CI/CD system, such as Jenkins, invokes a script that uses the Logix Designer SDK to download and run the project on an emulated controller.
- The script writes and reads tag values to verify expected ACD file behavior, providing automated feedback as part of a CI/CD tool chain.



## Example: Provision and validate

This example code shows how to construct an application that downloads a project and validates it by running it on an emulated or physical controller.

```
using RockwellAutomation.LogixDesigner;

namespace ProvisionAndValidate
{
    /// <summary>
    /// Example class to show how you might deploy a project to a controller and then validate the
    /// information.
    /// Uses the initializer pattern
    /// because the initial steps are asynchronous.
    /// </summary>
    public class ProvisionAndValidate
    {
        /// <summary>
        /// Reference to the <see cref="LogixProject"/>.
        /// </summary>
        private LogixProject _logixProject;

        private LogixProject.TagOperationMode tagModeOnline = LogixProject.TagOperationMode.Online;

        /// <summary>
        /// Instantiates an instance of the <see cref="ProvisionAndValidate"/> class.
        /// </summary>
        /// <param name="project">The <see cref="LogixProject"/>.</param>
        private ProvisionAndValidate(LogixProject project)
        {
```

```

        _logixProject = project;
    }

    /// <summary>
    /// Method to validate a given string matches a string value in a controller.
    /// </summary>
    /// <param name="tagPath">The full path to the tag.</param>
    /// <param name="expectedValue">The value we should expect to see.</param>
    /// <returns>Boolean true if the values match.</returns>
    /// <exception cref="ValueMismatchException">An exception if the values do not match the
    expected value.</exception>
    public async Task<bool> StringValueMatches(string tagPath, string expectedValue)
    {
        try
        {
            var actualValue = await _logixProject.GetTagValueSTRINGAsync(tagPath, tagModeOnline);
            bool valuesMatch = expectedValue == actualValue;
            if (!valuesMatch)
            {
                throw new ValueMismatchException($"Value ${tagPath} did not match. Actual:
                ${actualValue} Expected: ${expectedValue}");
            }
            return valuesMatch;
        }
        catch (LogixSdkException)
        {
            Console.WriteLine($"Something went wrong trying to get tag value ${tagPath}.");
            throw;
        }
    }

    /// <summary>
    /// Method to validate a given Dint value in a controller.
    /// </summary>
    /// <param name="tagPath">The full path to the tag.</param>
    /// <param name="expectedValue">The value we should expect to see.</param>
    /// <returns></returns>
    /// <exception cref="ValueMismatchException">An exception if the values do not match the
    expected value.</exception>
    public async Task<bool> DINTValueMatches(string tagPath, int expectedValue)
    {
        try
        {
            var actualValue = await _logixProject.GetTagValueDINTAsync(tagPath,
tagModeOnline); /*! Get the value !*/

```



```

        bool valuesMatch = expectedValue == actualValue; /*! Compare !*/

        if (!valuesMatch)
        {
            throw new ValueMismatchException($"Value ${tagPath} did not match. Actual:
${actualValue} Expected: ${expectedValue}");
        }

        return valuesMatch;
    }

    catch (LogixSdkException)
    {
        /*! If something went wrong getting the tag value, output an error message and
re-throw */

        Console.WriteLine($"Something went wrong trying to get tag value ${tagPath}.");
        throw;
    }
}

/// <summary>
/// Creates an instance of <see cref="LogixProject"/> and stores it for use later.
/// </summary>
/// <param name="acdPath">The ACD path.</param>
/// <param name="commPath">The comm path to the controller.</param>
/// <returns>An instance of the ProvisionAndValidate class.</returns>
public static async Task<ProvisionAndValidate> Init(string acdPath, string commPath)
{
    var proj = await LogixProject.OpenLogixProjectAsync(acdPath);
    await proj.SetCommunicationsPathAsync(commPath);
    await proj.ChangeControllerModeAsync(LogixProject.RequestedControllerMode.Program);
    await proj.DownloadAsync();
    await proj.ChangeControllerModeAsync(LogixProject.RequestedControllerMode.Run);

    return new ProvisionAndValidate(proj);
}

/// <summary>
/// Custom exception class for values that mismatch.
/// </summary>
public class ValueMismatchException : Exception
{
    public ValueMismatchException(string message)
        : base(message)
    {
    }
}

```

```
}  
}
```

# Rockwell Automation Support

Use these resources to access support information.

Technical Support Center	Find help with how-to videos, FAQs, chat, user forums, and product notification updates.	<a href="http://rok.auto/support">rok.auto/support</a>
Knowledgebase	Access Knowledgebase articles.	<a href="http://rok.auto/knowledgebase">rok.auto/knowledgebase</a>
Local Technical Support Phone Numbers	Locate the telephone number for your country.	<a href="http://rok.auto/phonesupport">rok.auto/phonesupport</a>
Literature Library	Find installation instructions, manuals, brochures, and technical data publications.	<a href="http://rok.auto/literature">rok.auto/literature</a>
Product Compatibility and Download Center (PCDC)	Get help determining how products interact, check features and capabilities, and find associated firmware.	<a href="http://rok.auto/pcdc">rok.auto/pcdc</a>

## Documentation feedback

Your comments help us serve your documentation needs better. If you have any suggestions on how to improve our content, complete the form at [rok.auto/docfeedback](http://rok.auto/docfeedback).





## Waste Electrical and Electronic Equipment (WEEE)



At the end of life, this equipment should be collected separately from any unsorted municipal waste.

Rockwell Automation maintains current product environmental information on its website at [rok.auto/pec](http://rok.auto/pec).

Rockwell Otomasyon Ticaret A.Ş. Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400 EEE Yönetmeliğine Uygundur

Connect with us.    

**rockwellautomation.com** — expanding **human possibility**<sup>™</sup>

AMERICAS: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444

EUROPE/MIDDLE EAST/AFRICA: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640

ASIA PACIFIC: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846