High Tech Industry Forum

C# and Python Connectivity to Logix & Micro800 Controllers
Integrated Architecture
Infrastructure for the Connected Enterprise

- Multi Discipline Control
  - Architecture simplification
  - Lower maintenance

- Scalable
  - Control architectures for small to large systems
  - Common design environment

- EtherNet/IP
  - Standard, unmodified Ethernet
  - “IT ready”

- Real-time Information
  - Turn data into actionable information
  - Starts where the information is born
Integrated Architecture Enablers

Smart
- FactoryTalk Cloud
- Energy Saving
- Premier Integration
- Self aware, system aware
- Multi-discipline platform

Productive
- FactoryTalk TeamONE
- FactoryTalk Analytics
- Open Interfaces
- Library Management
- Integrated Development Environment

Secure
- Tamper Detection
- Secure Network Infrastructure
- Content Protection
- Policy Management
- Authentication
This is the Integrated Architecture
Multiple Automation Cells Example
High Tech Portfolio Supporting IA

- SAFETY
- IPC
- OPEN CONTROLLER
- ANALYTICS
- MOTION
- MAGNEMOTION
- VISION
- ROBOTICS
- PROCESS
- SIMULATION
Scalable Open Controller Solutions

RA Hybrid Custom + Off-The-Shelf Solutions Reduce Time to Market & Engineering Costs
PC-Based Control Options

Meet customer OS specification with various connectivity options

PC-BASED CONTROL

C# .NET

PYTHON

iOS

WINDOW 10

LINUX
Getting information out of Machines & Equipment

Real-time Data
Voltage, Kwh, Running Time, Temperature

Information
CONTEXTUALIZATION
Energy/Product, OEE

Knowledge
ANALYTICS
Predict bearing will fail in 10 hours

Wisdom
OPTIMIZE
More efficient process workflows
Python & C# Connectivity Options

- CompactLogix 5480 & IXR DLL Calls
- Logix In-Rack C Module Solutions
- OPC C# Interface Solutions (FT Gateway & RSLinx Classic)
- C# .NET Driver Solutions
- Linux Python Socket Solutions
Introducing CompactLogix 5480

**Multi-disciple high performance controller**
- Multi-discipline Logix controller with Microsoft Windows IoT Enterprise for 3rd party applications
- Commercially available CPU for high performance

**Enhanced Security Features**
- Digitally-signed and encrypted firmware
- Controller change detection
- Role-based access control
- License-based content protection

**Secure Digital Card**
- Optional nonvolatile memory storage for the Logix user program and tag data

**USB Ports**
- USB device port for easy access for programming, troubleshooting, online edits and updating the controller firmware
- (2) USB 3.0 ports (OS)

**Gb EtherNet/IP Ports**
- (3) Logix GbE ports. 2 ports configurable for Dual IP or DLR
- (1) Operating System GbE port

**Monitor Interface**
- DisplayPort supports standard converters for HDMI, DVI, VGA displays
CompactLogix 5480
Product Overview

- USB device port
- Upstream CIP GbE port
- CIP GbE Configurable DLR
  A1/A2
- OS dedicated GbE port
  USB 3.0 (2)
  External display port
- Status LEDs and display
- DIN rail mount
- 5069 local IO
  (up to 31 modules)
- Redundant fans that can be replaced during runtime
- No Haz Loc or Marine

Copyright © 2016 Rockwell Automation, Inc. All Rights Reserved.
## CompactLogix 5480

### Ethernet Interfaces

- **B1**
  - GbE (EtherNet/IP)
  - Support for I/O, Motion, Drives etc.
  - Allows Domain Name, Primary and Secondary DNS Server Address attributes ("Uplink" use case)

- **A1**
  - GbE (EtherNet/IP)
  - Support for I/O, Motion, Drives etc.
  - Individual IP or Linear/DLR (w/A2)

- **A2**
  - GbE (EtherNet/IP)
  - Support for I/O, Motion, Drives etc.
  - Individual IP or Linear/DLR (w/A1)

- **X1**
  - GbE
  - Dedicated Windows 10 IoT Interface
  - No Bridging of I/O
### CMX 5480 Specifications

<table>
<thead>
<tr>
<th>Feature/Capability</th>
<th>CMX 5480</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Engine</td>
<td>Real time Logix</td>
</tr>
<tr>
<td>Operating System</td>
<td>Win 10 IoT Enterprise</td>
</tr>
<tr>
<td>Integrated HMI</td>
<td>Yes</td>
</tr>
<tr>
<td>High Level Languages</td>
<td>Future (R2)</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel i7 2.4GHz Quad</td>
</tr>
<tr>
<td>Storage</td>
<td>32GB* (eMMC not expandable)</td>
</tr>
<tr>
<td>SDRAM</td>
<td>5.75GB (Windows)</td>
</tr>
<tr>
<td>GbE Ports</td>
<td>3 (Logix) 1 (OS)</td>
</tr>
<tr>
<td>Monitor Interface</td>
<td>1 (DisplayPort)</td>
</tr>
<tr>
<td>USB Ports</td>
<td>1 (Logix) 2 (USB 3.0)</td>
</tr>
<tr>
<td>Operating Temp</td>
<td>0-60°C</td>
</tr>
</tbody>
</table>

* 16-18GB Free Space
## CompactLogix 5480
### Windows Release Plan

<table>
<thead>
<tr>
<th>Feature</th>
<th>Release 1 Q2CY17</th>
<th>Release 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>V30 (AFD 12/2016) support</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>V31 (AFD 10/2017) support</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Configurable Device Level Ring or dual IP</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Windows 10 IoT Enterprise</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Internal CIP communications (Logix ↔ Windows)</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>External Routine Calls (IXR &amp; 3rd party program support)</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Windows Events support</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Windows (WUC) NVS Store support</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>OPC-UA data access support</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FactoryTalk View Software compatibility</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PanelView 5000 Terminal support</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Use Cases

R2 Release
CompactLogix 5480 Use Cases
Additional functionality available in R2 Release

Software-based: Run Windows-based software and exchange data with Logix

- Implement a custom application developed in high level language
- High speed data transfer between 3rd party applications and Logix

Application-based: Run a 3rd party application e.g. Vision system and trigger an event in Logix

- Execute complex math calculations and algorithms for iTrak motion and re-pitching motion cams
- Advanced camming (process-intensive cam profile calculations)
- Emulate a 3rd party control engine (e.g. Robot control) in Windows
- Develop complex algorithm (power grid, wind turbine, adaptive control, multivariable control, time discrete control, etc.) for custom application needs
CompactLogix 5480 Use Cases

Additional functionality available in R2 Release

**Information-enabled: pass controller data to the Cloud via Windows**

- OPC-UA data access to 3rd party devices
- Enable OPC-UA to exchange data to the Cloud
- High speed data collection

**Automation-enabled: Optimization of Automation architecture**

- Perform complex calculations (e.g. path planning, adaptive control calculations for modeling) that are typically done on a computer
- External high level language routine calls (C, C++, C#, Visual Basic, Java, JavaScript, etc.)
Deep Dive – IXR
Interface to external and embedded user code

- Support of multiple high-level programming languages interfacing with a Dynamically Linked Library (DLL) within the Windows 10 IoT Enterprise execution space.

- CompactLogix 5480 real-time controller shall invoke an instruction to call an exported DLL function developed by the end user using standard Microsoft development tools. No special additions to the development tools shall be required to develop the DLL.

- DLL exported functions have the ability to interact with any Microsoft based programming language such as C, C++, C#, Visual Basic, Java, JavaScript and more.

- Adding an IXR to a user program shall require some configuration much like a message instruction does today.
IXR – Invoke External Routine

What is it?

• New instruction that is native to the Logix Designer development environment.
• Available in Ladder, Structured Text etc.

How do you implement it?

• Add the IXR instruction and point it to a DLL that is built using standard Windows-based development tools
• Any functions within the DLL will be presented to the user through Logix Designer. User selects the function they want to call, then matches parameters that will get passed into it.
• Download ACD file to the controller
  • DLL stays within the context of Logix Designer ACD file
  • Controller keeps local DLL copy and copies it into Windows
IXR – Things to Consider

- Data types supported mimic those supported in Logix Designer today (UDT, DINT, etc)
  - No Microsoft-based string data etc.
- Up to 64 unique instances of an IXR call per controller
- Pass up to 200,000 bytes of data at a time
- Function call execution is asynchronous to the program scan – so Logix continues to run (non-blocking call)
- When user clears memory or deletes the program
  - Windows will unload the DLL and delete it
- When they do a NVS store – it will save the DLL to the SD card
- Service is designed so that it can’t be stopped
- Any changes made to a DLL will require a download to the controller
- It can be embedded in an Add-On-Instruction (AOI).
- Works like a MSG instruction. The external routine will run, then when done, will return status to the controller (data, done or error, and then what error is)
Python & C# Connectivity Options

- CompactLogix 5480 & IXR DLL Calls
- Logix In-Rack C Module Solutions
- OPC C# Interface Solutions (FT Gateway & RSLinx Classic)
- C# .NET Driver Solutions
- Linux Python Socket Solutions
Logix In-Rack C Module Solutions
Python & C# Connectivity Options

- CompactLogix 5480 & IXR DLL Calls
- Logix In-Rack C Module Solutions
- **OPC C# Interface Solutions (FT Gateway & RSLinx Classic)**
- C# .NET Driver Solutions
- Linux Python Socket Solutions
OPC C# Interface Solutions

OPC Foundation

FactoryTalk

kepware
FactoryTalk Gateway
Python & C# Connectivity Options

- CompactLogix 5480 & IXR DLL Calls
- Logix In-Rack C Module Solutions
- OPC C# Interface Solutions (FT Gateway & RSLinx Classic)
- C# .NET Driver Solutions
- Linux Python Socket Solutions
C# .NET Driver Solutions
ASComm.NET C# .NET Driver

- Automated Solutions, Inc. established 1994
- Develops and sells software products for industrial and process automation
- Currently offer 4 Allen-Bradley connectivity products (drivers) for ControlLogix, CompactLogix, GuardLogix, Micro800, PLC5, SLC500, & MicroLogix

ASComm.NET Overview

- Fully managed .NET component allows Visual Studio .NET developers to read & write tags on A-B ControlLogix family, Micro800, PLC5, SLC, MicroLogix
- Does not require RSLinx or OPC
- Runtime-free for non-commercial applications (end-user, integrator custom, etc.)
  - Several licensing options including Machine, Single Developer, and Discounted Team
- Commercial Software Vendor program for commercial applications (COTS).
  - Runtime fees are fraction of the cost of OPC.
  - No separate application to configure.
- Use to build Windows, WPF, Web, Service, and Console applications targeting x64, x86 or Any CPU.
- Can build Windows Forms and console applications that run on Linux with Mono framework.
ASComm.NET C# .NET Driver Demo

ASComm.NET Usage

- C# example code to read and write three tags

```csharp
// Configuration at application startup
AB.Net.Channel myChannel = new AB.Net.Channel();
AB.Device myDevice = new AB.Device("192.168.0.55,1,0"); // IP Address 192.168.0.55, CPU in slot 0
AB.Group myGroup = new AB.Group(false, 100); // Group with 100 mSec update rate
AB.Item myItem1 = new AB.Item("myTag", 1); // Single atomic tag
AB.Item myItem2 = new AB.Item("myUDT", 1); // Single UDT tag
AB.Item myItem3 = new AB.Item("myIntArray[3]", 10); // INT array tag, 10 elements starting
myChannel.Devices.Add(myDevice); // Add Device to Channel, no limit on devices
myDevice.Groups.Add(myGroup); // Add Group to Device, no limit on Groups
myGroup.Items.AddRange(new Item[] { myItem1, myItem2, myItem3 }); // Add Items to Group, no limit on Items

try
{
    myItem1.Write(1234); // Write value 1234 to myItem1
    myItem1.Read(); // Read value on-demand from myItem1
    textBoxItemRead.Text = myItem1.Values[0].ToString(); // save value to text box
}
catch (Exception ex)
{
    // Implement error handler
}
```
ASBrowse.NET Overview
- Complimentary product allows for browsing controller tag database
- Fully managed .NET class library that can easily be incorporated into your application

ASComm Excel Add-in Overview
- No programming required to populate Excel spreadsheets
- VBA commands available for programmatic read and write
- Excel add-in uses ASComm.NET for communications engine

- Website
  http://automatedsolutions.com

- E-Mail
  sales@automatedsolutions.com
  support@automatedsolutions.com

- Phone
  707-578-5882
Live Demo
Python & C# Connectivity Options

- CompactLogix 5480 & IXR DLL Calls
- Logix In-Rack C Module Solutions
- OPC C# Interface Solutions (FT Gateway & RSLinx Classic)
- C# .NET Driver Solutions
- Linux Python Socket Solutions
What is a Socket?

- A **socket** is one endpoint of a two-way communication link between two programs running on the **network**.

- Socket API is available for many language and platforms:
  - C, C++, C#, Perl, Python, Java, etc...

- Socket programs written in any language and running on any platform can communicate with each other.

- The socket interface lets you use a AB controller to communicate via an Ethernet/IP module with Ethernet devices, such as barcode scanners, RFID readers, or other standard Ethernet devices, that do not support the Ethernet/IP application protocol.
Socket service compatible hardware.

- 1756-EN2T, 1756-EN2F, 1756-EN2TR, 1756-EN3TR ControlLogix® Ethernet/IP communication modules, firmware revision 5.007 or later
- 1769-L24ER-QB1B, 1769-L24ER-QBFC1B, 1769-L27ERM-QBFC1B CompactLogix controllers, firmware revision 20.011 or later
- 1769-L16ER, 1769-L18ER, 1769-L18ERM CompactLogix controllers, firmware revision 20.011 or later
- 1769-L19ER-BB1B CompactLogix controllers, firmware revision 28.011 or later
- 5580 series of ControlLogix controllers (examples: 1756-L85E, 1756-L83E)
- 5380 series of CompactLogix controllers (examples: 5069-L340ERM, 5069-L320ER)

Socket Services in above Ethernet modules can be used with any Logix Controllers rev 15 or higher.
Types of socket

- UDP socket—Sends and receives UDP datagrams.
- TCP client socket—The Logix5000 program initiates the connection.
- TCP server socket—Another device initiates the connection to the Logix5000 program.
- TCP listen socket—Listens on a specified port number for incoming connections.

Python 3.4 Socket TCP Demo

- Hardware used
  - Raspberry Pi 3
  - Rockwell Automation, Micro850

- Software used
  - Python 3.4
  - PyCharm Communicty 2016.3.2(64)
  - Rockwell Automation, Connected Components Workbench
Python 3.4 Socket TCP Demo

- Hardware used
  - Raspberry Pi 3
  - Rockwell Automation, Micro850

- Software used
  - Python 3.4
  - Rockwell Automation, Connected Components Workbench [sample code]
```python
import socket
import time

# Demo Micro800 IP address
host = '192.168.1.10'
# Demo Micro800 Socket server port number
port = 10001

# Create a socket object
micro800_socket = socket.socket()
# Make a socket connection
micro800_socket.connect((host, port))

def SocketTxRx(self):
    micro800_socket.send(self.msg.encode())
    data = micro800_socket.recv(82).decode()
    self.received.delete(0.0, END)
    self.received.insert(0.0, data)
    time.sleep(.05)
```
Micro850 Socket TCP demo program
### Summary

<table>
<thead>
<tr>
<th>Solution</th>
<th>Time to deploy</th>
<th>Effort required</th>
<th>Performance</th>
<th>Scalability</th>
<th>Good for?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rockwell Automation OPC</td>
<td>Low</td>
<td>Low</td>
<td>Best</td>
<td>Best</td>
<td>Plants, Large/Medium size projects</td>
</tr>
<tr>
<td>3&lt;sup&gt;rd&lt;/sup&gt; Party C#/OPC</td>
<td>Medium</td>
<td>Medium</td>
<td>Better</td>
<td>Better</td>
<td>Medium to standalone projects</td>
</tr>
<tr>
<td>Socket</td>
<td>High</td>
<td>High</td>
<td>Good</td>
<td>Low</td>
<td>Standalone projects</td>
</tr>
</tbody>
</table>

### Considerations
- ROI
- Time to market
- Support
Thank You!