Vi-CELL BLU Automation Integration Guide for Automation Solution Developers

Contents

Introduction

The Vi-CELL BLU analyzer can be attached to automation systems for situations where customers would like to be able to introduce samples without human intervention.
The Vi-CELL BLU when running in automation mode provides results consistent with stand-alone Vi-CELL BLU instruments with the convenience of online sampling for cell concentration and viability.

Refer to the Vi-CELL BLU and other device manuals for pertinent safety information.
Consult each applicable section in the user's manuals and heed all displayed and defined warnings and cautions.

Automation Configurations

The Vi-CELL BLU can be connected to any liquid pipettor which can accurately dispense 200 +/- 20 microliters to the Beckman Coulter Automation Sample Introduction Cup. The Sample Introduction Cup is supplied in Kit PN C57876 and installed by a Beckman Coulter service representative.

Additionally, the Vi-CELL BLU will accept 600µL deep 96 well plates (BEC PN C24841) from an automation robot that can reach into and place the plates onto the Instrument Plate Holder. The Instrument Plate Holder is supplied in Kit PN C57877 and installed by a Beckman Coulter service representative.

Sample information is passed to the Vi-CELL BLU as a work queue from the host system via OPC-UA. (See software automation procedures).

Physical requirements

Sample introduction Requirements

The liquid handler must be able to deliver 200+/- 20 µL to the bottom of the sample automation cup for accurate sample results. Any pipetting error greater than +/- 20 µL will result in system performance degradation and decreased count accuracy.

Bench Area Requirements

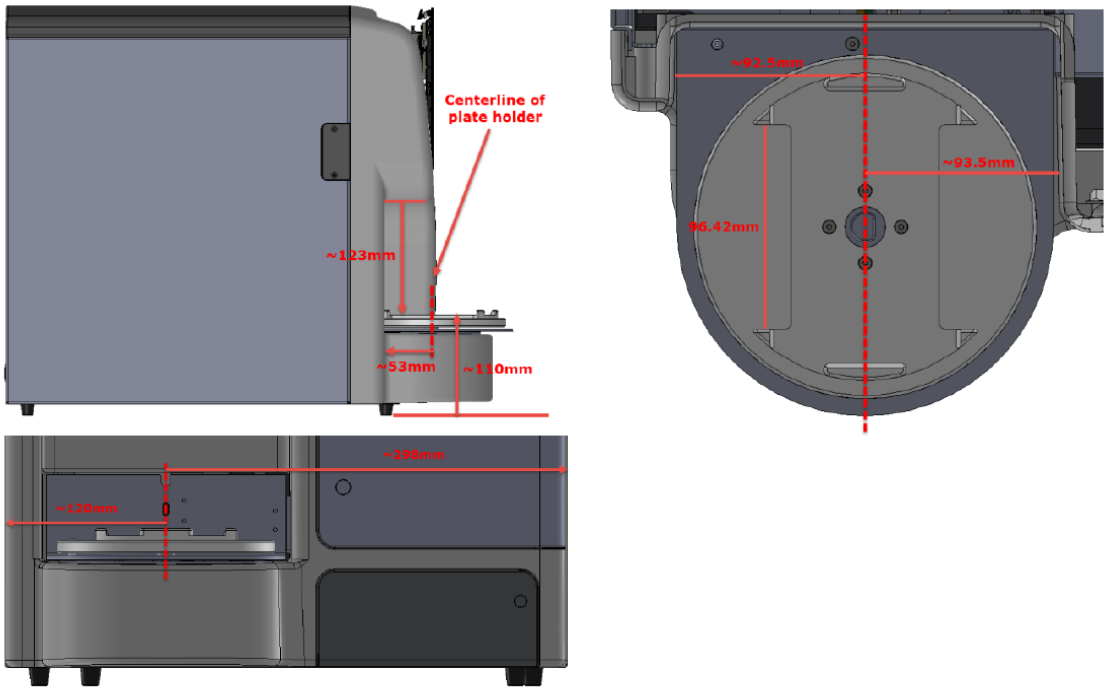
There must be adequate bench space to allow the Vi-CELL BLU instrument to sit close to the right or left of the liquid handler.

The instrument must be placed such that the user can access the touchscreen, replace the reagent pack and empty the waste bin. These can all be accessed from the front of the instrument.

The Automation sample cup can be located within 813mm (~32") from the left or right side of the Vi-CELL BLU instrument. The tubing length is precise and cannot be adjusted.

Additionally, the Automation Introduction cup may be located up to 850mm (~33.5”) above the base of the instrument.

For automation with a 96 well plate the instrument will present the deck (shown below) for access by a robotic gripper arm. Rough dimensions for clearances are shown below.



Electrical Requirements

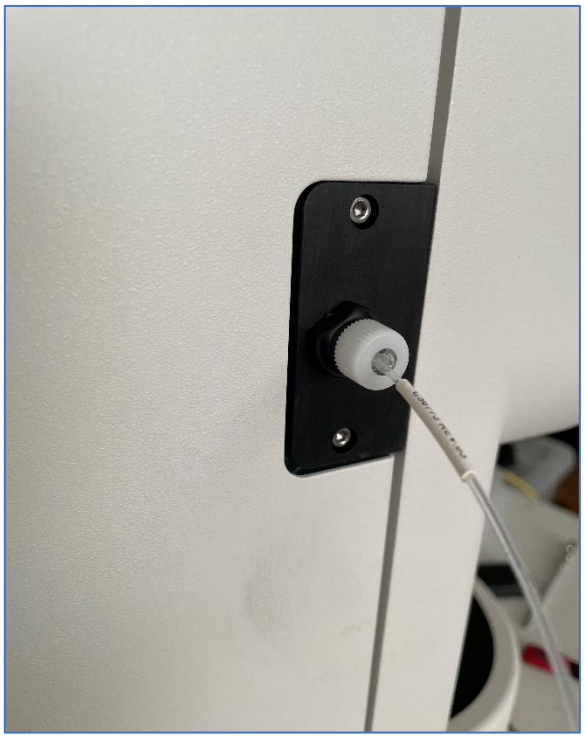
There are no impacts to the Electrical system circuit boards, devices, and cabling, due to the addition of the automation capability. Any changes to how the electro-mechanical system functions are handled in the software. The Vi-CELL BLU power requirements can be found in the Instructions for Use document.

Hardware Installation

Automation Kit Installation – The Beckman Coulter service representative will install the automation kit in either a left- or right-side configuration. The configuration includes a 1/4-28 threaded receptacle, external tubing, and a Sample Introduction Cup.

1. Right-Side tubing connection view

2. Left-Side tubing connection view



Automation Software Instructions for Vi-CELL BLU

This document will go over the process to use Vi-CELL BLU with external automation software. Vi-CELL BLU uses *Open Platform Communications Unified Architecture* (OPC UA) as the machine to machine communication implementation. If your automation software can function as an OPC UA Client, you can already connect to and communicate with the Vi-CELL BLU. If your automation software will not function as an OPC UA Client, you can use the Vi-CELL BLU .Net DLL to communicate with Vi-CELL BLU using .Net languages like C#, for example.

High Level Software Requirements for Automation

Automation system will be able to exercise the following with the reader:

- Request the health status
- Request operational metrics (e.g., amount of reagent left, tube tray count, available cell types)
- Request if the reader is idle and ready to be locked for automation
- Request an Automation Mode Lock on the reader to guarantee exclusive operational access
- Release the Automation Mode Lock
- Send sample configuration information to the reader
- Send basic operation commands like Start, Stop, Abort, Pause, etc.
- Request sample result(s) from the reader
- Read notification blocks provided by the reader (i.e. Notification Code, Notification Severity (Warning or Fatal), Notification Description); dispositioning these is at the discretion of the Automation System

Reader system (Vi-CELL BLU):

- Respond to the requests noted above
- Should look after its own error handling
- Should accept an ID for samples from the automation system
- Reader should provide the ability to download to the automation system the raw images used in the reading
- Reader should provide the ability to download unencrypted object data file (eBinary). The data will not be able to be imported back into the Reader. Exports can be resource intensive so unencrypted exports will be run when the Reader is not running samples.
- When the reader is “Locked” the operator cannot exercise any function that performs Create, Update, or Delete operations against Sample data or Configuration data. The operator will not be allowed to exercise resource-intensive operations (e.g., exporting). Some examples of operations that would be allowed are running reports, reviewing historical data, etc.

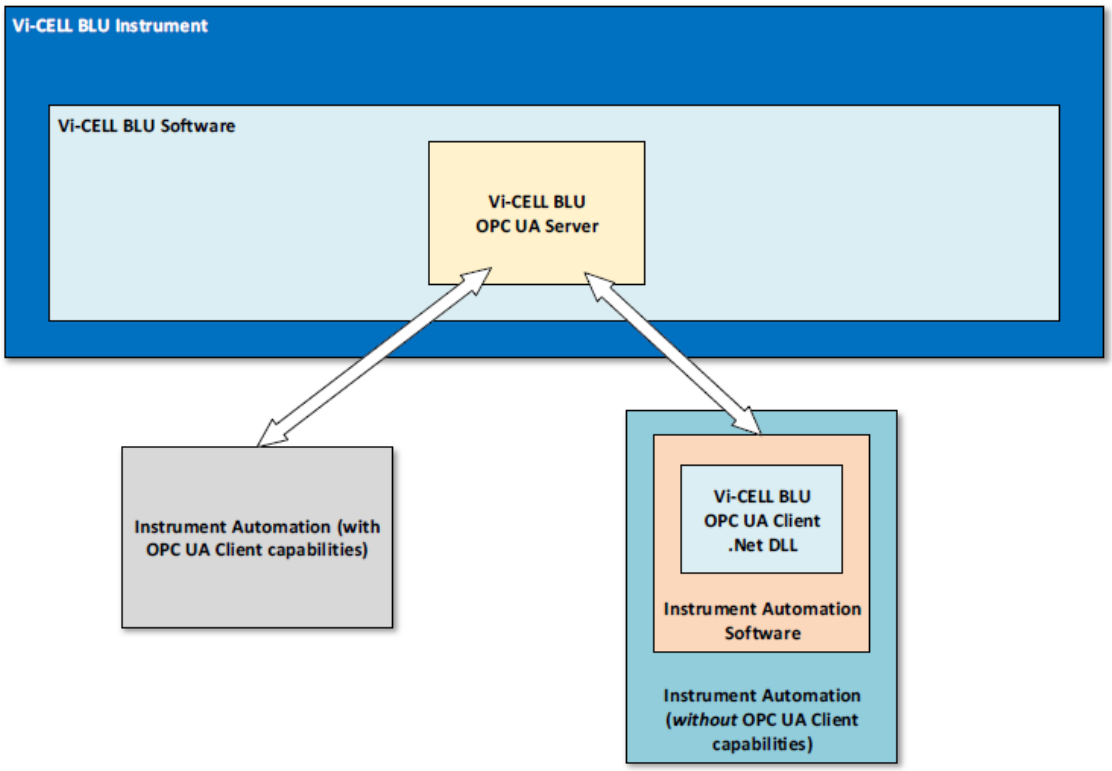
General:

- Interaction following a documented standard communication layer via Ethernet using OPC UA
- The OPC UA Server will be SOA oriented with a defined Service Layer, Operation Layer, and Data Layer
- Beckman Coulter will provide Technical Documentation and Sample Implementation (with source code) to Automation Software Engineers
- Beckman Coulter will provide a .Net client wrapper DLL if the Automation implementer desires to hide the OPC UA implementation from their code base
- When the Reader is locked the Splash Screen will show that the system is Locked for Automation; this visual should be obvious from a distance

System Diagram

There are 2 methods available for your automation software to communicate with Vi-CELL BLU:

1. Direct OPC UA communication using your OPC UA Client
2. Have your automation software reference the OPC UA Client .Net DLL and code your software to communicate using that library



API Functionality Overview

To aid in this document working for all integrators with and without OPC UA Client capabilities, we will assume:

- You already have a working understanding of OPC UA communication (if your automation software already contains OPC UA Client functionality).
- You already have the Vi-CELL BLU .Net DLL referenced in your automation software (if your automation software does not contain OPC UA Client functionality).

Restrictions

OPC UA requires an ethernet port to be open. This port is configurable via the Vi-CELL BLU software, but your company's IT department may need to intervene to allow network communication on this port.

You will need to have security enabled and use a username / password to connect to Vi-CELL BLU using OPC UA. There will be a default username/password for access in this case.

Vi-CELL BLU OPC UA Development Platform

Automation Instrument with OPC UA Client:

- OPC UA Clients can function on Microsoft Windows, Apple OSX, Android, or Linux
- OPC UA Clients can be written in a variety of programming languages including: C, C++, Java, .Net, JavaScript and Python

Automation Instrument with Vi-CELL BLU .Net Library:

- The Vi-CELL BLU .Net DLL was written in .Net Framework 4.8 and can be added as a reference to your automation software to directly access it if your software is also programmed in .Net
- The Vi-CELL BLU .Net DLL can function on Microsoft Windows™ 10 v1607 and newer

Programmer Skills Required

If you are using the Vi-CELL BLU .Net DLL, you do not need OPC UA programming skills. You must have to ability to refer to the DLL in your application and use its namespaces. You will also need to be able to call its methods and register .NET delegates for callback events.

If your automation software already has OPC UA Client capabilities, you will need to know how to connect to an OPC UA Server using user authentication and x509 certificates. You will also need to know how to communicate using OPC UA methods and handle its inputs/outputs.

Typical Workflow

1. Ensure the Vi-CELL BLU is powered on, running and connected to the network
2. Ensure your instrument automation is powered on, running and connected to the network
3. Connect your instrument automation to Vi-CELL BLU using user authentication by executing the Connect command
4. Request an automation lock from Vi-CELL BLU by executing the Automation Lock command with a Vi-CELL BLU defined username and password.
5. Using 96 well plate
    a. If using a 96 well plate, you can instruct Vi-CELL BLU to eject the stage for loading the plate. The stage will be closed/retracted when sample analysis begins.
    b. Send sample data information and start the analysis by executing the start sample set command.
6. If using external cup sampling, execute the start sample command with the details of the sample to be analyzed
7. If using a 96 well plate you can call Pause, Stop, Resume, or retrieve currently running sample information during analysis if you wish.
8. Wait for Vi-CELL BLU to send a sample complete event or all samples complete event (or poll ViCellStatus and wait for Idle to be returned).
9. Send the automation unlock command to release the lock on Vi-CELL BLU.
10. You can request the sample(s) analysis export by executing the retrieve sample export command.

Development Software Installation

Automation Kit Software Description

The automation kit option includes the offline SW version 1.4 USB (part number C40752) that can be provided to the Automation System Integrator along with this letter, which outlines where the SDK files are located.

Automation Kit Software Installation

Installing the offline version on any Windows 10 PC, will also install the OPC server so that the automation developer can test the functionality of their solution without requiring an instrument to be available. After installation, the SDK package files are located in C:\Instrument\Tools\SDK folder.

To install the offline software for use by the automation developer please run the install from a command prompt with the " -a" parameter. For example: "C:\C123456.exe -a". This will automatically enable the following for use by the automation developer:

1. Run the application in an application window instead of full screen.

2. Enable simulation mode so that simulated samples can be executed.

3. Turn on the OPC UA Server to support OPC UA Client debugging.

4. Enable the A-Cup simulator.

API Library

Vi-CELL BLU has several variables available for monitoring via OPC UA and several methods that can be called:

Properties

- ViCellIdentifier - string
- CurrentStatus ViCellStatus - Enum
- CurrentLockState - LockState - Enum
- ReagentUsesRemaining – uint32
- WasteTubeRemainingCapacityRemaining – uint32
- DiskSpaceAvailable uint32 – in MB
- CellTypes –List<CellType>
- QualityControls –List<QualityControl>
- CurrentRunningSampleName - string
- CurrentSamplePosition - SamplePosition

Methods:

- Connect – Establish a connection to a Vi-CELL BLU
    - Inputs
        - Username – string
        - Password – string
        - IpAddr – IPAddress
        - Port – uint32
        - discoverTimeout – uint32
        - cnxTimeout – uint32
    - Outputs – none
    - Return VcbResult – enum

- Disconnect
    - Inputs - none
    - Outputs – none
    - Return VcbResult - enum

- RequestLock – Request control of the Vi-CELL BLU
    - Inputs - none
    - Outputs ▪ LockState - the current state after request is completed
    - Return VcbResult - enum

- ReleaseLock
    - Inputs - none
    - Outputs - none
    - Return VcbResult – enum

- StartSample - Start for single sample (sample cup)
    - Inputs
        - SampleCfg
    - Outputs
        - none
    - Returns VcbResult
        - Callbacks called
        - On*SampleComplete* when the sample processing has completed, stopped, or cancelled
        - On*SampleSetComplete* when the sample processing has completed, stopped, or cancelled

- StartSampleSet - Start for multiple samples (96 well plate)
    - Inputs
        - SampleSetConfig
    - ▪ Outputs
        - none
    - Returns VcbResult
    - Callbacks called
        - On*SampleComplete* when the sample processing has completed, stopped, or cancelled
        - On*SampleSetComplete* when the sample processing has completed, stopped, or cancelled
- Pause
    - Inputs
        - None
    - Outputs
        - none
    - Return VcbResult
    - Callbacks called
        - OnPauseComplete once the instrument is paused
- Resume
    - Inputs
        - none
    - Outputs
        - none
    - Return VcbResult
    - Callbacks called
        - OnResumeComplete once the instrument has resumed
- Stop
    - Inputs

- none
- Outputs
  - none
- Return VcbResult
- Callbacks called
  - On StopComplete once the instrument has stopped

- EjectStage
  - Inputs
    - none
  - Outputs
    - none
  - Return VcbResult

- GetSampleResults

  - Inputs
    - Filter type - eFilterItem
    - From Date - DateTime
    - To Date - DateTime
    - Username - string (string.Empty for all users)
    - Search string for sample name or sample set name (depends on Filter Type) - string
    - Search string for sample tag - string
    - Cell type or quality control name - string
  - Outputs
    - List of SampleResult
  - Return VcbResult

- RetrieveSampleExport
  - Inputs
    - Samples – List<Uuid>
    - Filename – string- filename to save data to
  - Outputs
    - Callbacks called
      - OnSampleExportComplete – called when the sample export has completed
- DeleteSampleResults
  - Inputs
    - List of GUIDs (SampleRecordDomain uuids)
      - The automated instrument can make calls to GetSampleResults to get the uuids needed for the delete method
  - Outputs
    - none
  - Return VcbResult
  - Callbacks called
    - OnDeleteSampleResultsComplete once the instrument has completed the delete operation

- CreateCellType
  - Inputs

        - CellType
    - Outputs
        - none
    - Return VcbResult

- DeleteCellType
    - Inputs
        - CellTypeName - string
    - Outputs
        - none
    - Return VcbResult

- CreateQualityControl
    - Inputs
        - QualityControl
    - Outputs
        - none
    - Return VcbResult

- ImportConfig
    - Inputs
        - Filename – string – file to read and send to Vi-CELL BLU
    - Outputs
        - none
    - Return VcbResult

-
- ExportConfig
    - Inputs
        - Filename – string – filename to save config to
    - Outputs
        - none
- Return VcbResult

Callback functions

- OnSampleComplete
  - Triggered when a sample has completed analysis
  - Parameters
    - SampleResult – Result data for the sample that has completed (with UUID)

- OnSampleSetComplete
  - Triggered when all samples in work list have completed
  - Parameters
    - Status - bool

- OnPauseComplete
  - Triggered when the instrument has been paused
  - Parameters - none

- OnStopComplete
  - Triggered when the instrument has been stopped
  - Parameters – none

- OnResumeComplete
  - Triggered when the instrument has resumed
  - Parameters - none

- OnUpdateSystemStatus
  - Triggered when a system status update occurs
  - Parameters
    - Status - ViCellStatus

- OnUpdateLockState
  - Triggered when the Vi-CELL BLU lock state is updated.
  - Parameters:
    - LockState Enum

- OnDeleteSampleResultsComplete
  - Triggered when the DeleteSampleResults operation has completed.
  - Parameters:
    - Status – bool

Complex Types/Objects

- SampleConfig
  - Name – string
  - Position – SamplePosition – ignored for A Cup
  - Dilution – uint32
  - Tag – string
  - CellTypeName – string – used if set
  - QCName – string – used if CellTypeName is not set
  - SaveEveryNthImage – uint32
  - WashType – enum – ignored for A Cup

- SampleSetConfig
  - Name - string
  - Samples – List<SampleConfig>

- SampleResult
  - Properties:
    - Configuration - SampleConfig
    - SummaryResultUuid – GUID
    - Date – DateTime
    - CellCount – UInt32
    - ViableCellCount – UInt32
    - ViabilityPercent – double
    - AverageDiameter – double
    - AverageViableDiameter – double
    - AverageCircularity – double
    - AverageCellsPerImage – double
    - AverageBackgroundIntensity – double
    - BubbleCount – UInt32
    - ClusterCount – UInt32

- CellType
  - Properties:
    - Name – string
    - MinDiameter – double
    - MaxDiameter – double
    - NumImages – UInt32
    - Sharpness – double
    - MinCircularity – double
    - DeclusterDegree – uint32
    - NumAspirationCycles – UInt32
    - ViableSpotBrightness – double
    - ViableSpotArea – double
    - NumMixingCycles – UInt32
    - ConcentrationAdjustmentFactor – double

- QualityControl
  - Properties:
    - Name – string

        - CellTypeName – string
        - AssayParameter - enum
        - LotNumber - string
        - AssayValue – double
        - AcceptanceLimits - uint
        - ExpirationDate - DateTime
        - Comments – string

- SamplePosition – only used for processing plates
    - Properties:
        - Row – char
        - Column – uint32

Enumerations

- VcbResult
  - Values:
    - Error - 0
    - Success
    - NoConnection
    - NotLocked

- • ViCellStatus
  - Values:
    - Unknown
    - Idle
    - Initializing
    - Cleaning
    - Running
    - Error
    - Warning
    - RequiresUserInteraction

- • LockState
  - Values:
    - Unknown - 0
    - Locked
    - Unlocked

- • AssayParameter
  - Values:
    - Concentration
    - PopulationPercentage
    - Size

- • WashType – only used for Sample Sets (plate processing)
  - Values:
    - Normal
    - Fast

Certificates and Required Configuration

In order to connect to an instrument using the .NET API, you need to configure the OPC UA Security / Certificate validation of the .NET object. An example of how to configure this is provided below. Note that you must provide a callback function to validate the certificate. The callback may actually validate the certificate, or it can simply set the Accept flag to True. The callback function is called during the connection process.

C# Example code snippet:
_myBlu = new ViCellBLU();
SetOpcCertificate(_myBlu);
var result = _myBlu.Connect("username", "password", 192.168.1.1);
Note: username and password must be specified for a valid user.

Configure the .NET Object:
Note: The certificate validation callback function is set in this method.

```
// ****************************************************************
private static void SetOpcCertificate(ViCellBLU blu)
{
blu.OpcAppConfig.SecurityConfiguration.ApplicationCertificate = new CertificateIdentifier
{
StoreType = @"Directory",
StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\own",
SubjectName = "CN=Vi-CELL BLU Client, C=US, S=Colorado, O=Beckman Coulter, DC=" +
Dns.GetHostName()
};
blu.OpcAppConfig.SecurityConfiguration.TrustedIssuerCertificates = new CertificateTrustList
{
StoreType = @"Directory", StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\issuers"
};
blu.OpcAppConfig.SecurityConfiguration.TrustedPeerCertificates = new CertificateTrustList
{
StoreType = @"Directory", StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\trusted"
};
blu.OpcAppConfig.SecurityConfiguration.RejectedCertificateStore = new CertificateTrustList
{
StoreType = @"Directory", StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\rejected"
};
blu.OpcAppConfig.SecurityConfiguration.UserIssuerCertificates = new CertificateTrustList
{
StoreType = @"Directory", StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\issuerUser"
};
blu.OpcAppConfig.SecurityConfiguration.TrustedUserCertificates = new CertificateTrustList
{
StoreType = @"Directory", StorePath = @"%CommonApplicationData%\ViCellBlu_dotNET\pki\trustedUser"
};
blu.OpcAppConfig.SecurityConfiguration.AddAppCertToTrustedStore = true;
blu.OpcAppConfig.SecurityConfiguration.RejectSHA1SignedCertificates = false;
blu.OpcAppConfig.SecurityConfiguration.RejectUnknownRevocationStatus = true;
blu.OpcAppConfig.SecurityConfiguration.MinimumCertificateKeySize = 2048;
blu.OpcAppConfig.SecurityConfiguration.SendCertificateChain = true;
blu.OpcAppConfig.SecurityConfiguration.AutoAcceptUntrustedCertificates = true;
blu.OpcAppConfig.CertificateValidator.CertificateValidation +=
CertificateValidator_CertificateValidation;
}
```

Provide a certificate validation function:

```
// ************************************************************
private static void CertificateValidator_CertificateValidation(CertificateValidator validator,
CertificateValidationEventArgs e)
{
if (e.Error.StatusCode != StatusCodes.BadCertificateUntrusted)
{
return;
}
// Always accepts the certificate
e.Accept = true;
return;
}
```

Part Numbers:

Vi-CELL BLU Automation Kits

Part Number	Description
C57876	Vi-CELL BLU Automation Kit 1 with Sample Introduction Cup
C57877	Vi-CELL BLU Automation Kit 2 with Plate Holder

Vi-CELL BLU Compatible 96 well plates

Part Number	Description
C24841	600µL 96 well plates, Qty 5
C24842	96 well plate cover film, Qty 10

Vi-CELL BLU Reagent and Controls

Part Number	Description
C06019	Vi-CELL BLU Reagent Pack (Qty 1)
C09145	Vi-CELL BLU 50% Viability Control
C09147	0.5M Concentration Control, $0.5x10^6$/mL
C09148	2M Concentration Control, $2x10^6$/mL
C09149	4M Concentration Control, $4x10^6$/mL
C09150	10M Concentration Control, $10x10^6$/mL

Please reach out to a Beckman Coulter Sales Representative for a quote.

Limited Warranty

Beckman Coulter warrants that this software will substantially conform to the published specifications for the associated Product in which it is contained, provided that it is used on the computer hardware and in the operating system environment for which it was designed. Should the media on which your software arrives prove defective, Beckman Coulter will replace said media free of charge within 90 days of delivery of the associated Product. This is your sole remedy for any breach of warranty for this software. Except as specifically noted above, Beckman Coulter makes no warranty or representation, either expressed or implied, with respect to this software or its documentation including quality, performance, merchantability, or fitness for a particular purpose, or that this software or the associated Product does not infringer the intellectual property rights of any third party.

No Liability for Consequential Damages

In no event shall Beckman Coulter or its suppliers be liable for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or other pecuniary loss) arising out of the use of or inability to use this software. Because some states do not allow the exclusion or limitation of liability for consequential damages, the above limitation might not apply to you.