



Configuring CitectSCADA SNMP projects with GetIF

A reference for CitectSCADA Customers

Support Guide to GetIF and basic SNMP configuration

Introduction

As well as the monitoring and control of traditional IO on a SCADA network, CitectSCADA can be configured to communicate with SNMP Version 1¹ enabled network devices such as switches, routers, servers, and even PLCs /RTUs with SNMP capability.

The software is included with the CitectSCADA product and is pretty handy but unfortunately the official line is that we don't support it.

To live in a world without MIB2CIT, let's first understand what the application does and how we can use another tool, GetIF, to achieve the same results.

¹ At the time of writing this document, Citect does not support SNMP versions II & III

This document is made up of the following sections:

1. Using an alternative tool to MIB2CIT called GetIF to accomplish to configure SNMP projects
2. Testing traps using TrapGen.exe
3. Terms used in this document
4. Useful downloads

Intended Audience

This document has been written for CitectSCADA customers to assist in the setting up CitectSCADA SNMP projects.

Table of Contents.....	5
1 Introducing GetIF	6
1.1 Introducing GetIF; browsing MIBs and testing OIDs.....	8
1.1.1 The GetIF Landing Page.....	9
1.2 GetIF's MBrowser tab	10
1.3 Using the information identifiers (OIDs) with CitectSCADA DBF files.	11
1.3.1 SNMPVARS.DBF.....	11
1.3.2 VARIABLES.DBF	11
1.3.3 CitectSCADA Variable Tags.....	12
2 Generating Traps.....	13
2.1 Adding Traps to the project page	13
3 Useful Terminology.....	15
4 Compiling Private MIBs with GetIF (preparing a MIB to be read by GetIF).....	15
5 Additional Information.....	15

Life after MIB2CIT

When Citect's MIB2CIT works, it works well. The software is included with the CitectSCADA product and is pretty handy but unfortunately the official line is that we don't support it.

To live in a world without MIB2CIT, let's first understand what the application does and how we can use another tool, GetIF, to achieve the same results.

In order to use GetIF, it is useful to understand what Citect's MIB2CIT does so we can emulate the same tasks, albeit manually.

MIB2CIT uses its built in MIB browser to configure an SNMP project and populate the CitectSCADA DBF files necessary for CitectSCADA to communicate with our SNMP device (our PC in this example).²

There are two screens used in the MIB2CIT application, the _____ and the _____.

- The _____ requires two pieces of information only from the user, everything else is imported from the project:
 - Path to the Citect.INI
 - Select the CitectSCADA Project

- The _____ requires the user to select the following three parameters:
 - IODevice (IODev)
 - The SNMP OIDs (points in SCADA language) to monitor, taken from the MIB (SysUptime)
 - Unique Tag_suffix

² It is a good idea to check that the OIDs respond to a GET request before moving on to configuring the project further.

MIB2CIT - Tags

I/O Device: IODev (circled in red)
I/O Server: IOServer
Port: PORT1_BOARD1
Copy Tags

Tag Prefix/Suffix: Tag Suffix: JPC1 (circled in red)

Tag Name View: All, Device Only
Help, Close

MIBs: Add MIB, Clear MIB

Tree view: iso > org > dod > internet > directory > mgmt > mib-2 > system > sysUpTime (circled in red)

ObjectName = sysUpTime
ObjectOID = 1.3.6.1.2.1.1.3
DataType = TimeTicks
MAX-ACCESS = R
Description = The time (in hundredths of a second) since

Tag Name	OID	Data Ty...	Device	Add...
sysUpTime	.1.3.6.1.2...	LONG	IODev	100 (circled in green)

Buttons: Add >>, << Remove, Add Custom Item >>, Add Trap Tags >>, Group, Ungroup, Pack Address, Snmp (Get, Set, Trap, Events, Stats)

Device Address (MIB2CIT assigns this)

We can use GetIF in much the same way as MIB2CIT, to browse our MIB file and grab our OIDs to construct our variable tags.

The only difference is that tag information must be manually entered into the CitectSCADA .dbf files.

has a very simple MIB browser and can be used as a SNMP manager too, allowing us to run Get and SET requests ready for importing into your CitectSCADA Project.

What GetIF will do is create a device suffix and Device Address (Index) so the customer will need to do these things themselves but it's super easy!

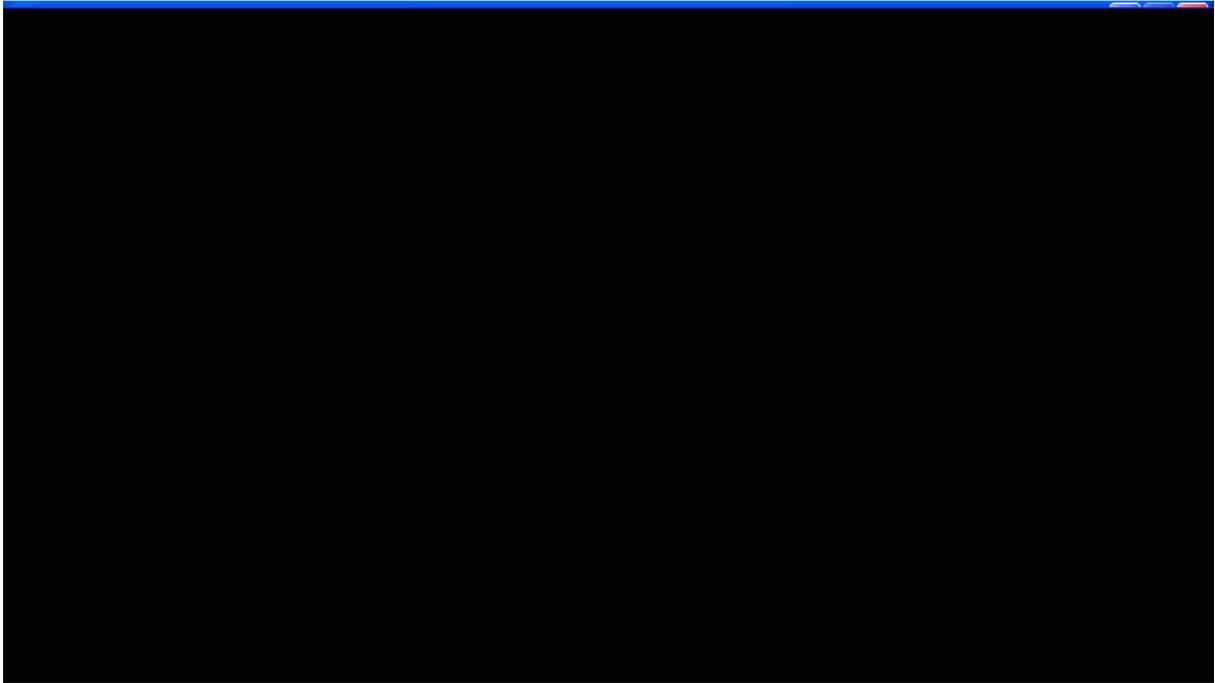
Download GetIF from here:

For our test, let's continue to use our own machines IP address of 127.0.0.1, so type 127.0.0.1 into the Hostname field of GetIF and click 'Start' to bring back some basic system information from your PC.

This basic information is part of a complimentary MIBII suite, which requires no fancy private MIBs to compile. MIBII is the same on all SNMP enabled devices. For this reason MIB2CIT and GetIF are already configured to read MIBII. MIBII includes common system information like System Location, System Description, System Uptime, and basic:945:L /B:Gj:jj8L /B:Gj:4xG:46xLt/B:G4959:Le/B:G;x&:6Lr/&G59;xxLf/

If you:945:L /B:Gj don't receive any information back from the

The MBrowser tab in GetIF allows you to browse the MIB tree and poll an OID in exactly the same way as you would have done with MIB2CIT.



You can poll the SysUpTime OID for example by clicking [here](#) to retrieve the uptime from your computer.

So now we have a way of identifying the OID within a device, we can begin to populate the CitectSCADA SNMPVARS.DBF and VARIABLES.DBF project files.

We will need to do the following manually:

- Assign a INDEX. This is a number unique to each type of Data Type (CTYPE). For example if you have 3 string entries, your Indexes could be 100,200,300 as shown below:

INDEX	DEVNAME	TAGNAME	CTYPE	SNMPNAME	OID
100	IODev	sysObjectID_TestPC1	STRING	sysObjectID	.1.3.6.1.2.1.1.2.0
200	IODev	sysContact_TestPC1	STRING	sysContact	.1.3.6.1.2.1.1.4.0
300	IODev	sysName_TestPC1	STRING	sysName	.1.3.6.1.2.1.1.5.0
100	IODev	sysUpTime_TestPC1	LONG	sysUpTime	.1.3.6.1.2.1.1.3.0

- Create a TagName (consisting of SNMP name and suffix)
- Populate the CType (data type)
- Populate the SNMPNAME and OID fields from your GetIF poll results

The information should look something similar the following:

100	IODev		_TestPC1	LONG	sysUpTime	.1.3.6.1.2.1.1.3
100	IODev		_TestPC1	STRING	sysName	.1.3.6.1.2.1.1.5

Next, we need to cross reference the information in the SNMPVARS.DBF file to the VARIABLES.DBF file so CitectSCADA can interpret an OID address and the device's logical address (Index) as a regular Tag address.

The only new piece of information here is the field used to identify the Device ID and OID.

sysUpTime_TestPC1	LONG	IODev	
-------------------	------	-------	--

We need to manually add in the variable tag address to the Variable Tags form in the CitectSCADA Project as shown below.

The screenshot shows the 'Variable Tags' configuration window for a project named 'SNMPTestProject'. The 'Variable Tag Name' is 'sysUpTime'. The 'Address' is 'N100'. The 'Data Type' is 'LONG'. The 'I/O Device Name' is 'IODev'. Other fields are empty or have default values. The status bar indicates 'Record: 1' and 'Linked: No'.

The preceding N in the INDEX field stands for 'Numeric' and the Address part (100) is taken from the INDEX. Data Type is LONG³

³ Refer to SNMPII Driver help > 'Data Types' for all the possible Data Types

In this section we will see how CitectSCADA handles traps. Firstly, make sure the SNMP Trap Service is running on your machine.

1. START > RUN > Services.msc

 SNMP Service	Includes a...	Started	Automatic	Local System
 SNMP Trap Service	Receives tr...	Started	Automatic	Local Service

2. Launch MIB2CIT and click on TAGS
3. Click the  button
4. That's it!

MIB2CIT has added the new Trap Tags to CitectSCADA, you can see them in the project editor under Tags > Variable Tags. Scroll through the tags and you will see the tags with addresses T0 through to T6 and TN created. The online help describes what these Tags mean and how Citect uses them.

Add in the new trap tags (T0-TN) to your project page. Create a button to call TagDebug and use this to READ and Write data to your addresses.

You can use a free tool called TrapGen (<http://www.ncomtech.com/trapgen.html>) to generate a Trap to your CitectSCADA project and have the results displayed in your page shown below. In the command line, run the following command to your own IP address with Generic Type 3:

```
C:\>TrapGen -d 127.0.0.1 -g 3
```

The results are now displayed on the screen.

SNMPTest

Pages Trends

Back

MIB2CIT SNMP Tag Test

1312553	Machine UpTime OID .1.3.6.1.2.1.1.3
SYD-D-DAVET	Machine Name OID .1.3.6.1.2.1.1.5

.1.3.6.1.4.1.2854 Enterprise

10.176.232.82 IP Address

SNMP Traps

Head	00000000	Generic Type	TrapOnLog
Write	00000000	Specific Type	TrapGenericType
Close	00000000	TrapTimeStamp	3
		TrapForward	127.0.0.1
		TrapNumber	1
		TrapNumberMib2	1

Additional Information

- - A MIB is a text file written in ASN.1 (adapted subset), which defines the data objects available from an SNMP "manageable entity". MIB2CIT and GetIF use the MIB to navigate to an OID
- - A numeric string that is used to uniquely identify an object within a MIB.
- - A clear text password (in SNMPV1 & 2) sent by the SNMP manager in order to Read () or Write () values in an .

Compiling additional MIBS - In order to add new private (SNMPv1) MIBS to the Getif browser, you must perform the following steps.

1. Be sure to shut down Getif
2. Copy your new MIB(s) into the MIBS directory under the Getif install directory (usually C:\Program Files\Getif 2.2\MIBS). Be sure that any requisite MIBS (some MIBS require that other MIBS are there) are also copied!
3. Delete the .index file in the the C:\Program Files\Getif 2.2\MIBS directory
4. Restart Getif

Ref: <http://www.wtcs.org>

- 1 GetIF: www.wtcs.org/snmp4tpc/FILES/Tools/SNMP/getif/getif-2.2.zip
- 2 Popular Device MIBs: <http://www.wtcs.org/snmp4tpc/FILES/Tools/SNMP/getif/GETIF-MIBS.ZIP>
- 3 List of Vendor Mibs: <http://www.iana.org/assignments/ianaiftype-mib>
- 4 TrapGen: <http://www.ncomtech.com/trapgen.html>

