



# Configuring CitectSCADA SNMP projects with GetIF

---

A reference for CitectSCADA Customers

**support@citect.com**

**Revision 1.00**

**Created 28-12-2009**

# Document revision history

---

Please update this table whenever you make an update to the document. Don't forget to increment the Cover Page

Revision author	Changes made	Date	New version No.

# 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 1<sup>1</sup> 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.

---

<sup>1</sup> 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

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.

## 1 Introducing GetIF

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).<sup>2</sup>

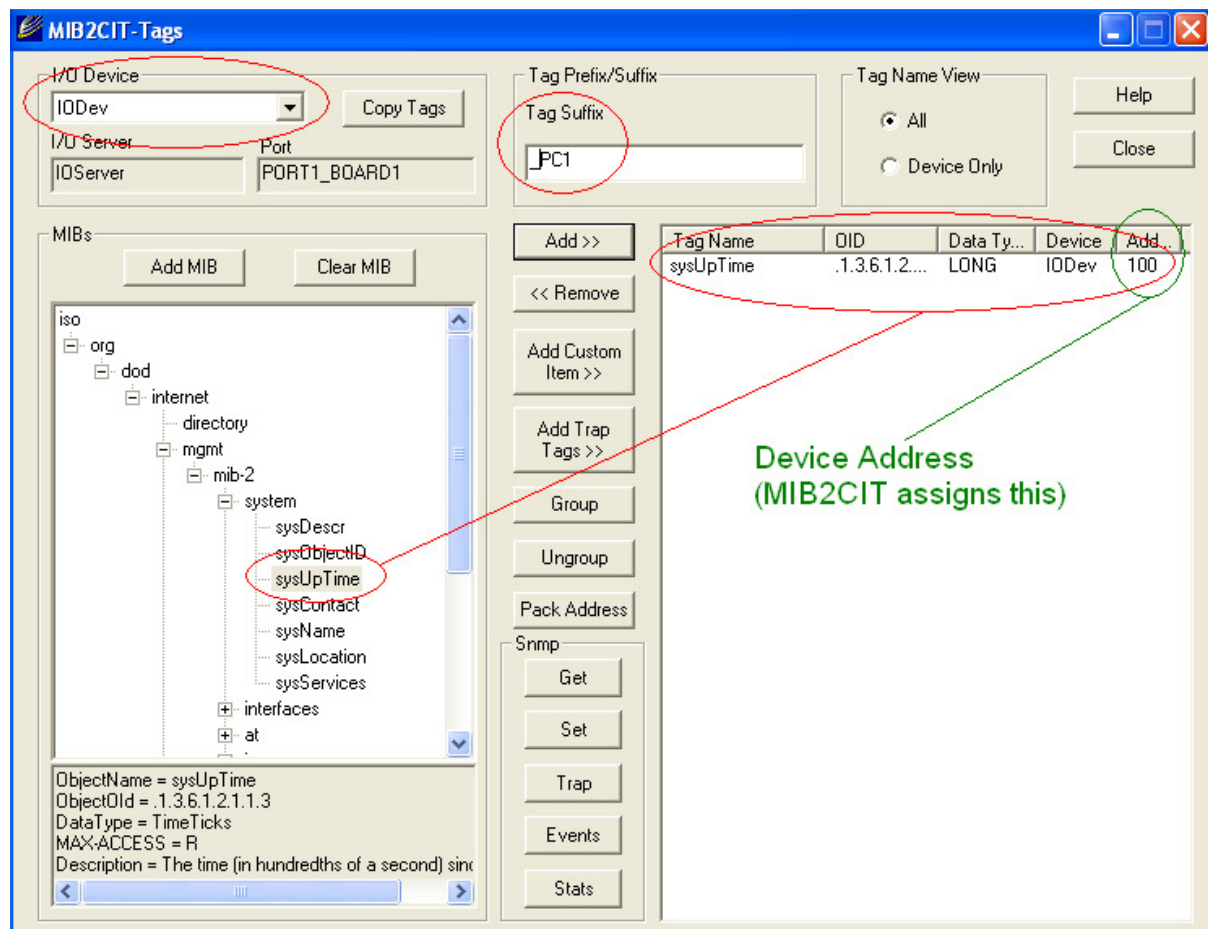
There are two screens used in the MIB2CIT application, the **Setup screen** and the **Tags screen**.

- The **Setup screen** 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 **Tags screen** requires the user to select the following three parameters:
  - IODevice (IODevice)
  - The SNMP OIDs (points in SCADA language) to monitor, taken from the MIB (SysUptime)
  - Unique Tag\_suffix

---

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

Figure 1: Showing the data we will replicate using GetIF



## 1.1 Browsing MIBs and testing OIDs.

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.

**GetIF** 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 **not** 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:

<http://www.wtcs.org/snmp4tpc/getif.htm>



### 1.1.1 The GetIF Landing Page

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 interface configuration status.

Figure 1: GetIF landing page

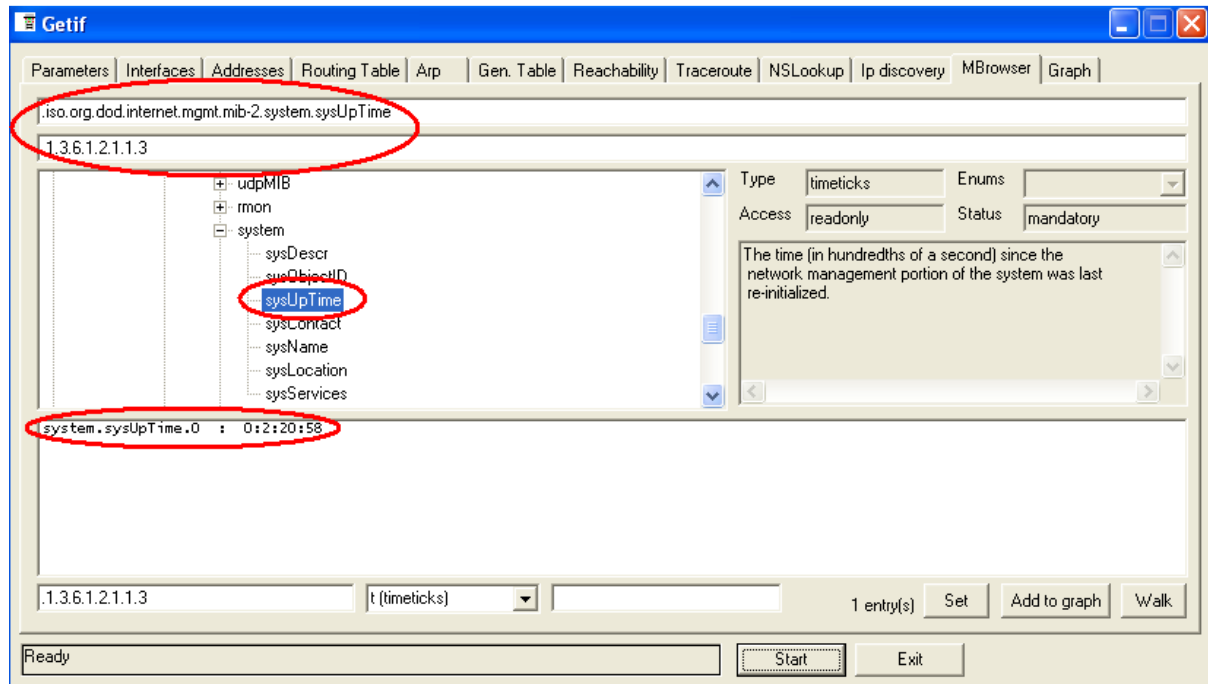
The screenshot shows the 'Getif' application window with the 'Parameters' tab selected. The 'Host name' field is set to '127.0.0.1' and is circled in red. Other fields include 'DNS name' (localhost), 'IP Address' (127.0.0.1), 'SysName' (SYD-D-DAVET), 'SysContact' (empty), 'SysLocation' (empty), 'SysDescr' (Hardware: x86 Family 15 Model 4 Stepping 10 AT/AT COMPATIBLE - Software: Windows 2000 Version 5.1 (Build 2600 Multiprocessor Free)), 'SysObjectID' (enterprises.microsoft.software.systems.os.windowsNT.1), and 'SysUpTime' (0:2:09:48). The 'SNMP Parameters' section shows 'Read community' (public), 'Write community' (private), 'Timeout (ms)' (2000), and 'Retries' (3). The 'Configuration' section has buttons for 'Set as default', 'Load default', and 'Factory settings'. The 'Telnet application' field is set to 'telnet.exe'. At the bottom, the 'Start' button is circled in red, and the 'Exit' button is also visible.

If you don't receive any information back from the device, check your SNMP parameters are correct and that your device is reachable over the network using PING. If this doesn't work; check your SNMP parameters are correct (top right).

## 1.2 GetIF's MBrowser tab

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.

Figure 2: Exploring the MIB Tree



You can poll the SysUpTime OID for example by clicking **Start** to retrieve the uptime from your computer.

## 1.3 Using the information identifiers (OIDs) with CitectSCADA DBF files.

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.

### 1.3.1 SNMPVARS.DBF

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:

Figure 3: SNMPVARS.DBF showing INDEX, SNMPNAME and OID

INDEX	DEVNAME		TAGNAME	CTYPE	SNMPNAME	OID
100	IODev		sysUpTime_TestPC1	LONG	sysUpTime	.1.3.6.1.2.1.1.3
100	IODev		sysName_TestPC1	STRING	sysName	.1.3.6.1.2.1.1.5

### 1.3.2 VARIABLES.DBF

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 **ADDR** field used to identify the Device ID and OID.

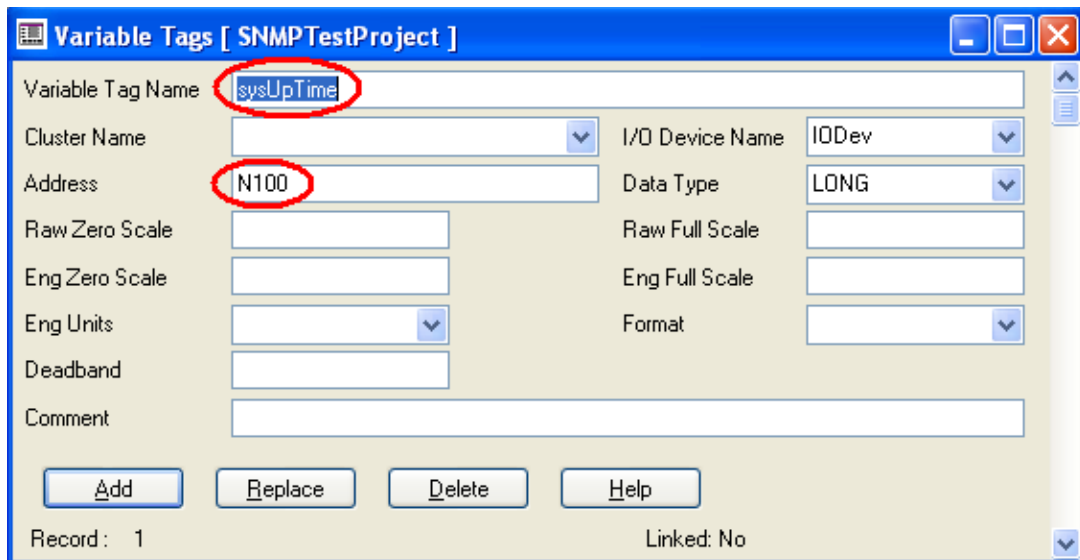
Figure 4: Showing VARIABLE.DBF with the ADDRESS field

NAME	TYPE	UNIT	ADDR
sysUpTime_TestPC1	LONG	IODev	N100

### 1.3.3 CitectSCADA Variable Tags

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

Figure 5: CitectSCADA Variable Tag creation using Address field from 2.4.2



The screenshot shows the 'Variable Tags [ SNMPTestProject ]' dialog box. The 'Variable Tag Name' field contains 'sysUpTime' and the 'Address' field contains 'N100'. Both fields are circled in red. The 'I/O Device Name' is 'IODev' and the 'Data Type' is 'LONG'. Other fields like 'Raw Zero Scale', 'Raw Full Scale', 'Eng Zero Scale', 'Eng Full Scale', 'Eng Units', 'Format', 'Deadband', and 'Comment' are empty. At the bottom, there are buttons for 'Add', 'Replace', 'Delete', and 'Help'. The status bar shows '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 LONG<sup>3</sup>



**All done!**

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

## 2 Generating Traps

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 **Add Trap Tags** 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.

### 2.1 Adding Traps to the project page

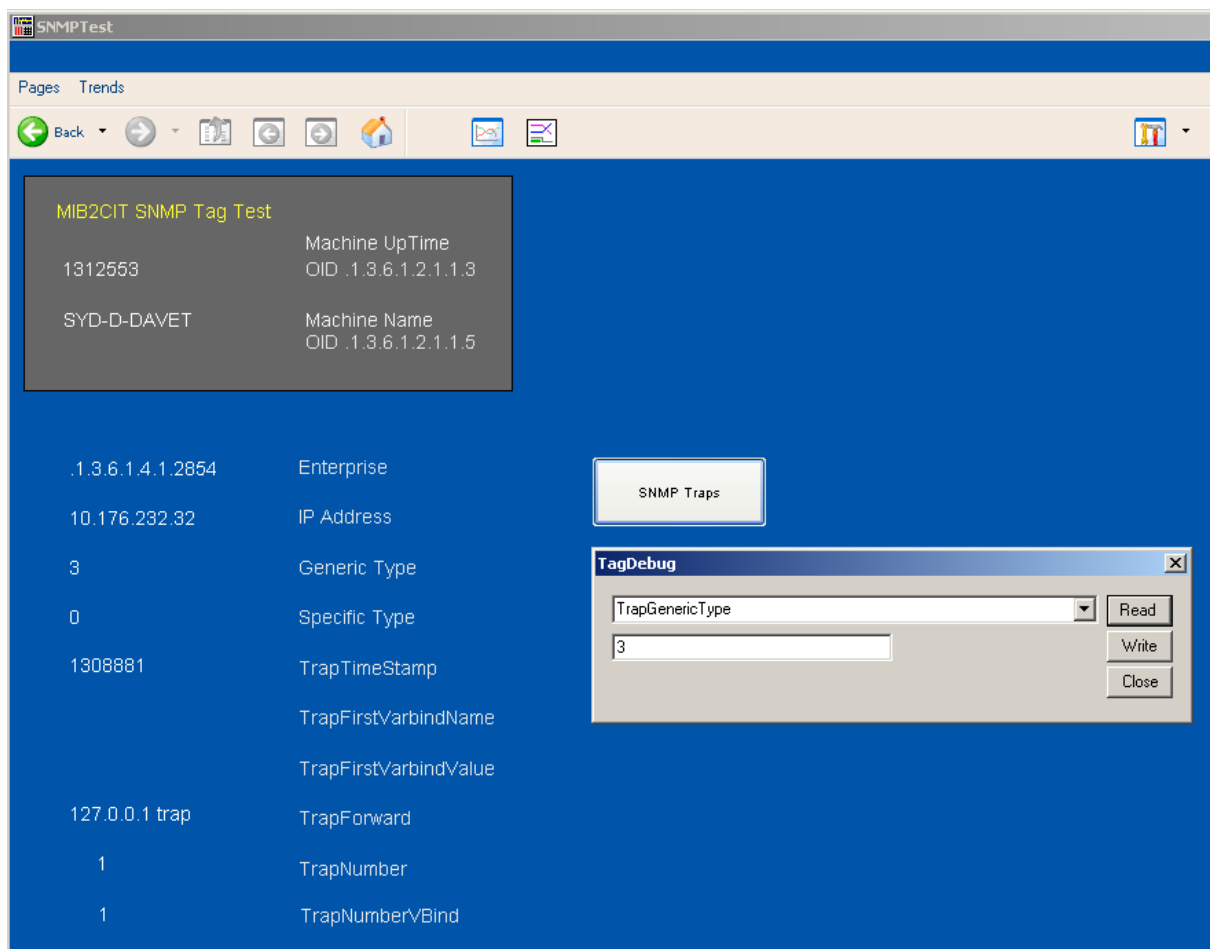
Add in the new trap tags (T0-TN) to your project page. Create a button to call TagDebug and use this to READ and Write data to your addresses.

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.

Figure 6: Traps received by TrapGen



# Additional Information

---

## 3 Useful Terminology

- **MIB: Message Information Block.**
  - 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
- **OID: Object Identifier**
  - A numeric string that is used to uniquely identify an object within a MIB.
- **Community String**
  - A clear text password (in SNMPV1 & 2) sent by the SNMP manager in order to Read (**Get**) or Write (**Set**) values in an **OID**.

## 4 Compiling Private MIBs with GetIF (preparing a MIB to be read by GetIF)

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>

## 5 Additional Information

- 1 GetIF: [www.wtcs.org/snmp4tpc/FILES/Tools/SNMP/getif/getif-2.2.zip](http://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>

