Application Report
SWRA671–June 2020

Cloning Z-Stack Network Properties Using the SimpleLink™ Wireless MCU Family

Marlyn Rosales

ABSTRACT

This application report describes how to clone the coordinator network properties within a ZigBee 3.0 mesh network using the SimpleLink CC13X2 and CC26x2 devices operating from Z-Stack 3.6.0 as included in the SimpleLink CC13X2/CC26X2 v3.4 SDK. The task is accomplished through the use of the TI Zigbee Network Properties Cloning Tool. The tool and its configuration files can be downloaded from the following URL: http://www.ti.com/lit/zip/swra671.

Contents

List of Figures

List of Tables

Trademarks

1 Introduction

The CC2652R device from Texas Instruments is the ideal System-on-Chip (SoC) from high-performance ZigBee applications, addressing many product specifications from a low-power standpoint. The CC2652R combines a powerful 48 MHz Arm Cortex-M4F CPU with up to 80KB of RAM and 352KB of on-chip flash. With a dedicated Radio Controller handling low-level RF protocol commands stored in ROM, it can handle complex network stacks ensuring ultra-low power and great flexibility.

In the world of IoT, battery life is tremendously valued by customers, cutting down on the bill of materials and battery replacement costs, while enabling easy maintenance and product convenience. Therefore, current consumption of devices inside a connected network must have their current consumption tightly controlled. The CC2652R is designed with the lowest power performance in sleep mode, active mode, and during sensor and data processing.

When range is an important consideration for an application, Texas Instruments offers the CC1352P device, which contains a +20 dBm integrated high-power amplifier with a best-in-class efficiency for long range applications. The CC1352P is a multiprotocol Sub-1 and 2.4-GHz with the same powerful system, offering the ability for a high-performance, long range ZigBee device.

This application report references examples from Z-Stack 3.6.0, which is based on the ZigBee 3.0 profile. Z-Stack comes packaged as part of the SimpleLink CC13X2/CC26X2 SDK, which is designed for simplified development within one environment using industry standard APIs, TI Drivers, and TI RTOS to provide a robust foundation for application development. The SDK version used in this report's test is v3.4.0.

There are different logical device types within a ZigBee Mesh Network: Coordinator, Router, and End Device. This application report focuses on the coordinator, for which there can only be one per network. The coordinator is responsible for forming and starting the network as well as managing the Trust Center (TC). If the coordinator were to fail, new devices would be prevented from joining the existing network since there would be no means of opening the network for joining, or exchanging network and application keys. Therefore, there is a need to back up the information contained in the coordinator and transfer it to a new device so that the new device can resume the role as coordinator. By being able to "clone" the device, the established network can continue as is, and new devices are able to successfully join. Section 4 discusses the TC and required Non-Volatile Memory (NVM) regions to be cloned. Afterwards, the setup and process to clone a Zigbee Coordinator (ZC) is discussed in Section 7 and Section 8 respectively.

NOTE: This application report is also applicable to Zigbee Network Processors (ZNPs)

2 Abbreviations and Acronyms

Table 1. Abbreviation and Acronyms

Abbreviation	Meaning
API	Application Process Interface
BDB	Base Device Behavior
EPID	Extended PAN ID
IC	Install Code
FCS	Frame Check Sequence
MT	Monitor and Test
NIB	Network Information Base
NPI	Network Process Interface
NVM	Non-Volatile Memory
NWK	Network
PAN	Personal Area Network
RX	Receive
SDK	Software Development Kit
SoC	System on Chip
SOF	Start of Frame
SREQ	Synchronous Request
SRSP	Synchronous Response
TC	Trust Center
TCLK	Trust Center Link Key
TX	Transmit
ZC	Zigbee Coordinator
ZED	Zigbee End Device
ZNP	Zigbee Network Processor
ZR	Zigbee Router

3 Tool Versions

The TI Zigbee Network Properties Cloning Tool along with any required files can be found here. There are three separate versions of the tool {CLI, MAC OS, Windows}. This application report focuses on the Windows version of the tool. For more detailed instructions that deal with the operation of the tool, see the device-specific READ ME files under the tool version folder of interest.

3.1 Command Line Interface (CLI)

This option is a singular python file that has the ability to be ran under a Windows, MAC OS, or Linux operating system. In order to run the CLI, the user must install the Pyserial and Questionary python library packages. This is usually done through the terminal commands 'pip install X' or 'pip easy_install X', where X represents the name of the library package.

In order to run the CLI, open a terminal window and navigate to the directory of the tool's folder. Once in the proper directory call the python file by typing 'py TI_Zigbee_Network_Properties_Cloning_Tool_CLI.py'. The python call "py" might be different depending on how python is configured within the machine. Depending on the platform and python version it could be "python", "python3", and so forth.

For more details on the configuration settings or how to use the CLI, see the READ ME file that is located within the CLI folder.

3.2 MAC OS Application

The MAC OS application was tested under OS versions 10.12.6 and 10.15.5. If running this application while on OS version 10.14.6, the application may cause the PC to log out the user. In order to avoid this, upgrade the OS versions to something other than 10.14.6.

To open the application for the first time, the user needs to approve the security restrictions and move the application into the "Applications" folder of the PC. When the application is opened, it will take a few seconds for the tool to scan for available ZC/ZNP devices upon startup. Once opened, the tool will operate in the same manner as the Windows instructions outlined in Appendix A. For more details on how to operate this tool, see the READ ME file located under the MAC OS folder.

3.3 Windows Executable

The TI_Zigbee_Network_Properties_Cloning_Tool.exe tool is meant to be ran on the windows operating system. Within the WINDOWS folder there are other files that the tool depends on to function properly such as the python37.dll. In order to ensure proper functionality of the tool, refrain from altering the file structure and or removing files from this directory. For more information on how to operate the tool, see Appendix A or the READ ME file within the Windows folder.

4 Trust Center and Non-Volatile Memory

In order to clone the network properties of a ZC it is important to recognize what kind of information is kept within the TC, and which of that information is crucial to maintaining the already formed network. Data within the TC is kept in non-volatile memory (NVM). The reason for this is that NVM can hold saved data even if the power is turned off. The necessary NV items to clone the network properties and a brief description as to the purpose of each is provided in Table 2.

Table 2. NV Memory Required Content

Content	NV Region Name	Description
Extended Address	ZCD_NV_EXTADDR	IEEE Identification Address (specific to each device)
Personal Area Network ID	ZCD_NV_PANID	2 byte ID used to represent the PAN
Extended PAN ID	ZCD_NV_EXTENDED_PAN_ID	By default the 64-bit EPID is set to the device's own IEEE Address. This value could be used during network joining instead of the 2 byte short address
On a Network Flag	ZCD_NV_BDBNODEISONANETWORK	Boolean used to represent if the device is part of a network (1= On NWK, 0= Not on NWK)
Group Table	ZCD_NV_GROUP_TABLE	Indicates the groups that have been formed between sets of end points in the network
Network Information Base	ZCD_NV_NIB	Contains the attributes required to manage the network layer of the device
Active Key Information	ZCD_NV_NWK_ACTIVE_KEY_INFO	The network key actively in use
Alternate Key Information	ZCD_NV_NWK_ALTERN_KEY_INFO	Latest alternate key proposed by the coordinator.
Network Security Table	ZCD_NV_EX_NWK_SEC_MATERIAL_TABLE	Stores the frame counter and the extended PAN ID of the ZC device
Trust Center Link Key Table	ZCD_NV_EX_TCLK_TABLE	Includes the TX/RX frame counters, extended address, key attributes/type, and the seed shift index of the corresponding devices added to the network
Trust Center Link Key IC Table	ZCD_NV_EX_TCLK_IC_TABLE	Includes the key and address of corresponding devices added to the network through the use of install codes.

Each of the entries in Table 2 is crucial to cloning the network properties of a ZC, and is the minimum information required. An item identification (Item ID) is given to each item in NVM so the NV driver is able to identify the location of where it is stored within Flash memory space. Within the example projects found in the SimpleLink CC13X2/CC26X2 SDK, the 'zcomdef.h' file (Stack -> Sys -> zcomdef.h') contains the item IDs for each of the NVM regions. Memory regions described in Table 3 contain Sub IDs and the table or list itself has a System ID. The value associated to each System ID can be found within the SimpleLink CC13X2/CC26X2 SDK under each project in the nvintf.h file (Application -> Services -> nvintf.h).

Table 3. System Identification

NV Region	System	System ID
ZCD_NV_EX_LEGACY	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_ADDMGR	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_BINDING_TABLE	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_DEVICE_LIST	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_TCLK_TABLE	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_TCLK_IC_TABLE	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_APS_KEY_DATA_TABLE	Z-Stack	NVINTF_SYSID_ZSTACK
ZCD_NV_EX_SEC_MATERIAL_TABLE	Z-Stack	NVINTF_SYSID_ZSTACK
ZCL_PORT_SCENCE_TABLE_NV_ID	Application	NVINTF_SYSID_APP
ZCL_PORT_PROXY_TABLE_NV_ID	Application	NVINTF_SYSID_APP
ZCL_PORT_SINK_TABLE_NV_ID	Application	NVINTF_SYSID_APP

The system ID is used to specify the system in which the tables or lists are a part of (for example, Zigbee, 15.4, Application). Depending on the number of max entries to a table or list (configurable at compile time), see Table 4. There will be that many locations in Flash reserved for information.

Table 4. NV Region Tables Required for Zigbee Coordinator Cloning

NV Region	Bytes Per Entry	Pre-Configurable Value	Default
ZCD_NV_EX_NWK_SEC_MATERIAL_TABLE	12	-	1
ZCD_NV_EX_TCLK_TABLE	20	ZDSECMGR_TC_DEVICE_MAX	40
ZCD_NV_EX_TCLK_IC_TABLE	20	ZDSECMGR_TC_DEVICE_MAX	40
ZCD_NV_GROUP_TABLE	322	APS_MAX_GROUPS	1

Sub IDs start at the value 0x0000 and increment upward to the entry of choice, maximum value configurable through the table presented above.

NOTE: For the purpose of this report and application, only the entries defined in Table 2 will be discussed.

5 Multi-Page NVM

Z-Stack 3.5.0 and onward has a multi-page NVM that grants users of the TI Z-Stack more flexibility in the amount of devices used within the network and the space required to store that information. Previously the NVM size used to be two pages (one for storage and one for compaction) while using the NVOCTP driver. Now, the NVM has been extended to five pages (four for storage and one for compaction) with the NVOCMP driver. With this transition came moving all tables out of 'ZCD_NV_EX_LEGACY' and creating new item IDs for each table/list. The TI Zigbee Network Properties Cloning Tool was created with the new implementation of NVM layout. If a device is running a project from Z-Stack 3.4.0 or earlier then the predefined symbol 'ZSTACK_NVOCMP_MIGRATION' must be defined in the new (Z-Stack 3.5.0 or newer) project in order to compensate for the changes involved with upgrading from the NVOCTP to the NVOCMP driver and to retain the NV memory sections as-is. For more details on this topic, see the *Non_Volatile Memory Items* section within 'Z-Stack Overview' in the TI Z-Stack User's Guide.

6 MT Interface

The MT API provides the structure of the commands sent to the device over the serial/UART connection in order for a host tester to communicate with a Zigbee device. In order to properly clone the network properties of a ZC, the contents in the memory regions for the entries in Table 2 will need to be read, stored, and then written to a new device that will take the place of the existing coordinator. The procedure for enabling MT commands on the ZC is in Section 8. For this application the SYS_OSAL_NV_READ, SYS_OSAL_NV_WRITE, SYS_NV_LENGTH, SYS_NV_READ, and SYS_NV_UPDATE commands are used to accomplish the task of cloning the ZC. Below is a summary of the frame content for these MT commands.

Table 5. Representation of Bytes within MT Frames

Frame Component	Description
Start of Frame	Represented by the hexadecimal byte, '0xFE', and is used to signify the beginning of the frame sent from either the host or the device itself.
Length	A byte that is used to indicate the length of the frame header excluding the CDM0, CMD1, and FCS bytes. In order to read the entire frame it is important to account for the bytes not represented by the length byte.
Command (CMD0)	Represents the type, upper 4 bits, and subsystem, lower 4 bits, of frame to be sent or received. The API guide lists the different types and subsystems. The 'S' before request (REQ) and response (RSP) signifies a synchronous response. The subsystem of interest is the system (SYS) type, represented by byte 0x01.
Command (CMD1)	8-bit command ID code, which maps to a specific interface message for the Subsystem specified in Cmd0. The CMD1 byte will be the same in the SRSP as it is in the SREQ sent to the device.
Sys ID	This value is only applicable to tables/lists listed in Table 3. It represents the system in which the tables/lists are associated to. The ID codes for this field can be found in the 'nvintf.h' file within (Application-> Services -> nvintf.h).
Item ID/ ID	Signifies a unique ID to be interpreted by the NV driver for particular memory content. As previously stated this value can be found within the 'zcomdef.h' file in the Z-Stack.
Sub ID	Sub IDs are only applicable to tables/lists listed in Table 3; it is used to index an entry in the specified table/list.
Offset	How far to start reading from the beginning of the memory region. In most cases this value is always zero since the entire memory content of an item is desired to be read or written to.
Frame Check Sequence	Byte responsible for error checking. Starting with the length byte each byte is XOR'ed with each other until the last byte before the FCS. This value is then XOR'ed with the FCS byte. If the final result is 0 then the check was successful, if it is 1 then there was a problem in the frame delivery. To send a frame this value must be calculated.
Status	Delivered from the device in order to indicate whether the read or write was successful. A zero means the read or write was successful, a one means it was unsuccessful.
LEN	The amount of bytes found within the data part of the frame. Not to be mistaken with the length portion of the frame.
Value	The content stored in the NVM of the requested location specified in the frame while taking into account the offset.

NOTE: Refer to the Z-Stack Monitor and Test API for specific details about the MT API Interface

7 Zigbee Coordinator Setup Procedure

In order to clone the network properties of a ZC, there needs to be a way to communicate with the device. This is similar to a Zigbee Network Processor where the Zigbee device just executes commands specific to the network, but the application is hosted on another MCU. In this case, the coordinator needs to be able to read in MT commands and respond accordingly. To do this the coordinator project that is flashed onto the device needs to include the MT and Network Processor Interface (NPI) APIs. The NPI is needed so that a host microcontroller (MCU) or platform is able to control the CC13xx/CC26xx device by sending serial commands. This section explains the required steps needed in order to use the TI Zigbee Network Properties Cloning Tool. The setup procedure is to be done on a fresh device before installation or network setup.

Refer to the 'Adding MT to a SimpleLink CC13x2/26x2 SDK Zigbee 3.0 Project' for detailed steps on incorporating the MT and NPI APIs into an existing Z-Stack Coordinator project. As a modification to the documentation, for the purpose of accomplishing this task the following change is made to step 3, *Include Predefined Symbols and Options*: along with the already stated predefined symbols the symbols in Table 6 also need to be incorporated. Ensure the project also contains the 'NV_RESTORE' predefine for the NV items to be retained after a reset.

Table 6. Predefined Symbols for MT/NPI APIs

Predefined Symbol	Purpose
FEATURE_NVEXID	Required in order to use the extended MT API commands.
MT_SYS_KEY_MANAGEMENT	Grants access to the security key data

After completing step 3, the project is ready to be flashed onto the device. Ensure to erase all of the NVM before flashing projects onto the device to avoid strange behavior based on old stored NV data. To erase the contents in the device use UniFlash. Once the project has been flashed onto the ZC device, start the BDB commissioning process. If using a LaunchPad device, pressing (BTN-1) will automatically start the BDB commissioning process. As soon as the network is formed other ZED/ZR devices may join the newly formed NWK.

In the case that the ZC has fallen off the network and the user wishes to clone it then the tools attached to this application report can be used. Note that the coordinator that will replace the pre-existing coordinator must also have its NVM erased and then have the same project file flashed with the inclusion of the MT and NPI APIs. The TI Zigbee Network Properties Cloning Tool will only detect devices with these APIs included.

8 Zigbee Coordinator Cloning Procedure

The TI Zigbee Network Properties Cloning Tool is an executable that allows the user to clone the existing network properties of a ZC to another ZC. The tool also has the capability to read and write to a ZC device. For details on how to operate the GUI and its functionality, see Appendix A.

There are two methods that can be used to clone the network properties of a ZC. Method one, described in Section 8.1, requires that both devices be available while the procedure will take place. Method two, described in Section 8.2, should be used when both devices are not available simultaneously.

8.1 Cloning: Method One

The following are steps used to clone the network properties of a ZC which underwent the procedure in Section 7:

1. Download and unzip the contents found here.
2. Connect the coordinator device (ZC1) to the PC and note the COM Port. *
3. Connect the new coordinator device (ZC2) to the PC and note the COM Port. *
4. Run the TI Zigbee Network Cloning Tool by clicking on the executable file within the contents of the unzipped folder.

NOTE: Step (4) can happen before steps (2) and (3). If so, within the tool go to *Tools -> Scan for Devices* in order to update the COM ports.

5. Select the 'Clone' application option within the tool, and then the corresponding COM Ports for the devices to be read and written to. Content should be read from ZC1 and written to ZC2.
6. Within the tool go to *Tools -> Form Network.* This will form a network on ZC2. It is important to form a network first so that ZC2 can have a Network Information Base (NIB) before writing to it.
7. Press the 'Start' button, and watch the output to make sure there are no errors. To view the content that was read from ZC1, open the read file under *View -> Read File* once the reading procedure is complete.
8. Remove ZC1 from the NWK by powering it off.
9. After completion, power cycle or restart ZC2. This is necessary for the ZC to update its NV content with the new content that was written. Once the device restarts, it will take the place of ZC1. **

8.2 Cloning: Method Two

Below is a list of steps used to clone the network properties of a ZC which underwent the procedure in Section 7:

1. Download and unzip the contents found here
2. Connect the coordinator device (ZC1) to to the PC and note the COM Port*
3. Run the TI Zigbee Network Cloning Tool by clicking on the executable file within the contents of the unzipped folder.

NOTE: Step (3) can happen before step (2). If so, within the tool go to *Tools -> Scan for Devices* in order to update the COM ports.

4. Select the 'Read' application option within the tool, and then the corresponding COM port for the device to be read. Choose the 'All Regions' selection in the 'Read Settings' portion to read all of the content available.
5. Press 'Start' to begin the reading procedure. To view the content that was read from ZC1, open the read file under *View-> Read File* once the reading procedure is complete. There may not be content for all of the NV regions. This is dependent on the type of application used.
6. Whenever ready, connect ZC2 to the PC and note the COM port.*
7. Run the TI Zigbee Network Cloning Tool by clicking on the executable file within the contents of this folder if the application is not running yet.
8. Select the 'Write' application option within the tool, and then the corresponding COM port for the device to be written to. If refreshing the COM ports is desired go to *Tools -> Scan for Devices*.
9. Within the 'Write Settings' segment choose to 'Write from a File'. Once the popup window appears, select the file 'content_read.txt' (or file saved from reading from ZC1). If a successful upload was done a notification of green text 'Uploaded' should appear next to the 'Write from File' option.
10. Within the tool go to *Tools -> From Network*. This will form a network on ZC2. It is important to form a network first so that ZC2 can have a NIB before writing to it.
11. Press the 'Start' button, and watch the output to make sure there are no errors. To view the content that was written to ZC2 check the output display for success or failure status.

12. Remove ZC1 from the NWK by powering it off.
13. When the process is complete, power cycle or restart ZC2. This is necessary for the ZC to update its NV content with the new content that was written. Once the device restarts, it will take the place of ZC1. **

* For a description on how to detect the required COM Port under a Windows operating system, reference Section B.2.

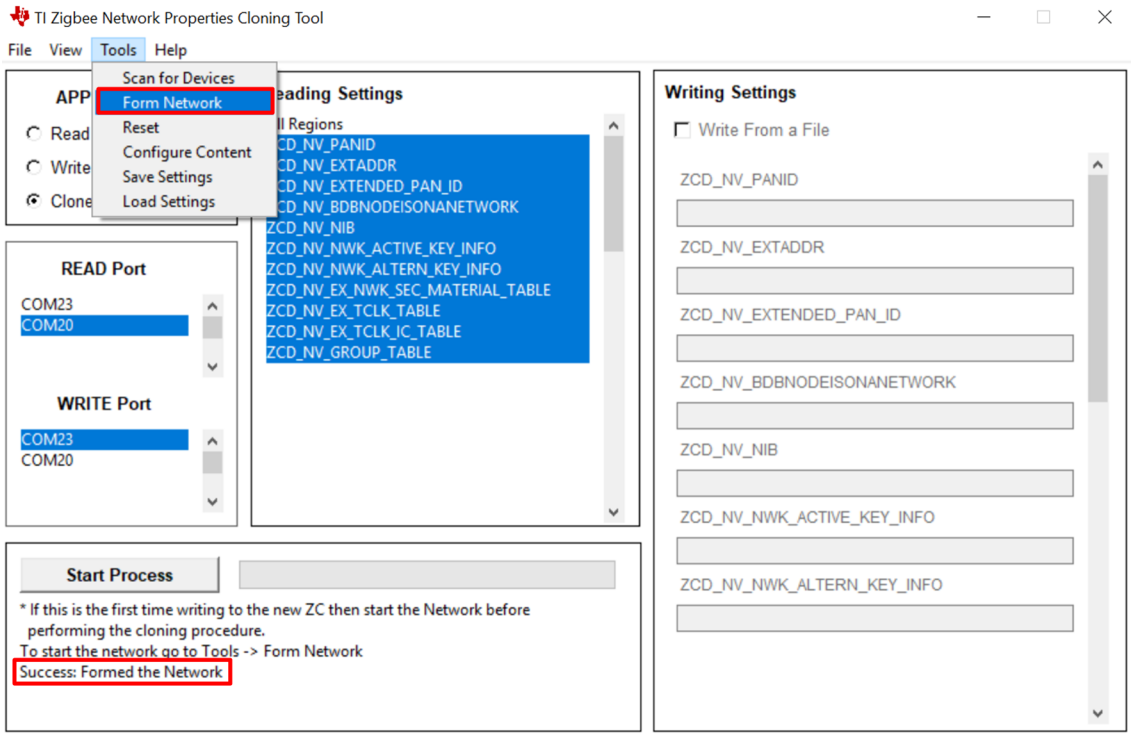
** If ZC2 was programmed with an XDS110 emulator then the first time the device is reset after programming the reset must be physical. For a physical reset, power cycle the device or if using a Launchpad press the reset button. Otherwise, within the tool navigate to *Tools -> Reset* in order to reset the device.

9 Zigbee Coordinator Cloning Procedure Example

This section provides a detailed example of the procedure outlined in Section 8. A sniffer log was used to ensure proper functionality. Before executing this example, a network was formed with a Zigbee Coordinator (ZC1), a CC135xR1 Launchpad running a modified version of the ZC light example project following the setup from Section 7. After the network was created, two ZRs joined the network.

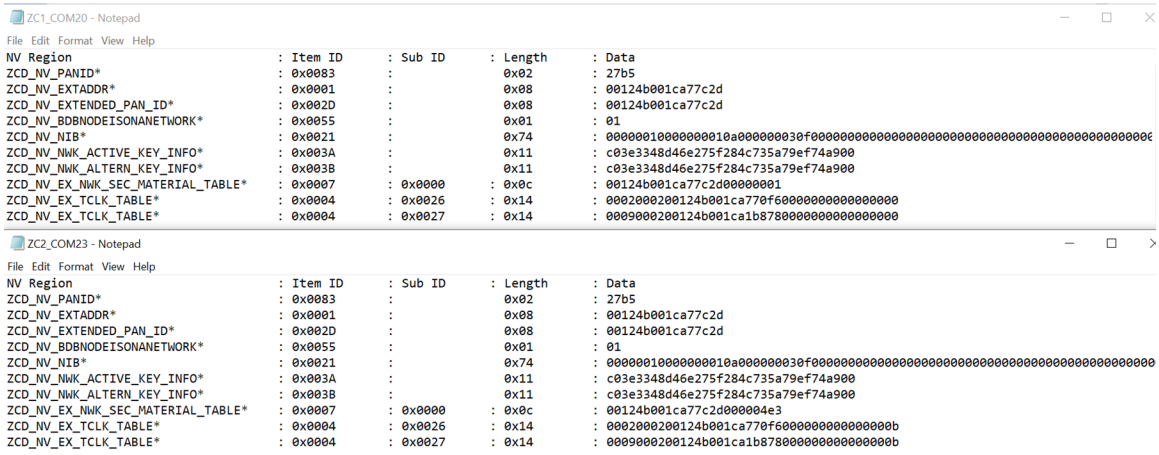
1. Identify the COM Ports for the devices: ZC1- COM20, ZC2- COM23
2. Open the TI Zigbee Network Properties Cloning tool, and select the 'Clone' option then the designated ports from step 1. Afterwards, go to *Tools -> Form Network.* A network will be formed on ZC2 when the 'Clone' option is selected. If a network is formed the output should show a success, if not a failure message will be displayed.

Figure 1. Formed Network Example



3. Press the 'Start' button and observe the output display to verify what was read and written. If desired the content from both ZC1 and ZC2 can be compared by analyzing the differences between their read_content.txt file output after reading from each device separately. If the content from both files is the same then the procedure was successful.

Figure 2. NV Content Read Example



ZC1_COM20 - Notepad

File Edit Format View Help

```
NV Region                       : Item ID     : Sub ID     : Length     : Data
ZCD_NV_PANID*                   : 0x0083      :              0x02       : 27b5
ZCD_NV_EXTADDR*                 : 0x0001      :              0x08       : 00124b001ca77c2d
ZCD_NV_EXTENDED_PAN_ID*         : 0x002D      :              0x08       : 00124b001ca77c2d
ZCD_NV_BDBNODEISONANETWORK*     : 0x0055      :              0x01       : 01
ZCD_NV_NIB*                     : 0x0021      :              0x74       : 0000001000000010a000000030f0000000000000000000000000000000000000000000000000
ZCD_NV_NWK_ACTIVE_KEY_INFO*     : 0x003A      :              0x11       : c03e3348d46e275f284c735a79ef74a900
ZCD_NV_NWK_ALTERN_KEY_INFO*     : 0x003B      :              0x11       : c03e3348d46e275f284c735a79ef74a900
ZCD_NV_EX_NWK_SEC_MATERIAL_TABLE* : 0x0007    : 0x0000     : 0x0c       : 00124b001ca77c2d00000001
ZCD_NV_EX_TCLK_TABLE*           : 0x0004      : 0x0026     : 0x14       : 0002000200124b001ca770f60000000000000000
ZCD_NV_EX_TCLK_TABLE*           : 0x0004      : 0x0027     : 0x14       : 0009000200124b001ca1b8780000000000000000
```

ZC2_COM23 - Notepad

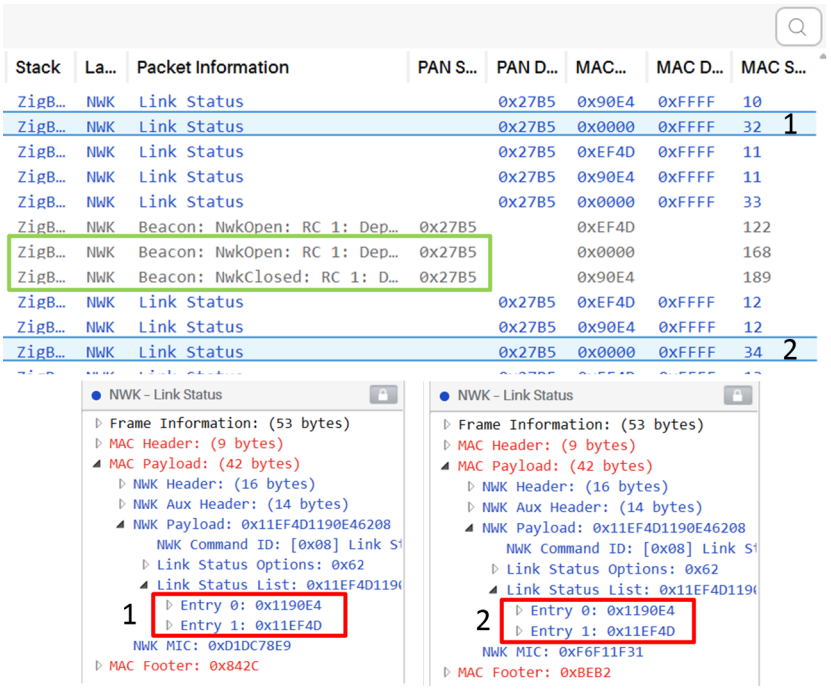
File Edit Format View Help

```
NV Region                       : Item ID     : Sub ID     : Length     : Data
ZCD_NV_PANID*                   : 0x0083      :              0x02       : 27b5
ZCD_NV_EXTADDR*                 : 0x0001      :              0x08       : 00124b001ca77c2d
ZCD_NV_EXTENDED_PAN_ID*         : 0x002D      :              0x08       : 00124b001ca77c2d
ZCD_NV_BDBNODEISONANETWORK*     : 0x0055      :              0x01       : 01
ZCD_NV_NIB*                     : 0x0021      :              0x74       : 0000001000000010a000000030f0000000000000000000000000000000000000000000000000
ZCD_NV_NWK_ACTIVE_KEY_INFO*     : 0x003A      :              0x11       : c03e3348d46e275f284c735a79ef74a900
ZCD_NV_NWK_ALTERN_KEY_INFO*     : 0x003B      :              0x11       : c03e3348d46e275f284c735a79ef74a900
ZCD_NV_EX_NWK_SEC_MATERIAL_TABLE* : 0x0007    : 0x0000     : 0x0c       : 00124b001ca77c2d000004e3
ZCD_NV_EX_TCLK_TABLE*           : 0x0004      : 0x0026     : 0x14       : 0002000200124b001ca770f6000000000000000b
ZCD_NV_EX_TCLK_TABLE*           : 0x0004      : 0x0027     : 0x14       : 0009000200124b001ca1b878000000000000000b
```

This image shows an example of what was read from ZC1 (Top) and from ZC2 (Bottom). The contents of TCLK were shorted to only show the last two table entries. This is so because the TCLK table gets populated from the end of the table first.

4. Remove ZC1 from the NWK.
5. Power Cycle or Restart ZC2. After ZC2 is part of the already existing network it will send out periodic link statuses. Within the status should be the devices that had joined when ZC1 was the coordinator.

Figure 3. Sniffer Log Capture Example



Stack	La...	Packet Information	PAN S...	PAN D...	MAC...	MAC D...	MAC S...
ZigB...	NWK	Link Status		0x27B5	0x90E4	0xFFFF	10
ZigB...	NWK	Link Status		0x27B5	0x0000	0xFFFF	32
ZigB...	NWK	Link Status		0x27B5	0xEF4D	0xFFFF	11
ZigB...	NWK	Link Status		0x27B5	0x90E4	0xFFFF	11
ZigB...	NWK	Link Status		0x27B5	0x0000	0xFFFF	33
ZigB...	NWK	Beacon: NwkOpen: RC 1: Dep...	0x27B5		0xEF4D		122
ZigB...	NWK	Beacon: NwkOpen: RC 1: Dep...	0x27B5		0x0000		168
ZigB...	NWK	Beacon: NwkClosed: RC 1: D...	0x27B5		0x90E4		189
ZigB...	NWK	Link Status		0x27B5	0xEF4D	0xFFFF	12
ZigB...	NWK	Link Status		0x27B5	0x90E4	0xFFFF	12
ZigB...	NWK	Link Status		0x27B5	0x0000	0xFFFF	34

1

2

NWK – Link Status

```
Frame Information: (53 bytes)
MAC Header: (9 bytes)
MAC Payload: (42 bytes)
  NWK Header: (16 bytes)
  NWK Aux Header: (14 bytes)
  NWK Payload: 0x11EF4D1190E46208
    NWK Command ID: [0x08] Link St
    Link Status Options: 0x62
    Link Status List: 0x11EF4D119(
      Entry 0: 0x1190E4
      Entry 1: 0x11EF4D
  NWK MIC: 0xD1DC78E9
MAC Footer: 0x842C
```

1

NWK – Link Status

```
Frame Information: (53 bytes)
MAC Header: (9 bytes)
MAC Payload: (42 bytes)
  NWK Header: (16 bytes)
  NWK Aux Header: (14 bytes)
  NWK Payload: 0x11EF4D1190E46208
    NWK Command ID: [0x08] Link St
    Link Status Options: 0x62
    Link Status List: 0x11EF4D119(
      Entry 0: 0x1190E4
      Entry 1: 0x11EF4D
  NWK MIC: 0xF6F11F31
MAC Footer: 0xBEB2
```

2

This image is part of the sniffer log that was captured for this example. In link status (1), sent by the coordinator device ZC1 (0x0000), we can confirm that there were two devices who were part of the network. The green box shows ZC2 joining the network as ZC1 leaves signified by the opening and closing of the network. Link status (2), sent by the new coordinator ZC2, still shows the same devices as part of its network.

6. The network should continue to operate as before.

10 Other Application Considerations

With this procedure it is possible to be able to clone any ZC device that is operating in a network. This application report and the associated resources are for demonstrative purposes. The implementations discussed are left to the user to apply in the case a more secure system is desired.

Nothing is preventing illicit NV memory reads from unwanted sources. Thus, when discussing security in ZC cloning it is important to take precautions. One possible way to accomplish this is to have a password protected device. In order to be able to read certain memory regions of the device then the user must enter said password. Refer to Understanding Security Features for SimpleLink™ Zigbee CC13x2 and CC26x2 Wireless MCUs for both device and OTA Zigbee considerations.

Another implementation is backing up the coordinator every time a new device joins the network. This can be done by sending a message to the host processor when there is a device announcement to let the host know to kick off the backup. The reason this is important is for the circumstances when the issue with the coordinator is in the memory itself. Therefore, if the memory gets corrupted, the contents of the TC would still be saved up to the point of the last time a device joined the network and updated key information. The tool provided only gives base functionally of cloning a ZC device at a time of interest.

11 Summary

In conclusion, this application report discussed how to use the TI Zigbee Network Properties Cloning Tool and the MT API to perform ZC cloning by reading and writing pertinent NV items. The tool is versatile enough to have the user specify which flash memory contents they would like to back up. The information that is read from the existing coordinator is also stored for the user to analyze if they wish to do so. Cloning will allow recovery of Zigbee networks for which the coordinator device has failed. Further application specific features are left to the user to implement.

12 References

- TI Z-Stack Monitor and Test API
- SimpleLink CC13X2/CC26x2 SDK: http://www.ti.com/tool/SIMPLELINK-CC13X2-26X2-SDK
- Zigbee SimpleLink Academy (Look under the Zigbee folder from the SimpleLink CC13X2/CC26X2 SDK in http://dev.ti.com/tirex
- Texas Instruments Packet Sniffer: http://www.ti.com/tool/packet-sniffer
- Texas Instuments: *Understanding Security Features for SimpleLink™ Zigbee CC13x2 and CC26x2 Wireless MCUs*
- LAUNCHXL-CC1352P LaunchPad Development Kit: http://www.ti.com/tool/LAUNCHXL-CC1352P
- LAUNCHXL-CC26X2R1 LaunchPad Development Kit: http://www.ti.com/tool/LAUNCHXL-CC26X2R1

TI Zigbee Network Properties Cloning Tool Guide

A.1 Tool Layout

Figure 4. TI Zigbee Network Properties Cloning Tool

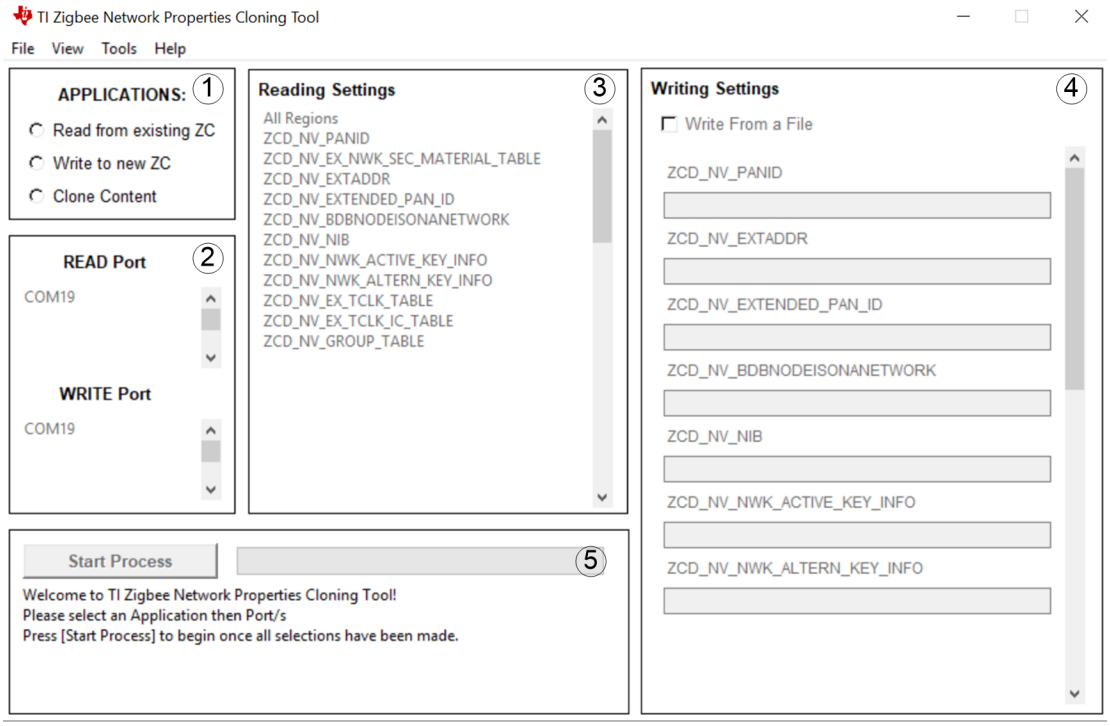


Table 7. Tool Layout Explanation

Number	Label	Section	Functionality
1	Application Selection	Section A.4	Choose the type of application to be used
2	COM Port/s	Section A.5	COM Port selection based on the application chosen
3	Read Settings	Section A.6	NVM Regions to be read from
4	Write Settings	Section A.7	NVM Regions to be written to
5	Output Display	Section A.8	Status updates indicating the tasks

A.2 Configuration File

The configuration file, *config_nv_regions.txt*, contains the content that the tool uses to pre-load with the NVM Regions and other necessary information. Content in this file is stored in the following manner, {NV Region : Item ID : System ID: Sub ID : Entries }. The 'System ID' and ‘Sub ID’ field is for tables/lists within Table 3. The Sub ID is to indicate which Sub ID to start indexing from. ‘Entries’ is used to represent how many items within the table/list region to read from. For all other nv regions these fields may be left empty. To separate the items within the file use the (:) operand.

This file can be modified to include other NV Regions, even custom items, than the ones specified. To add new NV items simply append new items with the necessary information following the format of the already existing file. Go to *Tools -> Configure Content* in order to open the configuration file. To view the reflected changes in the tool, either re-open the tool or go to *File -> Clear/Refresh*.

NV Regions that end with an (*) character are used to signify that they are required for cloning. The program will read and write to NV Regions for which have an (*) if the 'Clone' option within the application section is enabled. For NV Regions not in Table 2, do not incorporate a (*) character as part of the NV Region name.

A.3 Content File

The content file, *read_content.txt*, contains data which has been read from a ZC. The structure of the file is the following, {NV Region : Item ID : System ID: Sub ID : Length : Data}. Only tables/lists within Table 3 will display content for the System ID and Sub ID. Each entry of the table/list will be its own row in the file and the length field signifies the length of the content within the corresponding Sub ID. To view the file, go to *View -> Read File*. This file can be saved and used later to write the content to a different ZC. Every time a device is read this file will be overwritten unless saved intermediately under a different name.

A.4 Application Selection

At startup of the tool all functionally is disabled until an application is selected. There are three options {Read, Write, or Clone}. Cloning requires both current and new ZCs for the application to complete the task. If both devices are not available at once then the current ZC can be read from, then at a later time the information can be written to the new ZC with the ‘Write’ option.

A.5 COM Ports

COM Ports are filtered for Texas Instruments devices and a port type of ‘Application/User UART’. Among that only COM Ports which are associated to devices that have the capability to interface with the MT commands will be visible to the program. The tool will automatically scan for devices when it is run. If a device is open in another application (ex. serving as a sniffer or in a debug state within CCS) the tool will not register this device. To scan for device changes after the tool has been ran go to *Tools-> Scan for Devices*.

A.6 Read Settings

The ‘Read’ option in the application section enables the user to select which items to read from. All NV Regions specified within the configuration file will be visible using the ‘NV Region’ name. If ‘All Regions’ is selected as the option to read then all NV Regions that appear will be read from, else only the selected items will be read from. If there is an error reading a region or there is no content for the region then the tool will not read from the device; if this happens an error message will be displayed in the output display. To view the data that is read from a device go to *View -> Read File*. Within this file is the NV Region, Item ID, System ID (if any), Sub ID (if any), length of the data read, and the data itself.

A.7 Write Settings

The ‘Write’ option in the application section enables the user to select which NV items to write. There are two types of writing: from a file or use the tool to manually enter data. To write from a file simply check the 'Write from File' option and upload the desired file. If there was a successful upload then the 'Uploaded' status will appear to the right of the 'Write from File' checkbox. For manual entries, only the NV regions configured without a System ID, Sub ID, and entries field will appear as options for the NV Regions. To write to an NV region simply enter the desired value into the text box for the corresponding region. Data can be entered with a prefix of '0x' or just the data itself in hex representation. If the value written to an NV Region is not of correct length or type then the value will not be written to the NV Region and the user will be prompted of this through the output display.

A.8 Output Settings

The output display contains the ‘Start’ button used to run the program; alternatively, pressing the *File -> Start* option will also complete the same task. The progress bar to the right of the ‘Start’ button indicates how far a task is from completion. The output display box will deliver status updates for the user as an indication of what is being done or if there are any warnings and or errors. To store the content of this “log” after an operation is complete go to *View -> Export Log File*.

A.9 Save/Load Settings

Within the Tools menu tab is the option to save and load settings. Saving the settings enables the user to store all of the selections and entries made so that they may be pre-loaded at a different point in time. When loading the settings choose the file which was created from saving the settings. If there is an error with this process then the error message will be shown in the output display.

A.10 Help

Within the 'Help' section of the tool's menu bar is various resource links, such as this application report, and information the user can reference in order to understand how to properly use the tool or leverage it to implement something similar on their own.

Establishing a Serial Connection

B.1 Serial Connections

The TI Zigbee Network Properties Cloning Tool uses Python's PySerial package to establish a serial connection with the device and create a Network Process Interface (NPI). In order to establish a serial connection the following criteria needs to be defined: Port, Baudrate, Parity, Stop bits, and Bytesize.

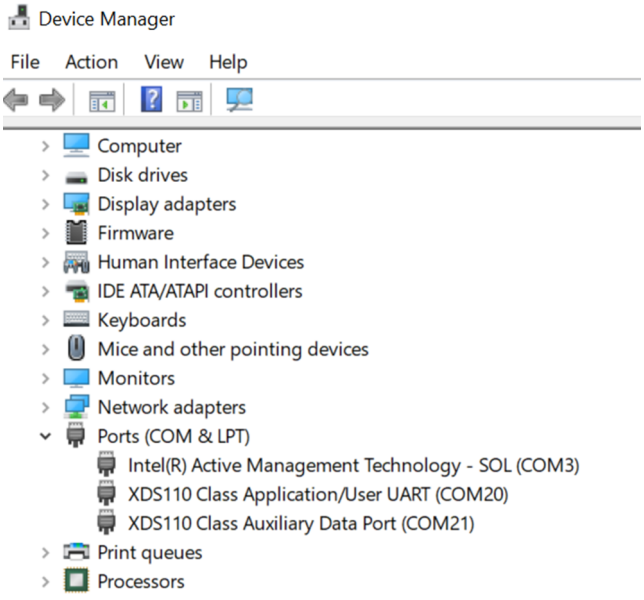
Table 8. Serial Connection Parameters

Criteria	Value	Explanation
Port	--	The value assigned to the application serial port interface used for the transmission of the data. This value will vary across devices and Personal Computers (PC)s. *
Baudrate	115200	The number of bits that are sent during the communication for a specific unit of time.
Parity	NONE	Parity is used as a method to detect errors in transmission. For the serial connection this is set to NONE.
Stop Bits	1	The amount of bits used to indicate the end of a frame
Bytesize	8	Number of data bits

B.2 Detecting COM Ports

In order to know which port to use for a machine running on a Windows operating system, connect the device in question to the host computer, and use the 'Ports (COM & LPT)' drop down in *Device Manager* to identify the port number of the device, refer to Figure 5 for an example.

Figure 5. COM Port in Device Manager



IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265