

# Debugging SDLC

Document ID: 5210

---

## Introduction

### Prerequisites

- Requirements

- Components Used

- Conventions

### Common Problems

### Troubleshoot

- debug sdlc

- debug sdlc local-ack

- debug stun packet

- debug sdllc

### Related Information

---

## Introduction

This document provides information about common problems with Synchronous Data Link Control (SDLC). It also shows how to use various **debug** commands to help resolve such problems.

## Prerequisites

### Requirements

There are no specific requirements for this document.

### Components Used

The information in this document is based on Cisco IOS® software with IBM Feature Set.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

### Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

## Common Problems

There are several common problems that can occur when SDLC is used, all of which are related to how the host is configured. Often, the solution is to check the line gen to determine what is happening. Sometimes, trial and error is also an effective method.

First, ensure that the serial line between the host and the router is in an up state. Then, ensure that the protocol is also in an up state. If you see them both in a down state, then there is probably a clocking issue. Determine which device is supposed to provide the clocking signal, and ensure that all cables are properly connected.

**Note:** The serial interface of the Cisco router is sensitive to data terminal equipment (DTE) and data communications equipment (DCE) cables and it can detect the type of cable that you are attaching. Therefore, you must use the correct cable. Refer to Troubleshooting Serial Lines for more information.

If you notice that the serial line is cycling between an up and a down state, one of these problems is probably occurring:

- Your host is configured for half-duplex instead of full-duplex.
- The router is not configured for half-duplex (if you are using a modem sharing device [MSD]).
- The cables are not strapped high (again, if you are using an MSD).

If the serial line is showing an up state, but the protocol is in a down state, then one of these problems is probably occurring:

- One end or the other of your host is set for nonreturn to zero inverted (NRZI), but you have not configured the router for NRZI.
- You have configured the router for NRZI, but the host is not set for NRZI.

You should either remove or add **nrzi-encoding** as appropriate to the router interface.

- To add **nrzi-encoding** to the serial interface, issue these commands:

```
kilcot# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
kilcot(config)# interface serial 1/0
kilcot(config-if)# nrzi-encoding
kilcot(config-if)# ^Z
kilcot#
```

- To remove **nrzi-encoding** from the serial interface, issue these commands:

```
kilcot# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
kilcot(config)# interface serial 1/0
kilcot(config-if)# no nrzi-encoding
kilcot(config-if)# ^Z
kilcot#
```

Second, once the serial line is operational, the most frequent problem is that the wrong SDLC address is set on the host (or on the router, depending on your perspective). The SDLC addresses on the host and on the router must be identical. If you receive this message, then the SDLC addresses do not match up and you need to change the address either on the host or on the router:

```
Received data from wrong address! Expect for output address C2/Got C4
```

The remainder of this document shows you how to use some Cisco **debug** commands to address these common problems. The format of command output may slightly differ from version to version, but the critical information about the frame type (Unnumbered Acknowledge [UA], Set Normal Response Mode [SNRM], and so forth) is consistent.

# Troubleshoot

Before you issue **debug** commands, if the SDLC interface is role primary, increase the SDLC poll-pause-timer to 500 ms. The default is 10 ms, which can overrun the debug output with polls.

```
sdhc poll-pause-timer 500
```

Also before you issue **debug** commands on the router, you must configure millisecond timestamp resolution for logged and debugged messages with these commands:

```
router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

router(config)# service timestamps debug datetime msec

router(config)# service timestamps log datetime msec

router(config)# ^Z

router#
```

To send log information to the buffer instead of to the console, issue these commands:

```
router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

router(config)# logging buffered buffer_size

!--- buffer_size depends on how much memory you have.
!--- If you have around 5 MB of free memory, then use 100000,
!--- which is 100 KB of buffer size.

router(config)# no logging console

router(config)# ^Z

router# clear log

Clear logging buffer [confirm] y

router#
```

**Note:** Refer to Important Information on Debug Commands before you use **debug** commands.

Use these **debug** commands to troubleshoot the router:

- **debug sdhc** Use to determine if you are establishing a good SDLC connection.
- **debug sdhc local-ack** Use to debug SDLC when you are using local acknowledgment.
- **debug stun packet** Use to debug serial tunnel (STUN) problems.
- **debug sdllc** Use to debug SDLC Logical Link Control (SDLLC) problems.

## debug sdhc

Refer to SDLC and SDLLC States for information about the states shown in **debug sdhc** command output.

The **debug sdhc** command output should appear similar to this:

```
SDLC: Sending RR at location 4
Serial3: SDLC O (12495952) C2 CONNECT (2) RR P/F 6
Serial3: SDLC I (12495964) [C2] CONNECT (2) RR P/F 0 (R) [VR: 6 VS: 0] rfp: P
```

The key item here is the SDLC address, which is C2 in this example.

## debug sdlc local-ack

The **debug sdlc local-ack** command output should appear similar to this:

```
SLACK (Serial3): Input      = Network, LinkupRequest
SLACK (Serial3): Old State = AwaitSdlcOpen New State = AwaitSdlcOpen
```

This output shows the input packets and the states that the local acknowledgement is handling.

## debug stun packet

The **debug stun packet** command output should appear similar to this:

```
STUN sdlc: 0:00:04 Serial3 NDI: (0C2/008) U: SNRM PF:1 <- XID1
STUN sdlc: 0:00:00 Serial3 SDI: (0C2/008) S: RR PF:1 NR:000 <- XID2
STUN sdlc: 0:00:00 Serial3 SDI: (0C2/008) S:I PF:1 NR:000 NS:000 <-XID3 Packet
```

In this example, the SDLC address is C2 and the modulo is 008. Modulo 8 is always supported, while modulo 128 is only supported with STUN Basic. NDI and SDI indicate Network Data Input and Serial Data Input, respectively.

## debug sdllc

Refer to SDLC and SDLLC States for information about the states shown in **debug sdllc** command output.

The **debug sdllc** command output should appear similar to this:

```
SDLLC: rx explorer rsp, da 4000.2000.1001, sa C000.1020.1000, rif 8840.0011.00A1.0050
```

This next line indicates that the router sent the exchange identification (XID) command (Format 0, Type 2) to the front-end processor (FEP):

```
SDLLC: tx long xid, sa 4000.2000.1001, da C000.1020.1000, rif 88C0.0011.00A1.0050,
      dsap 4 ssap 4
```

This last line is the Set Asynchronous Balanced Mode Extended (SABME) response to the XID command previously sent to the FEP by the router:

```
D>Rcvd SABME/LINKUP_REQ pak from TR host
```

---

## Related Information

- [Technology Support](#)
  - [Product Support](#)
  - [Technical Support & Documentation – Cisco Systems](#)
-

