**Flashing/Updating Linux kernel via U-Boot console**

V 1.0

Contents

[1 Introduction 3](#_Toc82782927)

[2 Tera Term Setup 3](#_Toc82782928)

[3 Section 1: UART 4](#_Toc82782929)

[3.1 Method 1: Y Modem Protocol - ✔️ 4](#_Toc82782930)

[3.1.1 Steps 4](#_Toc82782931)

[3.1.2 Observations 6](#_Toc82782932)

[3.2 Method 2: X Modem Protocol 6](#_Toc82782933)

[3.2.1 Steps 6](#_Toc82782934)

[3.2.2 Observations 6](#_Toc82782935)

[3.3 Method 3: Kermit Protocol 6](#_Toc82782936)

[3.3.1 Steps 6](#_Toc82782937)

[3.3.2 Observations 6](#_Toc82782938)

[4 Section 2: USB - ✔️ 7](#_Toc82782939)

[4.1 Steps 7](#_Toc82782940)

[4.2 Observations 7](#_Toc82782941)

[5 Section 3: Fastboot in UDP - ToDo 8](#_Toc82782942)

[6 Section 4: TFTP - ToDo 8](#_Toc82782943)

[7 References 9](#_Toc82782944)

# Introduction

To transfer file to U-Boot

This document briefly analyses the options available to transfer file to U-Boot from a PC. This document also lists some of the methods and the procedure to try those in the i.MX 6UL EVK. The objective is to **update kernel binary from U-Boot when the SDP is no longer available**.

The procedures mentioned here are verified in Windows PC because “Tera Term” has inbuilt support for most of these serial/network protocols.

From the initial analysis the fastboot can be enabled over UDP and U-Boot natively supports TFTP and UART (different protocols) for file transfer. This document will further be extended over time to cover all the possibilities.

# Tera Term Setup

To connect board for Access

1. Connect micro-USB cable between host PC and the micro-USB (labelled as “J1901”) console on the EVK. [7]
2. Identify the COM port on the host PC as “Device Manager > Ports (COM & LPT)” (you can remove and reconnect the USB cable to determine the COM port)
3. On your host PC, start “[Tera Term](https://osdn.net/projects/ttssh2/releases/)” with the following serial port settings (Setup > Serial Port… > Speed > New setting): (*ref table 1*)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Baud Rate** | **Data Bits** | **Stop Bits** | **Parity** | **Flow Control** |
| 115200 | 8 | 1 | none | none |

Table 1: Console setting for i.MX 6UL EVK

1. Enable serial console Logging (File > Log… > “Select location” > save) for later use. (*ref fig 2.1*)

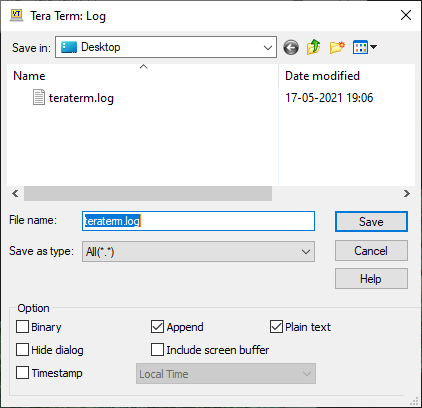


Figure 1: Tera term serial console log

1. Plug the power supply unit into an AC outlet and turn on the board.
2. Quicky press some key on the keyboard to interrupt the boot sequence and enter into U-Boot console.

# Section 1: UART

Methods to transfer file over UART

Here are the possible methods available (or achievable) on U-Boot for UART data transmission. This is to analyse the possible methods to flash/update Linux kernel on i.MX device over UART connection. [1], [2], [3]

1. Y Modem protocol
2. X Modem protocol
3. Kermit protocol

We’ll see these protocols and their procedures in details, as follows. But the quick summary can be viewed in the following table 2.

|  |  |  |  |
| --- | --- | --- | --- |
| File Size | | 7.49MB (*Linux kernel*) | |
| **S.No** | **Protocol** | **Time Taken** | **Speed** |
| 1 | YModem | 12 minutes | ~10kB/s |
| 2 | XModem | 14 minutes | ~9kB/s |
| 3 | Kermit | 1 hour 20 minutes | ~2kB/s |

Table 2: UART file transfer protocol comparison

## Method 1: Y Modem Protocol - ✔️

Transmission mode: YModem-1K

In YModem-1k, 124-byte information is used for fast transmission, and CRC teaching and research is used to ensure the accuracy of data transmission. [4]

### Steps

* 1. Access the U-Boot console, as directed in [section 2](#_TeraTerm_Setup)
  2. Delete Existing kernel with the following command:

fatrm mmc 1 zImage

* 1. Start YModem receiver command on U-Boot console. This will configure the UART port in receiver mode and the data received is temporarily stored in ${loadaddr}. It is user’s responsibility to write it in a file to store it permanently.

# load binary file over serial line (ymodem mode)

loady

* 1. Initiate Transmission in Tera Term (from PC). Go to “File -> Transfer -> YMODEM -> Send” (*ref fig 3.1*)

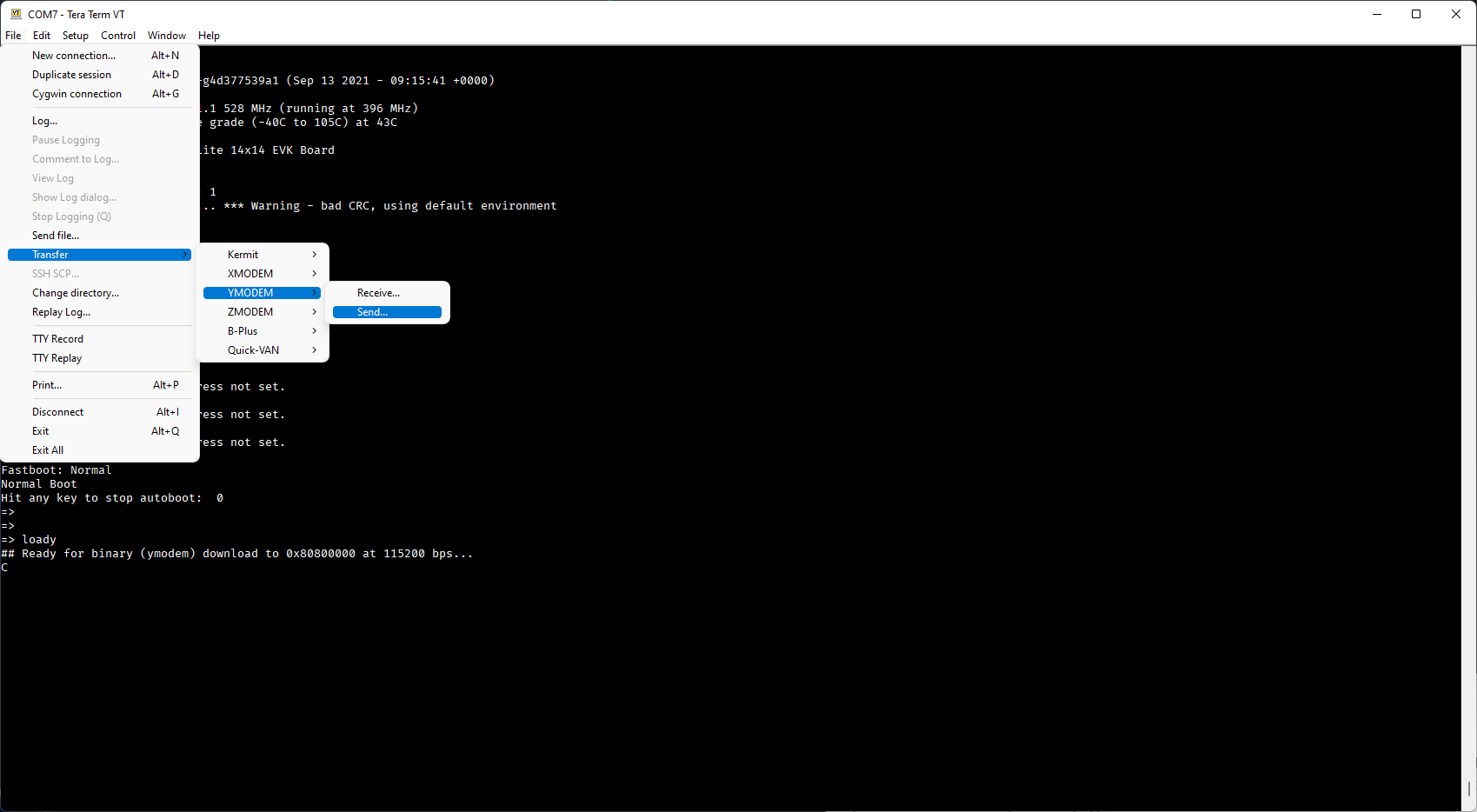


Figure 3.1: YMODEM transmission from Tera Term

* 1. Select the file to be transferred (Linux kernel, in our case) and wait for the transmission to be completed. (*ref fig 3.2*)

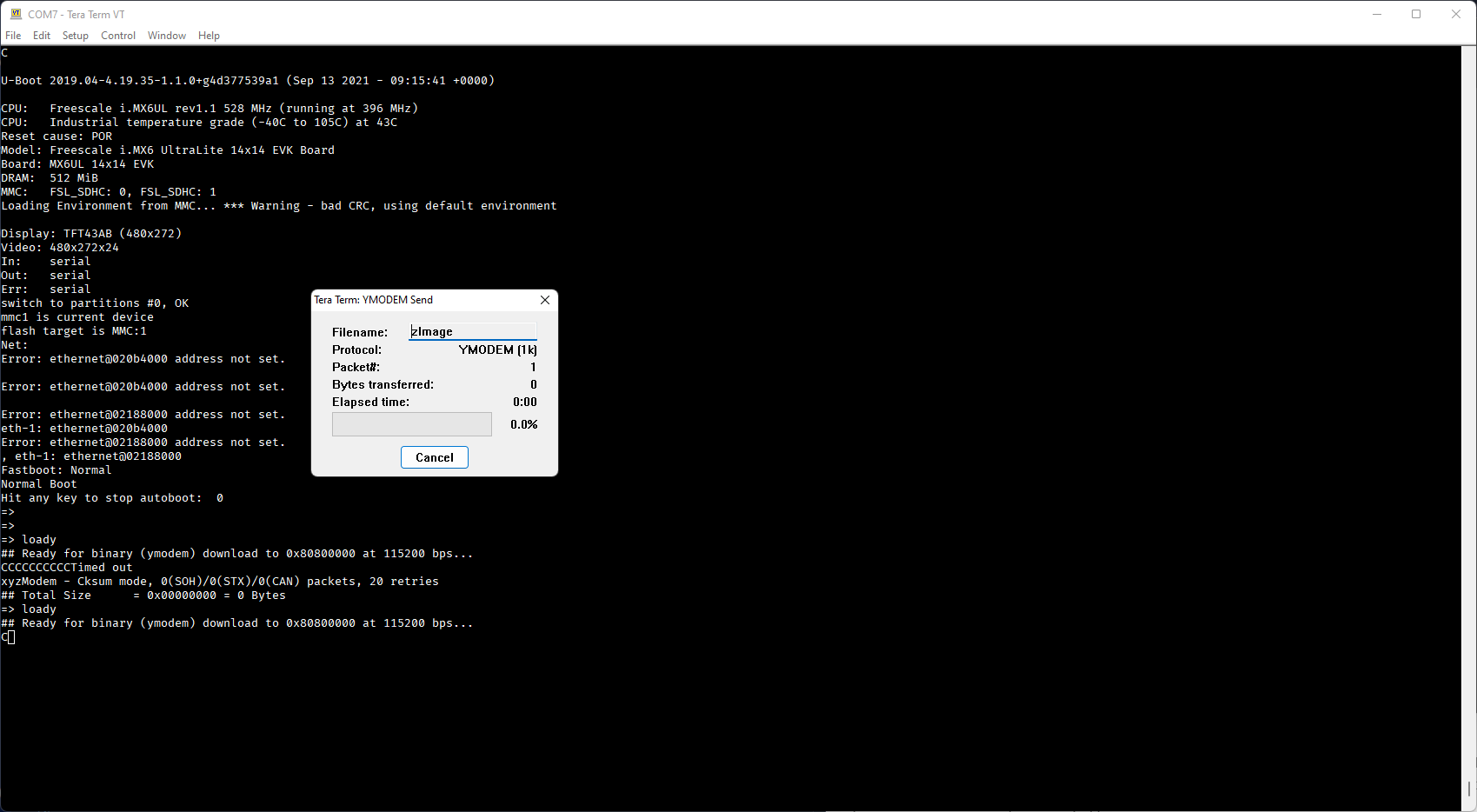


Figure 3.2: File transfer dialog in Tera Term

* 1. Once the transmission is finished, write the received kernel from memory to a file (zImage). The size of transferred bytes will be displayed in the U-Boot console after the transmission is finished, use this value as size. (*ref fig 3.3*)

# Write kernel image from memory to file

fatwrite mmc 1 ${loadaddr} zImage 0x77FF50

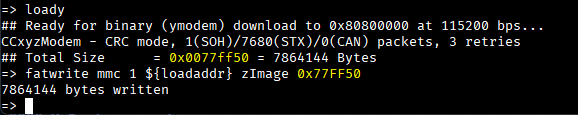


Figure 3.3: Write to file after the transmission completion

**NOTE**: For fatwrite use hexadecimal values for address and size

### Observations

1. This method is working
2. Transmission is very slow

## Method 2: X Modem Protocol

Transmission mode: XMODEM (CRC)

The Xmodem protocol [5] was created years ago as a simple means of having two computers talk to each other. With its half-duplex mode of operation, 128-byte packets, ACK/NACK responses and CRC data checking, the Xmodem protocol has found its way into many applications.

### Steps

The steps are quite [similar to the YModem](#_Steps), with differences in U-Boot receiver command and the Transmission protocol selection in Tera Term.

* 1. Start XModem receiver command on U-Boot console

# load binary file over serial line (xmodem mode)

loadx

* 1. Initiate Transmission in Tera Term (from PC). Go to “File -> Transfer -> XMODEM -> Send”

### Observations

1. This method is also working and slightly slower than YMODEM method
2. This transmission protocol is very slow

## Method 3: Kermit Protocol

The Kermit protocol [6] supports text and binary file transfers on both full-duplex and half-duplex 8-bit and 7-bit serial connections in a system and medium-independent fashion, and is implemented on hundreds of different computer and operating system platforms.

### Steps

The steps are quite [similar to the YModem](#_Steps), with differences in U-Boot receiver command and the Transmission protocol selection in Tera Term.

* 1. Start Kermit receiver command on U-Boot console

# load binary file over serial line (kermit mode)

loadb

* 1. Initiate Transmission in Tera Term (from PC). Go to “File -> Transfer -> Kermit -> Send”

### Observations

1. This method is also working and much slower than YMODEM & YMODEM method. It's unbearably slow and may not be feasible to consider as a viable solution
2. This transmission protocol is very very slow
3. Speed gradually decreases over transmission (started at ~6kB/s and ended in ~1kB/s)

# Section 2: USB - ✔️

This method uses a USB stick that contains the kernel file (zImage), when it’s inserted in the board the U-Boot will recognise it as a USB storage and assigns it a number. After this, the file can be loaded into memory with fatload command.

## Steps

The detailed procedure is given below.

1. Format the USB stick in FAT filesystem
2. Copy the kernel binary to the formatted USB stick
3. Login to U-Boot console (*steps given in* [*section 2*](#_Tera_Term_Setup))
4. Insert the USB to “J1101”
5. Start the USB and identify the number with the following commands:

# start (scan) USB controller

usb start

# show details of USB storage devices

usb storage

1. Load the kernel image in memory

fatload usb 0 ${loadaddr} zImage

1. Write the kernel image to a file (zImage)

fatwrite mmc 1 ${loadaddr} zImage 0x77FF50

The example flow is given in the following fig 4.1

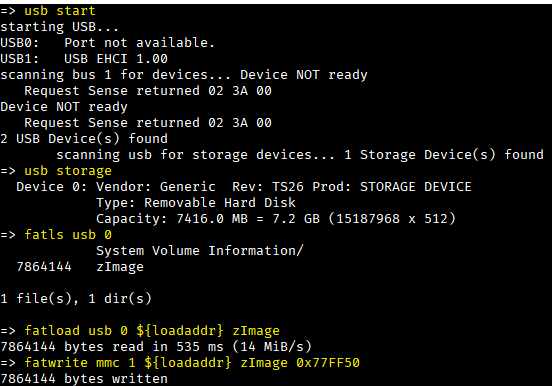


Figure 4.1: File transfer with USB on U-Boot

## Observations

1. It’s very fast and reliable method
2. USB storage support needs to be added in the U-Boot in build configuration

# Section 3: Fastboot in UDP - ToDo

To-Do

# Section 4: TFTP - ToDo

To-Do

# References

1. How to send file to embedded system with only serial console? - <https://stackoverflow.com/a/44689591/10773894>
2. How to transfer a file over serial console - <https://wiki.st.com/stm32mpu/wiki/How_to_transfer_a_file_over_serial_console>
3. Loading Linux Images over UART - <https://www.emcraft.com/stm32f429discovery/loading-linux-images-over-uart>
4. YModem protocol - <https://programmer.ink/think/ymodem-protocol-learning.html>
5. XModem Protocol with CRC - <http://web.mit.edu/6.115/www/amulet/xmodem.htm>
6. The Kermit Project - <https://www.kermitproject.org/kermit.html>
7. i.MX 6UltraLite Evaluation Kit Quick Start Guide - <https://media.digikey.com/pdf/Data%20Sheets/Freescale%20Semi/i.MX_6UL_EVK_QSG.pdf>