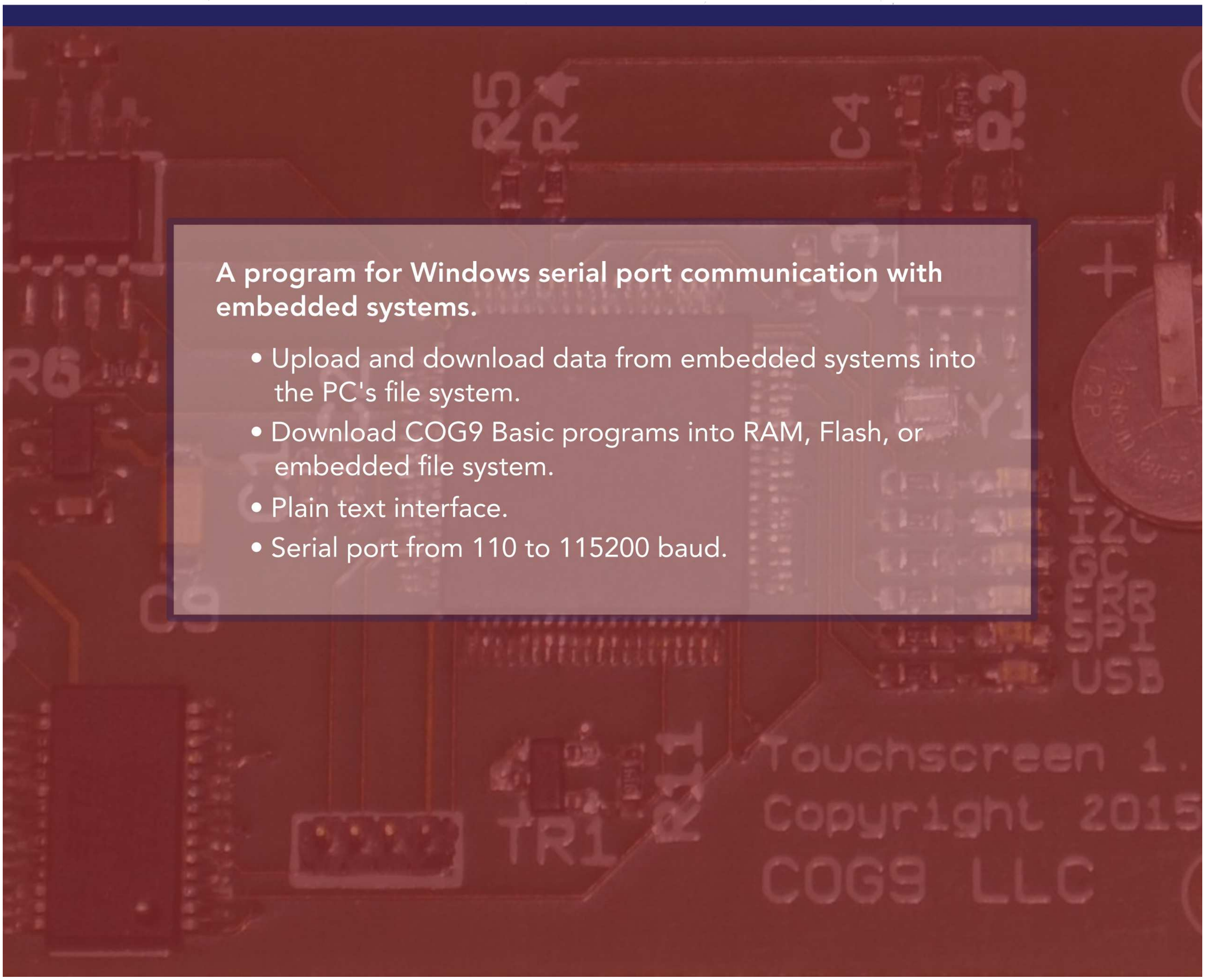
COG9 LLC

CTERM Serial Communications



A program for Windows serial port communication with embedded systems.

- Upload and download data from embedded systems into the PC's file system.
- Download COG9 Basic programs into RAM, Flash, or embedded file system.
- Plain text interface.
- Serial port from 110 to 115200 baud.

MN-02-2015, Revision 1, April 12, 2016, COG9 Enterprises LLC

Copyright ©2016 by COG9 Enterprises LLC. All rights reserved. This publication and its contents are proprietary to COG9 Enterprises LLC. No part of this publication may be reproduced in any form or by any means without the written permission of COG9 Enterprises LLC, 1858 East Bryan Avenue, Salt Lake City, 84108.

COG9 LLC has made every effort to ensure the correctness and completeness of the material in this document. COG9 LLC shall not be liable for errors contained herein. The information in this document is subject to change without notice. COG9 Enterprises LLC makes no warranty of any kind with regard to this material.

Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation.

Last Updated April 12, 2016.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A personal computer is required for operations with most embedded systems for long term storage, compilation, and communications with the internet. The `cterm` program provides the following facilities:

1. Download COG9 Basic programs and data files into the embedded system's mass storage or directly into Flash and RAM.
2. Communicate with COG9 Basic's (or any other program's) control program - things you type in a command window go to the embedded system and things it displays are echoed on the terminal.
3. Upload data from the embedded system into the PC's file system.

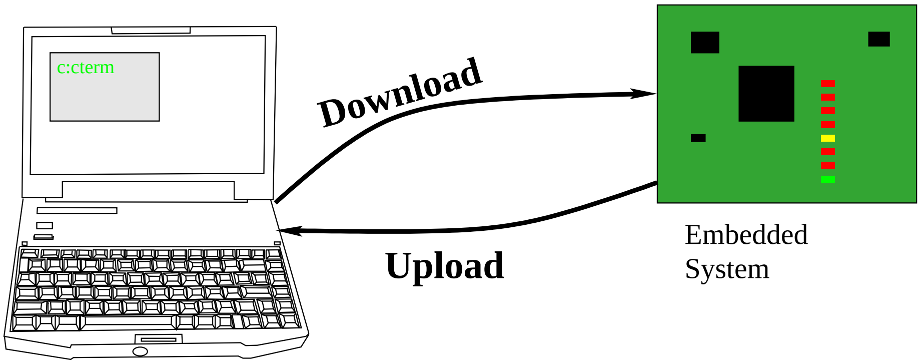


Figure 1.1: `cterm` program on a PC communicating with an embedded system.

1.1 Requirements

You need the following to run `cterm`:

1. A computer running Windows XP or newer.
2. The `cterm.exe` program.
3. A common place to put the `cterm.exe` program.
4. A way to talk to a serial port - **COM1** → **COM9**.
5. Something to talk to (could be another PC, embedded system, modem, ...).

1.2 Installation

The program is available as a Windows install script named `cterm_install.exe`. This installs the program in the 32 bit `Program Files` directory (this can change in newer systems) under the directory `cog9\bin`. An uninstall script is also included.

The program can be accessed by a button in the startup menu but this is not recommended as an initial error will cause the terminal to flash off. It is best that you start the program from the `cmd` window.

1.3 The License

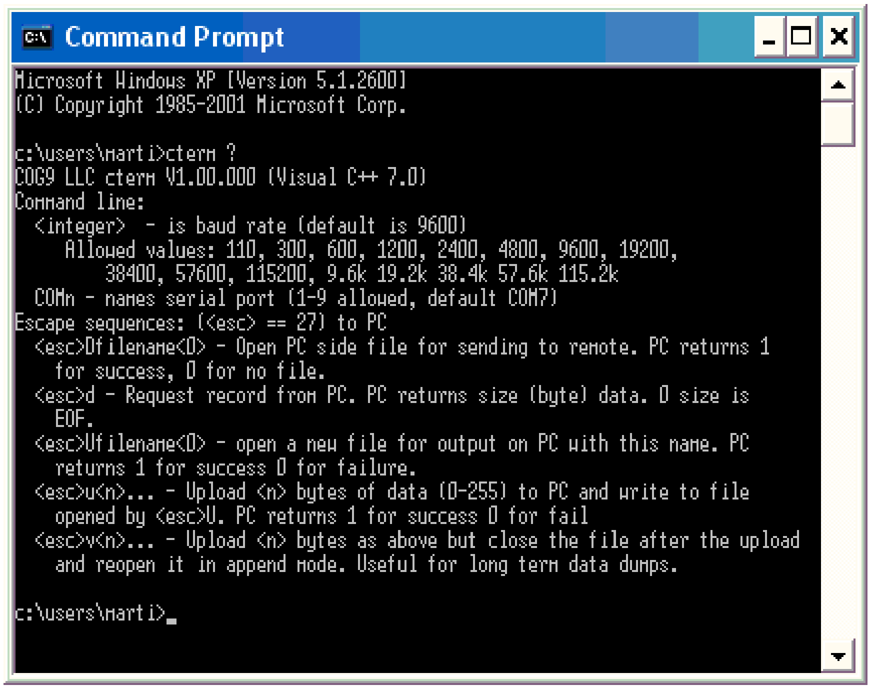
`cterm` is covered by the GNU Public license. This leaves you free to do pretty much whatever you want other than to sell it outright. There are no guarantees either.

Chapter 2

The Command Line Interface

`cterm` runs through the command line interface program. In Windows, this program is called `cmd.exe`.

If you've installed the program as directed and can execute it, you can retrieve all the current information by entering `cterm ?` in the command line window.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\users\marti>cterm ?
COG9 LLC cterm V1.00.000 (Visual C++ 7.0)
Command line:
  <integer>  - is baud rate (default is 9600)
     Allowed values: 110, 300, 600, 1200, 2400, 4800, 9600, 19200,
         38400, 57600, 115200, 9.6k 19.2k 38.4k 57.6k 115.2k
  COMn - names serial port (1-9 allowed, default COM7)
Escape sequences: (<esc> == 27) to PC
  <esc>Dfilename<0> - Open PC side file for sending to remote. PC returns 1
    for success, 0 for no file.
  <esc>d - Request record from PC. PC returns size (byte) data. 0 size is
    EOF.
  <esc>Ufilename<0> - open a new file for output on PC with this name. PC
    returns 1 for success 0 for failure.
  <esc>u<n>... - Upload <n> bytes of data (0-255) to PC and write to file
    opened by <esc>U. PC returns 1 for success 0 for fail
  <esc>v<n>... - Upload <n> bytes as above but close the file after the upload
    and reopen it in append mode. Useful for long term data dumps.

c:\users\marti>_
```

Figure 2.1: `cterm` run-time help.

All command line options can occur in any order. There are defaults for each.

2.1 COM port

The communications port has a precompiled default (currently **COM7** for obscure reasons) but any port from **COM1** to **COM9** is permissible.

A common problem is the use of USB serial communications adapters. Every new one you plug in is assigned a new communications port number and eventually you reach COM10 which the current version of our C compiler can't handle. The solution is to:

1. Activate the device manager.
2. Find the serial communications port list.
3. Locate the COM port you're having trouble with.
4. Examine its properties.
5. Go to the advanced section where the port number will be listed.
6. Select a different COM port number less than 10.
7. Ignore the system complaint and do it anyway.
8. Don't override a hardware UART communications port (usually COM1 and COM2). These have disappeared from most computers so it's unlikely you'll be bothered.

Microsoft likes to change this interface with every new release so you're on your own.

2.2 Baud Rate

Entering a number (in any order) modifies the baud rate from the default (usually 9600 baud). The following values and abbreviations are implemented:

Number	**Abbreviation**
110	
300	
600	
1200	
2400	
4800	
9600	9.6k
19200	19.2k
38400	38.4k
57600	57.6k
115200	115.2k

Table 2.1: Baud rate

2.3 Other Command Line Options

Other command line options can be interspersed at will.

Option	**Abbreviation**	**Description**
-noisy	-n	Generate messages about what's happening. All messages are prefixed by CTERM: to indicate where they're coming from.
-nonewline	–nnl	Do not generate a new line code (\n) when encountering a $< cr >$ character (code 13).
-newline	–nl	Do generate a new line code (\n) when receiving a $< cr >$ code (13) from the remote. This is the default.
-quiet	-q	Don't generate messages about what's happening.
-sendsize *nnn*	–ss *nnn*	Set the size of the download message to any value from 1 to 255. The default is 64.
-upload *filename*	–u *filename*	Open the file name specified in append mode to continue uploading from some previous effort. New uploaded data will be appended to this file.

Table 2.2: Other command line options.

Chapter 3

Escape Codes

Escape codes allow the remote connection to control the `cterm`'s operation. The remote sends an ASCII *esc* code (decimal 27) followed by a control character and perhaps additional information.

3.1 Open File to Download *esc* D *filename* 0

This sequence sets `cterm` to start downloading a file to the remote. The filename following the **D** must be present in the current directory that `cterm` is running in or must be the fully qualified name for somewhere else. The file name is terminated with a code 0 character. `cterm` will send a 1 in response if the file could be opened and 0 if not.

The file name may not have more than 254 characters. Only one file can be open for input at one time. The file is closed automatically when end of file is reached.

3.2 Request Buffer of Download File *esc* d

The remote sends this request when it's ready to receive a buffer from the open input file. `cterm` responds with a single byte indicating the number of characters to follow and then that number of characters. If this count is 0, end-of-file has been reached.

The remote must have previously opened a file for download and you must not send a read request after the file is closed.

The buffer size defaults to 64 bytes but you can use the **-sendsize** *nnn* or **–ss** *nn* command line options to set this as high as 255 or as low as 1 (very slow).

3.3 Change Send Buffer Size *esc*s*n*

This command lets the remote to temporarily change the send buffer size to the byte following. Sizes from 1 to 255 are permitted. Setting the size to small will cause very slow download transfers. A setting value of 0 will cause an error.

3.4 Open a File to Upload *esc* U *filename* 0

When the remote sends this sequence, `cterm` is ready to accept characters from the remote to store in a file. The *filename* can be fully qualified (up to 254 characters) or just the name of a file in the current directory. `cterm` responds with a 1 if the file was opened for output and 0 if this somehow or other failed. Uploading the file is then accomplished by sending *esc*u or *esc*v sequences.

Only one file can be open for output at a time.

3.5 Upload a Record *esc* u *nnn* . . .

This escape sequence sends a string of bytes from 1 to 255 to `cterm` for storage in the open upload file. The 'u' is followed by a single byte indicating the number of bytes that follow (1 - 255). If this count is 0, the upload file is closed permanently. When the write has completed, `cterm` sends a 1 in response.

The upload file must be open for this to function.

3.6 Upload Append Mode *esc* v *nnn* . . .

This escape sequence also sends a string of bytes from 1 to 255 to `cterm` for storage in the open upload file. The 'v' is followed by a single byte indicating the number of bytes that follow (1 - 255). If this count is 0, the upload file is closed permanently.

After the record is written, the upload file is closed and then reopened in append mode. Consequently, if the `cterm` host crashes or is turned off, the complete file to this point will be available. When this activity has completed, `cterm` sends a 1 to the embedded system or 0 if the activity failed (disk full, etc.).

The upload file must be open for this to function. 'u' and 'v' escape sequences can be mixed on the same upload file.