ToughDev

We Love Technical Sharing!

HOME SOFTWARE DEVELOPMENT HARDWARE VINTAGE COMPUTING TERMS OF USE

.NET Framework Hardware Oscilloscope Software Development Useful Projects Vintage Computing

Programming the Tektronix TDS 340 100MHz digital storage oscilloscope

January 14, 2014 ToughDev 8 Comments

★★★★★

5.00 avg. rating (**95**% score) - **2** votes

In my previous article I provided some information on the Tektronix TDS 340 100 MHz digital storage oscilloscope and instructions on how to install the Option 14 card to get VGA output and support for hard copy. This article will provide some further information on interfacing the oscilloscope with a computer using the RS-232 port to retrieve raw signal data and share some of my interesting findings.

Using the serial interface

The oscilloscope serial port must first be configured inside the **Utility**>**System I/O** menu and a cross-over serial cable is needed to connect the oscilloscope to the PC. It is best to turn off both hardware and software handshaking as they are not going to help much and will just cause problems. The recommended serial settings are 19200bps, 8 data bits, no parity, 1 stop bit and carriage-return (CR) line ending.

To check if the connection is working, use a terminal software such as Tera Term Web 3.1, configure it to echo characters locally with CR line ending, and type **ID?** followed by the ENTER key to ask the oscilloscope to return its identifier string, which should look like:

```
TEK/TDS340,CF:91.1CT,FV:v1.00
```

Commands are case-insensitive and multiple commands or responses are separated by a line ending character (CR, LF or CR/LF as configured). Commands ending with a question mark (?) are queries and a response from the oscilloscope is to be expected. Commands not ending with a question mark will simply be executed by the oscilloscope with no value returned. To check if there are any errors during command execution, use the following:

```
*ESR?   // Show the value of the Event Status Register
ALLE?   // Return the error log, which will show any errors with the commands
previously sent
```

The following command set, extracted from the programmer manual, measures the frequency of the input signal at channel 1:

```
MEASU:IMM:SOURCE CH1   // Set measurement source to Channel 1
MEASU:IMM:TYPE FREQ    // Measure the signal frequency
MEASU:IMM:VAL?         // Get the measurement value
```

If the probe of channel 1 is connected to the 1kHz calibration point in front of the oscilloscope, the above command set would return a value of approximately 1.0000E3, indicating a frequency of 1000Hz.

Retrieving raw waveform data

Command CURV? is used to ask the oscilloscope for the raw measurement data of the waveform being displayed. The following code will return all 1000 data points in the oscilloscope data acquisition memory:

```
DAT:SOU CH1     // Measurement source to Channel 1
```

Search

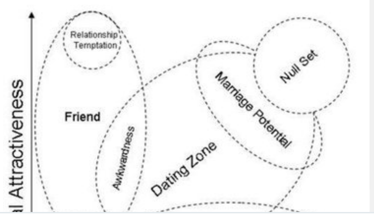
ToughDev - We L
1,502 likes

Like Page

Tweets by @mrtoughdev

Tough Developer
@mrtoughdev

Why do some men just friend-zone the women they love?



Follow @mrtoughdev

Recent Comments

Ken on Calibration and acquisition problems on Tektronix TDS 340 oscilloscope

```
DAT:ENC ASCI      // ASCII format
DAT:WID 2         // 2 bytes data width
DAT:STAR 1        // first data point
DAT:STOP 1000     // last data point (1000th)
CURV?             // get waveform data
```

The response will be a set of comma separated integers:

```
22784,23040,-6656,[….],-6656,23040
```

The range of each returned integer value is –128 to 127 when DAT:WID is 1. Zero is center screen. The range is –32768 to 32767 when DAT:WID is 2. The upper limit is one division above the top of the screen and the lower limit is one division below the bottom of the screen.

To properly interpret the data, it will be useful to know the oscilloscope settings via the WFMPR? command, which will return the following:

`2;16;ASC;RP;MSB;"Ch1, DC coupling,` **`2.0E0 V/div, 5.0E-4 s/div,`** `1000 points, Sample mode";1000;Y;"s";`**`1.0E-5`**`;500;"Volts";3.125E-4;1.28E4;0.0E0`

Among the returned values, the voltage per division, the time per division and sampling rate parameters (highlighted) will be needed to accurately analyze the returned data.

Taking a screenshot of the oscilloscope

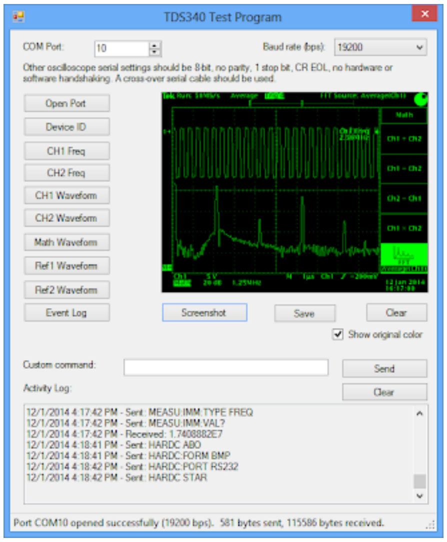
A screenshot of the oscilloscope display can be captured progammatically using the following commands:

```
HARDC ABO           // abort any existing hard copy
HARDC:FORM BMP      // set hard copy format to bitmap
HARDC:PORT RS232    // hard copy port to RS232
HARDC STAR          // start hard copy
```

The image data will be sent via the serial link. Unlike other commands, there is no end of file marker for the HARDC STAR command. To identify when to stop receiving data programmatically, one way is to count the number of bytes received and compare with the expected value. As the file size could vary depending on the hard copy output settings, an easier way is to assume that the hard copy operation has ended if no data is received after a certain period, e.g. 2 seconds.

Custom PC interface software

With some free time I made a .NET application that allows the user to retrieve the frequency measurements, waveform data as well as taking a screenshot of the oscilloscope:



COM Port: 10

Baud rate (bps): 19200

Other oscilloscope serial settings should be 8-bit, no parity, 1 stop bit, CR EOL, no hardware or software handshaking. A cross-over serial cable should be used.

☑ Show original color

Custom command:

Activity Log:

```
12/1/2014 4:17:42 PM - Sent: MEASU:IMM:TYPE FREQ
12/1/2014 4:17:42 PM - Sent: MEASU:IMM:VAL?
12/1/2014 4:17:42 PM - Received: 1.7408882E7
12/1/2014 4:18:41 PM - Sent: HARDC ABO
12/1/2014 4:18:41 PM - Sent: HARDC:FORM BMP
12/1/2014 4:18:42 PM - Sent: HARDC:PORT RS232
12/1/2014 4:18:42 PM - Sent: HARDC STAR
```

Port COM10 opened successfully (19200 bps). 581 bytes sent, 115586 bytes received.

To use the application, first configure the serial port settings (port number, baud rate) and click on **Open Port** to initialize the serial interface. The **Activity Log** text box shows the commands sent and responses received. The **Screenshot** button will request for a screenshot from the oscilloscope and show it in the application. If option **Show original color** is checked, the application will convert the black-on-white image

Briog on eBox 3350MX x86 Compact PC

ToughDev on Tweaking the AlphaSmart Neo, a great portable word processor with 700-hour battery life

b. on Tweaking the AlphaSmart Neo, a great portable word processor with 700-hour battery life

Frank on English text to speech on a PIC microcontroller

Categories

.NET Compact Framework
.NET Framework
3CX
Android
ASP/ASP.NET
Blackberry
Blogger
Bluemix
C#
Database
DirectShow
FreePBX
General Programming
GPS
Hardware
Hoiio
I2C
iOS
LCD Modules
Linux
Logic Analyzer
Mac OS
Microcontroller
Microsoft Office
Microsoft Virtual PC
Microsoft Visual Studio
Mobile Development
Networking
Objective-C
Office OneNote
Oscilloscope
PDF
PHP
Research
Reverse-engineer
Scriptcase
SMS Migration

returned by the TDS340 to green-on-black, to make it look more like a real screenshot.The **Event Log** button will show all error messages currently in the oscilloscope event log.

The application is using the SerialPort component of the .NET framework. It assumes that both hardware and software handshaking is disabled in the oscilloscope serial settings. Interestingly, even with handshaking disabled, the DTR (Data Terminal Ready) and RTS (Ready To Send) lines must be on, otherwise the oscilloscope will not respond to the data sent. This is done using the following C# code:

```
SerialPort1.Handshake = false;
SerialPort1.DtrEnable = true;
SerialPort1.RtsEnable = true;
```

Due to the asynchronous nature of the DataReceived event of the .NET SerialPort component and the limited time that I have, I did not attempt to make the application wait for all data to be received before enabling the action buttons. For this reason, you will need to wait for a while and check the activity log after pressing any button to make sure that the command has finished executing before peforming the next action, otherwise the application behavior may be unexpected.

Download the PC application here

The Visual Studio 2012 source code is included in the download package. Microsoft .NET Framework 2.0 (which is installed by default on Windows 7 or later) is required. The executable can be found in the **bin** folder. You can also find on GitHub the application source code and executable. The GitHub repository also contains some minor user interface improvements made by joesphan that are not available in the original source code.

Downloads for TDS 340A, TDS 360 & TDS 380 oscilloscopes:

User Manual
Technical Reference
Programmer Manual
Service Manual

See also:
Exploring Tektronix TDS 340 100MHz digital storage oscilloscope
Calibration and acquisition problems on Tektronix TDS 340 oscilloscope

Share Tweet Share Share Share Share

★★★★★
5.00 avg. rating (**95%** score) - **2** votes

← Exploring Tektronix TDS 340 100MHz digital storage oscilloscope

Calibration and acquisition problems on Tektronix TDS 340 oscilloscope →

8 thoughts on “Programming the Tektronix TDS 340 100MHz digital storage oscilloscope”

STB448
November 3, 2016 at 2:05 am
Permalink

I was enthusiast for your very good article.
I’ve followed your article connecting my TDS340A to my laptop.
After open port command I’ve found that no buttons (commands) sent to the instrument have an answer

Software Development
Speech
SPI
Telerik
Tips & Tricks
USB to Serial
Useful Projects
VB.NET
Vintage Computing
VmWare
VoIP
Web Development
Web Hosting
Wikipedia
Win32 API
Windows Embedded
Windows Mobile
Windows Phone 7
Windows Tips
Xcode

Tools

Log in
Entries RSS
Comments RSS
WordPress.org

from it.
If I push “hardcopy” button onto the instrument I see incrementing bytes received up a certain point in which it stops with an error “parameter not valid”.
Can you help me to troubleshoot this?

ToughDev Post author
November 8, 2016 at 4:44 pm
Permalink

Hi,

Sorry for the late reply.

Are you using a USB to serial adapter? Make sure that your USB to serial driver is correctly installed and it can send/receive data from serial equipments properly. Being able to send but not to receive data (and vice versa) is a common problem when using USB to serial adapters.

Also check the serial settings on both the oscilloscope and your laptop and make sure they match. Try to select a lower baudrate e.g. 9600bps and disable hardware / software handshaking. If everything is correct it should work.

Let me know how it goes.

Vyacheslav
February 5, 2017 at 1:05 am
Permalink

Hello! Thank you for your work! You helped me a lot to deal with some problems. I bought myself the same oscilloscope with a small defect (menu is not displayed Components) first soldered adapter with DB9 to 15-pin on general schematic (without referring to the manual) and unfortunately Screen does not appear on the monitor (because of SCL SDA) problem was solved with the help of guides programmer, sent a team of bright, then the idea to write a programa that will perform terminal commands by pressing a simple button, but you have already made this!
Once again many thanks for the interesting informative articles !!!

ToughDev Post author
February 7, 2017 at 11:03 am
Permalink

Hi Vyacheslav,

Glad you find the article helpful. Let me know if you need any further info 🙂

Vyacheslav
February 9, 2017 at 5:49 am
Permalink

You have not experienced such a device:
http://www.ebay.com/itm/New-3-5-USB-1-44MB-SSD-Floppy-Drive-Emulator-For-Embroidery-Machine-/160767259603
FDD interface is very simple I think should work!
I ask purely out of curiosity, It may not come up

ToughDev Post author
February 10, 2017 at 8:02 pm
Permalink

I played with a lot of floppy emulators, mostly using SD cards / USB drives to emulate 3.5″ PC floppy drives. Your link points to one of these models. The one I am currently playing with is FloppyEmu http://www.bigmessowires.com/floppy-emu/ which emulates Apple II / early Macintosh floppy drives. I bought a few weeks ago but it's still in the packaging. When I have some time, probably next week, I will take a look and write another article about it 🙂

STB448
November 6, 2020 at 2:29 am
Permalink

Hi! I'm back to you after ages, because finally I've found a solution to the annoying problem to RS-232 dialog errors. Hope this can help other people.
I've waste a lot of time trying a solution in order to grab images from TDS 340A.
GPIB seems a good solution, but it cost money (more than 500 € for a USB/GPIB converter)
Is possible to change floppy drive with a floppy emulator (using SD cards), but to buy a good working product you have to choose a one customized for TDS 300 oscilloscopes (i.e.: FlexiDrive S-USB Slim, DATEX FloppyFlash Emulator, etc.); setup and wiring is anyway a tricky and painful thing.
RS-232 Dialog therefore remain the most simple solution; I've tried several software experiencing some random dialog problems. Finally this is what I've found:
1. Few softwares can handle TDS 340A, TEKTRONIX invite to download WaveStar 3.0 telling that it works on this instrument also, but is not true. I've found an old version (2.5) that have TDS 340A in its internal database; this version works perfectly and you can also drive the scope remotely…but it's hard to setup RS-232 dialog. (works only with 9600 bps, EOL = LF). Despite to TEKTRONIX indications, hardware or Xon/Xoff control don't work, so you have to set NONE for both.
2. RS-232 cable wiring indicated on WaveStar User Manual works only if plugged inverted (PC side on scope and scope side to PC). With RealTerm I've found this cable generate several frame errors. For WaveStar and TDS340_NET applications I'v found working perfectly without any error a cable wired TX/RX crossed, DCD jumped w. DTR (scope side), CTS jumped w. RTS (PC side), DCD+DSR+DTR jumped (PC side)
3. Any time TDS 340A is powered on, is needed to go to RS-232 setup changing Hardware Flagging to ON and later OFF. Otherwise data stream remain stopped therefore RS-232 dialog doesn't work at all.

ToughDev Post author
November 6, 2020 at 6:59 pm
Permalink

Thanks a lot for sharing your findings. 🙂 I usually set handshaking to OFF so that I don't have to worry about CTS/DTR/etc. Good to know that WaveStar 2.5 works with the TDS340A – I never had the chance to play with WaveStar. I did not bother with the USB/SD floppy emulator, too much trouble to find the right unit/firmware. I have since developed my own solution using a PIC microcontroller that listens to the serial port and write the output to the SD card. The unit is plugged to the back of the scope and good enough for screen capturing. Nowadays my main scope is the Rigol DS1054Z but somehow I still prefer the TDS despite its limited memory length. The user interface is responsive and doesn't lag at times like the Rigol. Old electronics are reliable and built to last. My Rigol rotary encoder failed after around 3 years and needs to be replaced. 🙂

Leave a Reply

Your email address will not be published. Required fields are marked *

Name *

Email *

Comment

□ I'm not a robot

reCAPTCHA
Privacy - Terms

Post Comment

□ Notify me of follow-up comments by email.

□ Notify me of new posts by email.