Programming Example: Retrieve data from an XE series Oscilloscope using Kotlin

February 26, 2019

The SDS series of oscilloscopes all feature remote programming and data collection capabilities. They can be integrated easily into many automated test environments to ease the setup and data acquisition during testing.

One of our helpful customers developed a nice programming example designed to set up and retrieve data from a SIGLENT SDS1202X-E Oscilloscope using Kotlin, a free open source coding environment (more on Kotlin here).

The code utilizes a LAN connection and open sockets.

Thanks to Chris Welty for the code!

Here is a text file of the example:

SDSDataRetrievalKotlinExample

```
/**
* License: 3-Clause BSD
*
* Copyright 2018 Chris Welty
*
* Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
*
* 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
*
* 3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING,
* BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
* EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
* PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
```

```
IN CONTRACT, STRICT LIABILITY,
* OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE
* POSSIBILITY OF SUCH DAMAGE.
*/

package scope

import java.io.BufferedWriter
import java.io.OutputStreamWriter
import java.io.Serializable
import java.net.Socket

/**
* Contains a single waveform downloaded from a Siglent 1202X-E
*/
class Waveform(val vDiv: Double, val vOffset: Double, val tDiv: Double, val tOffset: Double, val data:
ByteArray) : Serializable {

val xs: DoubleArray
get() = DoubleArray(data.size, { i -> i * tDiv * 14 / data.size + tOffset – tDiv * 7 })

val ys: DoubleArray
get() = DoubleArray(data.size, { i -> data[i] * vDiv / 25 – vOffset })

companion object {
/**
* Download the waveform displayed on the scope’s screen
*/
fun download(): Waveform {
Socket(“192.168.1.222”, 5025).use { socket ->

println(“connected to ” + socket.inetAddress)
val output = BufferedWriter(OutputStreamWriter(socket.getOutputStream(), Charsets.US_ASCII))

// since the socket can return binary data, we can’t use an InputStreamReader to
// translate the bytes to characters. We’ll have to do it ourselves.
// SCPI generally uses US ASCII, shouldn’t be too hard.
val input = socket.getInputStream()

/**
* Read from the scope until \n is encountered.
* The bytes are translated to characters numerically (so US_ASCII).
*/
fun readLine(): String {
val sb = StringBuilder()
while (true) {
val c = input.read()
when (c) {
```

```
-1, '\n'.toInt() -> return sb.toString()
else -> sb.append(c.toChar())
}
}
}

/**
* Read a number of bytes from the scope.
*
* The bytes are not translated into characters.
*/
fun readBytes(n: Int): ByteArray {
val result = ByteArray(n)
var i = 0
while (i < n) {
i += input.read(result, i, n – i)
}
return result
}

fun writeLine(string: String) {
output.write(string)
output.write("\n")
output.flush()
}

/**
* Read a numerical response from the scope.
*
* The scope returns responses like "C1:VDIV 1.00E+00V".
* This function extracts the "1.00E+00", converts it to a double, and returns it.
*/
fun readNumber() = readLine().split(" ")[1].dropLast(1).toDouble()

writeLine("*IDN?")
println(readLine())

// reset the scope response format to its default so readNumber() works
writeLine("CHDR SHORT")

writeLine("C1:VDIV?")
val vDiv = readNumber()

writeLine("C1:OFST?")
val vOffset = readNumber()

writeLine("TDIV?")
val tDiv = readNumber()
```

```
writeLine("TRDL?")
val tOffset = readNumber()

// request all points for the waveform
writeLine("WFSU SP,0,NP,0,F,0")
writeLine("C1:WF? DAT2")

// parse waveform response
val header = String(readBytes(21))
println("header is $header")
val length = header.substring(13, 21).toInt()
println("length is $length")
val data = readBytes(length)
readBytes(2) // 2 garbage bytes at end

println("V/div = $vDiv; offset = $vOffset; t/div = $tDiv; tOffset = $tOffset")

return Waveform(vDiv, vOffset, tDiv, tOffset, data)
}
}
}
}
```

North American Headquarters

SIGLENT Technologies America, Inc
6557 Cochran Rd Solon, Ohio 44139
Tel: 440-398-5800
Toll Free:877-515-5551
Fax: 440-399-1211
info@siglent.com
www.siglentamerica.com/

European Sales Offices

SIGLENT TECHNOLOGIES EUROPE GmbH
Staetzlinger Str. 70
86165 Augsburg, Germany
Tel: +49(0)-821-666 0 111 0
Fax: +49(0)-821-666 0 111 22
info-eu@siglent.com
www.siglenteu.com

Asian Headquarters

SIGLENT TECHNOLOGIES CO., LTD.
Blog No.4 & No.5, Antongda Industrial Zone,
3rd Liuxian Road, Bao'an District,
Shenzhen, 518101, China.
Tel:+ 86 755 3661 5186
Fax:+ 86 755 3359 1582
sales@siglent.com
www.siglent.com/ens