TELEDYNE TEST TOOLS
Everywhereyoulook™

Operator’s Manual

RF-Multiplexer T3SP-D4MX

T3 stands for Teledyne Test Tools.

Contents

1. General Information about the T3SP-D4MX

Please read this operation manual carefully before commissioning the T3SP-D4MX , to avoid incorrect operation of the instrument.

The T3SP-D4MX multiplexer unit can be used in applications requiring a high frequency (RF) bandwidth in combination with pair-wise phase-matched signal paths. The multiplexer was developed for test & measurement environments as well as for industrial applications, by using RF micro-electro-mechanical-systems switches (MEMS).

A high level of protection against electromagnetic discharge (ESD) is given by internal protection diodes.

A main advantage of using MEMS technology is a very long lifetime, which goes hand in hand with a high number of cycle times (up to 1 billion). The device typically is used in multi signal path testing applications (e.g. cable testing). It can be programmed either by a stand-alone applications or integrated into automated test environments.

Teledyne Test Tools offers TDR systems for measuring impedance profiles and S-parameters. The Teledyne T3SP-10D/15D series is operated by using the universal instrumentation software *Seunis* that provides control of the T3SP-D4MX as well as calibration (open, short, load, thru) and/or de-embedding of the multiplexer unit.

2. Technical Specifications

2.1 Key Features

- Double pole 4 throw (DP4T) switch with additional OFF state
- RF bandwidth up to 10 GHz (3dB bandwidth: 8 GHz)
- Phase-matched differential signal paths (typ. <5 ps skew)
- Up to 1 billion switching cycles (cold) / 500 million (hot, 10dBm@50 Ω)
- Starts in *not connected* state
- Internal ESD protection
- SMA connectors
- Controlled and powered via USB

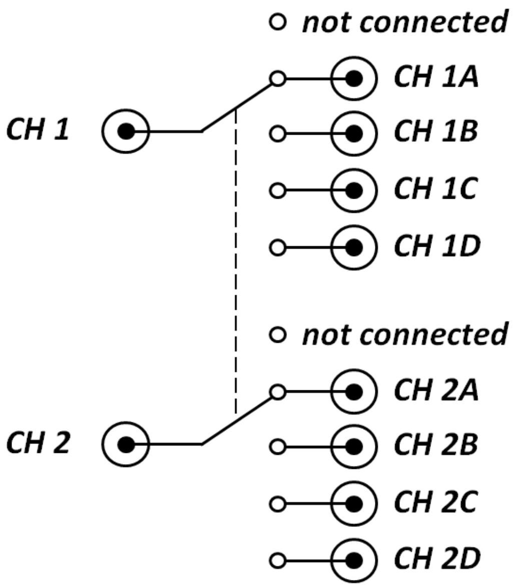


Figure 1: Simplified functional diagram showing input ports CH1 and CH2 to the left and the blocks of four output ports for each input port. Alternatively, the input can be set to an unconnected state.

2.2 Electrical Specifications

General Characteristics

Parameter	Value	Remark/Condition
Operating frequency	0 - 10 GHz	3 dB bandwidth: 8 GHz
RF max. power ratings	30 dBm 24 dBm	50 Ω load open
RF input voltage range	±6 V	DC
RF input current range	±200 mA	DC
Contact resistance (ON)	3.5 Ω 1.8 Ω	max. typ.
Contact resistance (OFF)	>1 GΩ	@ ±6 V
Switching time	<100µs <10 ms	without software latency with software latency
Power supply voltage	5 V	via USB
Power supply current	<12 mA	
Electrostatic discharge	5 kV*	RF_{in}, RF_{out}
Switch life time+	10^9 $500 \cdot 10^6$	cold switching hot switching, 10dBm@50 Ω

* *Human body model (HBM)*

+ *Caution: hot switching reduces switch life time*

RF Characteristics (single-ended)

Parameter	min.	typ.	max.	Frequency
Insertion loss		< 0.4 dB	0.5 dB	DC – 0.3 GHz
		< 1.0 dB	1.2 dB	0.3 GHz – 2 GHz
		< 3.0 dB	3.5 dB	2 GHz – 8 GHz
		< 5.0 dB	6.0 dB	8 GHz – 10 GHz
Return loss ($CH_{1,2}$ input)	20 dB	> 25 dB		DC – 0.3 GHz
	16 dB	> 18 dB		0.3 GHz – 2 GHz
	13 dB	> 15 dB		2 GHz – 8 GHz
	8 dB	> 10 dB		8 GHz – 10 GHz
Isolation between contacts of same channel	40 dB	> 45 dB		DC – 0.3 GHz
	30 dB	> 32 dB		0.3 GHz – 2 GHz
	25 dB	> 27 dB		2 GHz – 10 GHz

RF Characteristics (differential, between same paths of CH1 & CH2)

Parameter	min.	typ.	max.	Frequency
Isolation between CH1 & CH2	55 dB	>60 dB		DC – 10 GHz
Group delay deviation between CH1 & CH2		±5 ps	±10 ps	DC – 10 GHz

2.3 Mechanical Specifications

- SMA in-/output connectors
- USB mini-B connector (programming & power)
- Ground jack (4mm)
- Dimensions: 49x108x180mm^3 (with connectors & feet)
  42x108x165mm^3 (without connectors & feet)

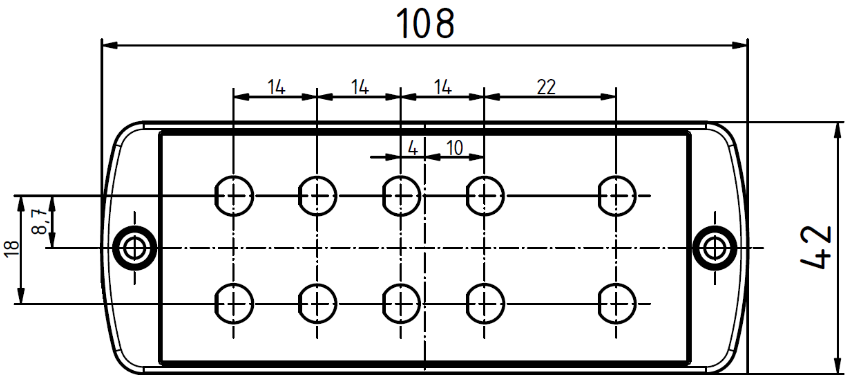


Figure 2: Front panel dimensions

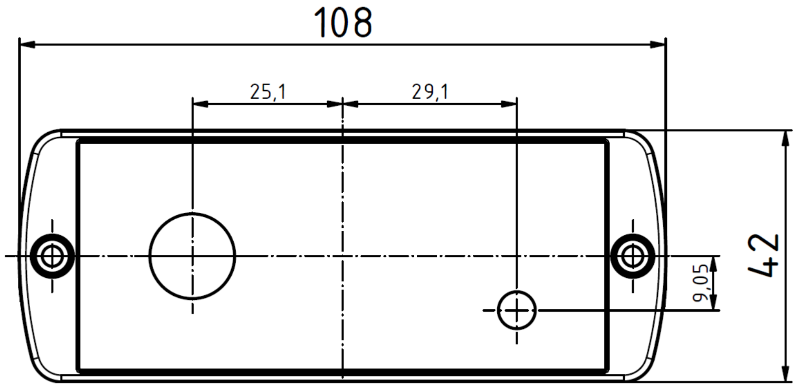
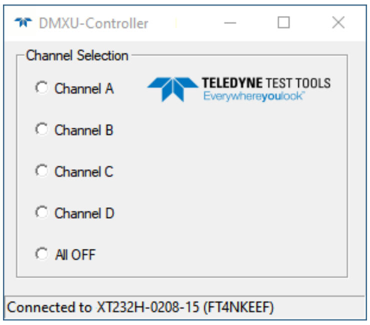


Figure 3: Rear panel dimensions

3. Programming

To have a quick access to the T3SP-D4MX and to be able to perform tests, the Windows application `WinD4MX.exe` can be downloaded from the Teledyne LeCroy website. The program does not need to be installed and does not require any additional driver components.



Connected to XT232H-0208-15 (FT4NKEEF)

Figure 4: Control application `WinD4MX.exe`

To integrate the device into your own software environment for remote control, please read the following sections.

3.1 USB Interface Description

The T3SP-D4MX is controlled via a USB 2.0 interface based on a communication chip from Future Technology Devices International Limited (FTDI). Windows operating systems usually have the required D2XX-driver for this chip family pre-installed. If this is exceptionally not the case, the corresponding driver can be downloaded from the FDTI driver download page.

The device is not controlled via a virtual comport protocol, therefore it is necessary to control it via the `cDMXU.dll` 32-bit shared library shipped with the device. In order to simplify the device connection, some programming examples are listed in the following section. In case a 64-bit shared library is required, please contact our Teledyne LeCroy support.

Please use only the supplied USB cable to avoid transmission errors.

3.2 Programming Examples

3.2.1 C/C^{++}-Code

The T3SP-D4MX control library `cDMXU.dll` is written in C^{++}. All function declarations / signatures can be found in the corresponding header file:

All functions return a positive value on success and `-1` on error. For more detailed information on the individual functions, please refer to the C source code below:

```
int cDMXU_Init(char* id);
int cDMXU_IsInitialized(void);
int cDMXU_SetChannel(int channel);
int cDMXU_GetChannel(void);
int cDMXU_GetSerialNumber(void);
int cDMXU_GetNumberOfChannels(void);
int cDMXU_Release(void);
```

The following is a simple main program that calls all functions as an example:

```
// make sure cDMXU.lib is added to linker dependencies
#include <stdio.h> // include header file
#include "cDMXU.h" // include T3SP-D4MX header

// macro for checking on errors
#define CHK_ERROR(e,m) {if (e<0) {printf(m); return e; } }

int main(void)
{
  int no_of_channels = -1;  // number of channels of multiplexer
  int channel = -1;         // channel variable
  int serno = -1;           // serial number (50001 --> 05.0001)

  // initialize library (id string not required yet)
  if (cDMXU_Init(NULL) != 0)
    CHK_ERROR(-1, "Initialization failed");
  // check if initialized
  if (cDMXU_IsInitialized() < 0)
    CHK_ERROR(-1, "Library not initialized");

  // read serial number as integer
  serno = cDMXU_GetSerialNumber();
  CHK_ERROR(serno, "Reading serial number failed ");

  // read number of available channels
  no_of_channels = cDMXU_GetNumberOfChannels();
  CHK_ERROR(no_of_channels, "Getting number of channels failed ");

  // 1-based numbers (0: ALL OFF)
  if (cDMXU_SetChannel(1) < 0)
    CHK_ERROR(-1, "Setting channel failed ");

  // 1-based numbers (0: ALL OFF)
  channel = cDMXU_GetChannel();
  CHK_ERROR(channel, "Getting channel failed ");

  // release library
  if (cDMXU_Release() < 0)
    CHK_ERROR(channel, "Getting channel failed ");

  // return OK
  return 0;
}
```

3.2.2 C$^{\#}$-Code

Interfacing the `cDMXU.dll` shared library from C$^{\#}$ can be achieved by using the `System.Runtime.InteropServices` service. An example class definition for wrapping the library function is outlined below:

```
using System.Runtime.InteropServices;

namespace nsDMXU
{
  public class cDMXU
  {
    // make sure the DLL can be found
    const string shared_lib = "cDMXU.dll";
    const CallingConvention ccv = CallingConvention.Cdecl;

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_Init(string id);

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_IsInitialized();

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_SetChannel(int channel);

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_GetChannel();

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_GetSerialNumber();

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_GetNumberOfChannels();

    [DllImport(shared_lib, CallingConvention = ccv)]
    static extern int cDMXU_Release();

    public int Init(string id) {
      return cDMXU_Init(id); }

    public int Release() {
      return cDMXU_Release(); }

    public bool IsInitialized() {
      return (cDMXU_IsInitialized() == 0); }

    public int SetChannel(int channel) {
      return cDMXU_SetChannel(channel); }

    public int GetChannel() {
      return cDMXU_GetChannel(); }

    public int GetNumberOfChannels() {
      return cDMXU_GetNumberOfChannels(); }

    public int GetSerialNumber() {
      return cDMXU_GetSerialNumber(); }
  }
}
```

3.2.3 Python

Interfacing the cDMXU.dll shared library from Python can be achieved by using the ctypes foreign function library. It provides C compatible data types, and allows calling functions in shared libraries. It can be used to wrap these libraries in pure Python, as shown in the following example:

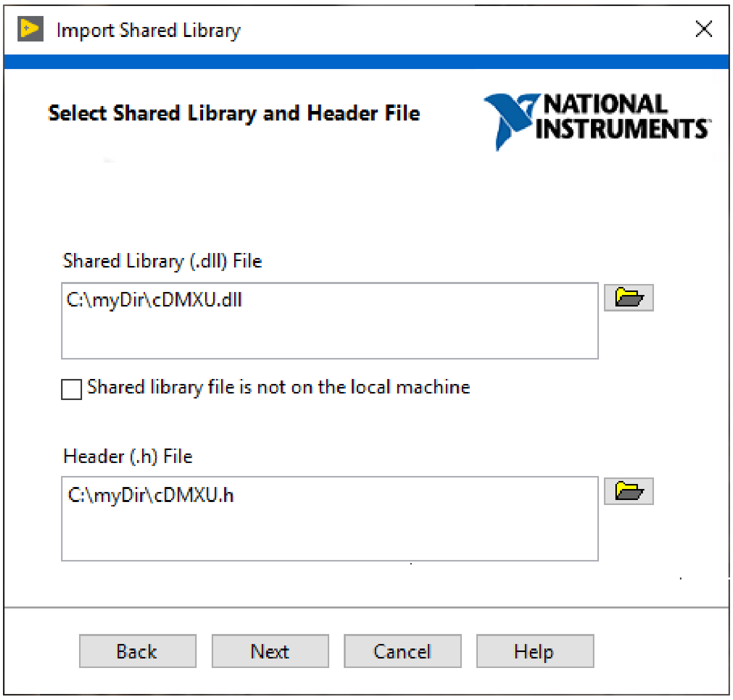
```
import os
from ctypes import *

# set DLL path and check availability
dll = "c:\\MY_PATH\\cDMXUD.dll"
assert(os.path.isfile(dll)), "MUX error: DLL not found"

# load DLL
mux = cdll.LoadLibrary(dll)

# setup argument and result types
mux.cDMXU_Init.restype = c_int
mux.cDMXU_Init.argtypes = [c_char_p]

mux.cDMXU_IsInitialized.restype = c_int
mux.cDMXU_IsInitialized.argtypes = []

mux.cDMXU_SetChannel.restype = c_int
mux.cDMXU_SetChannel.argtypes = [c_int]

mux.cDMXU_GetChannel.restype = c_int
mux.cDMXU_GetChannel.argtypes = []

mux.cDMXU_GetSerialNumber.restype = c_int
mux.cDMXU_GetSerialNumber.argtypes = []

mux.cDMXU_GetNumberOfChannels.restype = c_int
mux.cDMXU_GetNumberOfChannels.argtypes = []

mux.cDMXU_Release.restype = c_int
mux.cDMXU_Release.argtypes = []
```

```
# init library, passing ID not necessary yet
res = mux.cDMXU_Init(b"")
assert(res == 0), "MUX error: init failed"

# check if the library is initialized correctly
res = mux.cDMXU_IsInitialized();
assert(res == 0), "Library not initialized"

# read serial number (e.g. 50001 --> 05.0001)
serno = mux.cDMXU_GetSerialNumber()
assert(serno >= 0), "Reading serial no. failed"

# read number of channels
no_of_channels = mux.cDMXU_GetNumberOfChannels();
assert(no_of_channels > 0), "Reading number of channels failed"

# set mux channel A (B: 2, C: 3, D: 4)
res = mux.cDMXU_SetChannel(1);
assert(res >= 0), "Setting channel A failed"

# read back current channel
channel = mux.cDMXU_GetChannel()
assert(channel >= 0), "Reading current channel failed"

# to set switch to OFF state, pass 0
res = mux.cDMXU_SetChannel(0);
assert(res >= 0), "Setting all OFF failed"

# release library
res = mux.cDMXU_Release()
assert(res == 0), "MUX error: release failed"
```

3.2.4 LabVIEW

Interfacing the `cDMXU.dll` shared library from LabVIEW can be achieved by importing the DLL functions with Labview's *Import Shared Library Wizard*.

This tool parses the header file, lists the functions in the shared library, converts data types in the shared library to LabVIEW data types, and generates a wrapper VI for each function. The wizard saves the VIs in a LabVIEW project library. Continue to the *Use Header File* with the *Import Shared Library Wizard* section. Start the import as follows:


Import Shared Library

Select Shared Library and Header File

Shared Library (.dll) File

C:\myDir\cDMXU.dll

□ Shared library file is not on the local machine

Header (.h) File

C:\myDir\cDMXU.h

1. Launch LabVIEW
2. Navigate to *Tools* → *Import...* → *Shared Library (.dll)* to launch the *Import Shared Library Wizard*.
3. Select *Create VIs for a shared library* and then *Next*

4. Input the file paths for the *Shared Library (.dll) File* and *Header (.h) File*.

5. Continue configuring each page as needed, importing your desired functions, and selecting *Next*.

6. When finished with configuration, select *Finish* to create your LabVIEW Project library `lvlib` file.

An example with the VIs created by means of the wizard is shown below. The example performs some initialization and responds to events triggered by the *Channel* input control. The file `mux-example.vi` can be downloaded from the Teledyne LeCroy website.

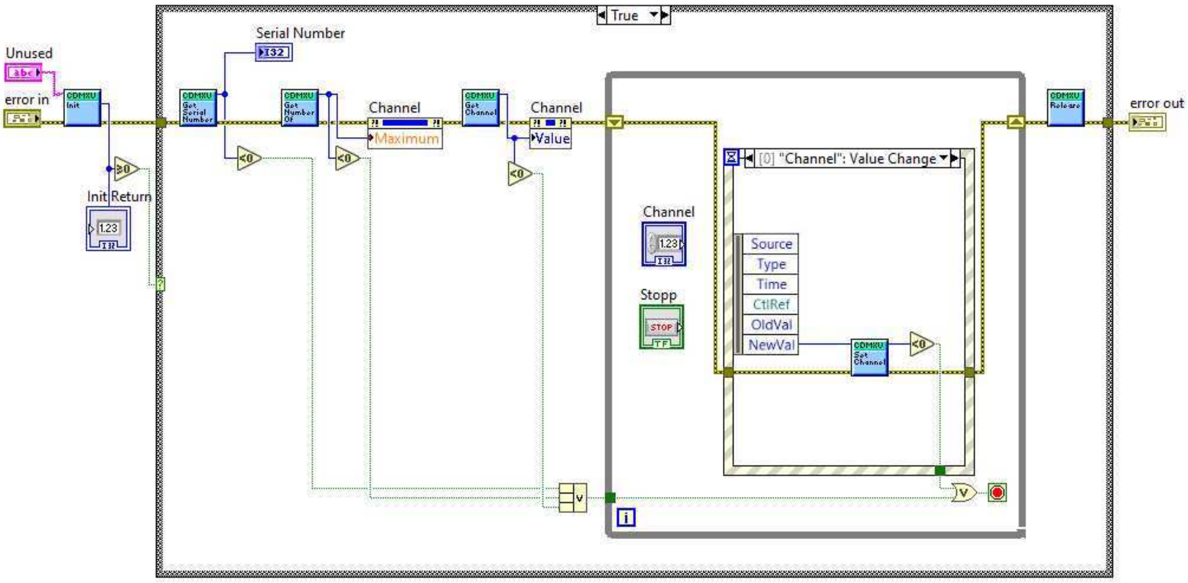


Figure 5: LabVIEW example VI for controlling the T3SP-D4MX

3.2.5 Matlab

Interfacing the `cDMXU.dll` shared library from Matlab can be achieved by using the `calllib` function. It allows for calling functions in shared libraries. It can be used to wrap these libraries in a Matlab m-file, as shown in the following example:

```matlab
function varargout = mux_example(varargin)
% function varargout = mux_example(varargin)
%
% Examples:
%   mux_example('init');
%   mux_example('release')
%   mux_example('getserialnumber')
%   mux_example('getnumberofchannels')
%   mux_example('setchannel', 1); % starting with 1 (0: ALL OFF)
%   mux_example('getchannel');

if ~nargin
  cmd = 'init';
else
  cmd = lower(varargin{1});
end

dll = 'cDMXUD';
shrlib = ['C:\YOUR_PATH\' dll '.dll'];
header = 'C:\YOUR_PATH\cDMXU.h';
assert(exist(header,'file')>0, 'Unable to locate header file');

if ~libisloaded(dll)
  loadlibrary(shrlib,header);
end

is_init = (calllib(dll,'cDMXU_IsInitialized') == 0);

if ~is_init,
  err = calllib(dll,'cDMXU_Init','');
  assert(err == 0, 'Initialization failed');
  is_init = true;
end

if strcmpi(cmd,'init'),
  return;
end

NoC = calllib(dll,'cDMXU_GetNumberOfChannels');
assert(NoC > 0, 'Getting number of channels failed');

% return value
ret = [];

if strcmpi(cmd,'getserialnumber')
  ret = calllib(dll,'cDMXU_GetSerialNumber');
  assert(ret >= 0, 'Getting serial number failed');

elseif strcmpi(cmd,'getnumberofchannels')
  ret = NoC;

elseif strcmpi(cmd,'setchannel')
  % check for errors
  error(nargchk(2,2,nargin));
  ch = varargin{2};

  assert(isnumeric(ch) && isscalar(ch),'Channel must be a scalar
     integer');
  assert(ch == int32(ch) && ch >= 0 && ch <= NoC, 'Invalid channel');

  err = calllib(dll,'cDMXU_SetChannel',ch);
  assert(err >= 0, 'Setting channel failed');

elseif strcmpi(cmd,'getchannel')
  ret = calllib(dll,'cDMXU_GetChannel');
  assert(ret >= 0, 'Getting channel failed');

elseif strcmpi(cmd,'release')
  if is_init,
    err = calllib(dll,'cDMXU_Release');
    assert(err == 0), 'Error while releasing library');
  end
  unloadlibrary(dll);
else
  error('Unknown command');
end

% assign return value
if ~isempty(ret)
  if nargout > 0
    varargout{1} = ret;
  else
    disp(sprintf('%s returned: %d', cmd, ret));
  end
end
```

4. Safety Instructions

4.1 Operating Environment

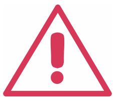
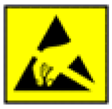
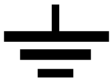
Temperature	0°C to 40°C Direct sunlight, radiators, and other heat sources should be avoided and have to be taken into account when assessing the ambient temperature.
Humidity	5% to 90% RH (non condensing) up to 31°C decreasing linearly to 50% RH at 40°C
Altitude	Up to 3000m at or below 30°C

4.2 Power

DC Voltage	5V via USB port
Power Consumption	< 60mW

4.3 Safety Symbols & Terms

Where the following symbols or terms appear on the instrument front or rear panels, or in this manual, they alert you to important safety considerations.

	This is a CAUTION symbol to alert the user to attend to the accompanying information to protect against personal injury or damage. Do not proceed until conditions are fully understood and met
	This symbol states that ESD-precautions are required at all RF ports.
	This symbol is used to denote a safety ground connection.

OBSERVE ALL TERMINAL RATINGS

Do not apply a voltage to any input that exceeds the maximum rating of that input. Refer to the markings on the unit's front panel.

Pleaser refer to the T3SP Series User Manual for more detailed information at http://teledynelecroy.com/

5. Maintenance

- Do not use abrasive cleaning agents. To remove tenacious contaminations use a commercial, non-abrasive cleaning agent.

- If necessary, clean the T3SP-D4MX carefully by wiping it with a humid cloth.

6. Certifications

Teledyne LeCroy certifies compliance to the following standards as of the time of publication. Please see the EC Declaration of Conformity document shipped with your product for current certifications.

6.1 EMC Compliance

6.1.1 EC DECLARATION OF CONFORMITY - EMC

The instrument meets intent of EC Directive 2014/30/EU for Electromagnetic Compatibility. Compliance was demonstrated to the following specifications listed in the Official Journal of the European Communities:

EN 61326-1:2013, EN 61326-2-1:2013 EMC requirements for electrical equipment for measurement, control, and laboratory use [1]

Electromagnetic Emissions:

EN 55011:2016+A1:2017, Radiated and Conducted Emissions Group 1, Class A [2,3]

Electromagnetic Immunity:

EN 61000-4-2:2009 Electrostatic Discharge, 4 kV contact, 8 kV air, 4 kV vertical/horizontal coupling planes [4]

EN 61000-4-3:2006+ A2:2010 RF Radiated Electromagnetic Field, 3 V/m, 80 - 1000 MHz; 3 V/m, 1.4 - 2 GHz; 1 V/m, 2 - 2.7 GHz

[1] To ensure compliance with all applicable EMC standards, use high-quality shielded interface cables.

[2] Emissions which exceed the levels required by this standard may occur when the instrument is connected to a test object.

[3] This product is intended for use in nonresidential areas only. Use in residential areas may cause electromagnetic interference.

[4] Meets Performance Criteria ”B” limits of the respective standard: during the disturbance, product undergoes a temporary degradation or loss of function or performance which is self-recoverable.

European Contact:*

Teledyne GmbH, European Division
Im Breitspiel 11c
D-69126 Heidelberg, Germany
Tel: +49 6221 82700

6.1.2 AUSTRALIA & NEW ZEALAND DECLARATION OF CONFORMITY – EMC

The instrument complies with the EMC provision of the Radio Communications Act per the following standards, in accordance with requirements imposed by Australian Communication and Media Authority (ACMA):

AS/NZS CISPR 11:2015 Radiated and Conducted Emissions, Group 1, Class A.

Australia / New Zealand Contacts:*

RS Components Pty Ltd.
Suite 326 The Parade West
Kent Town, South Australia 5067

RS Components Ltd.
Units 30 & 31 Warehouse World
761 Great South Road
Penrose, Auckland, New Zealand

* Visit teledynelecroy.com/support/contact for the latest contact information.

6.2 Safety Compliance

6.2.1 EC DECLARATION OF CONFORMITY – LOW VOLTAGE

The instrument meets intent of EC Directive 2014/35/EU for Product Safety. Compliance was demonstrated to the following specifications as listed in the Official Journal of the European Communities:

EN 61010-1:2010 Safety requirements for electrical equipment for measurement, control, and laboratory use – Part 1: General requirements

EN 61010-2:030:2010 Safety requirements for electrical equipment for measurement, control, and laboratory use – Part 2-030: Particular requirements for testing and measuring circuits

The design of the instrument has been verified to conform to the following limits put forth by these standards:

- Mains Supply Connector: Overvoltage Category II, instrument intended to be supplied from the building wiring at utilization points (socket outlets and similar).
- Measuring Circuit Terminals: No rated measurement category. Terminals not intended to be connected directly to the mains supply.
- Unit: Pollution Degree 2, operating environment where normally only dry, non-conductive pollution occurs. Temporary conductivity caused by condensation should be expected.

6.3 Environmental Compliance

6.3.1 END-OF-LIFE HANDLING

The instrument is marked with this symbol to indicate that it complies with the applicable European Union requirements of Directives 2012/19/EU and 2006/66/EC on Waste Electrical and Electronic Equipment (WEEE) and Batteries.

The instrument is subject to disposal and recycling regulations that vary by country and region. Many countries prohibit the disposal of waste electronic equipment in standard waste receptacles. For more information about proper disposal and recycling of your Teledyne LeCroy product, please visit teledynelecroy.com/recycle.

6.3.2 RESTRICTION OF HAZARDOUS SUBSTANCES (RoHS)

EC DECLARATION OF CONFORMITY – RoHS

Unless otherwise specified, all the materials and processes are compliant with RoHS Directive 2011/65/EU in its entirety, inclusive of any further amendments or modifications of said Directive.

CHINA RoHS 2

Unless otherwise specified, all the materials and processes are compliant with the latest requirements of China RoHS 2. The hazardous substances contained in the instrument are disclosed in accordance with the standards SJ/T 11364-2014 (Marking for the restricted use of hazardous substances in electronic and electrical products) and GB/T 26572-2011 (Requirements on concentration limits for certain restricted substances in electrical and electronic products). The instrument is marked with an appropriate Environmental Friendly Use Period (EFUP) symbol. The packaging materials include the appropriate recycling labels. The below substance disclosure tables (in Chinese and English languages) provide the required compliance information.

部件名称	有毒有害物质和元素					
	铅 (Pb)	汞 (Hg)	镉 (Cd)	六价铬 (Cr6+)	多溴联苯 (PBB)	多溴二苯醚 (PBDE)
PCBAs	X	O	O	O	O	O
机械硬件	O	O	O	O	O	O
金属片	O	O	O	O	O	O
塑料部件	O	O	O	O	O	O
电缆组件	X	O	O	O	O	O
显示器	O	O	O	O	O	O
电源	O	O	O	O	O	O
风扇	O	O	O	O	O	O
电池	O	O	O	O	O	O
电源线	O	O	O	O	O	O
外部电源(如有)	X	O	O	O	O	O
探头(如有)	X	O	O	O	O	O
熔丝(如有)	O	O	O	O	O	O
产品外壳(如有)	O	O	O	O	O	O
适配器/模块(如有)	O	O	O	O	O	O
鼠标(如有)	O	O	O	O	O	O
O: 表明该有毒有害物质在该部件所有均质材料中的含量均在SJ/T11364-2014标准规定的限量要求之下。						
X: 表示该有毒有害物质至少在该部件的某一均质材料中的含量超出SJ/T11364-2014标准规定的限量要求。						

EFUP (对环境友好的使用时间): 30年。

使用条件：参阅用户手册“环境条件”部分的规定。

探头EFUP: 10年。

Part Name	Toxic or Hazardous Substances and Elements					
	Lead (Pb)	Mercury (Hg)	Cadmium (Cd)	Hexavalent Chromium (Cr6+)	Polybrominated Biphenyls (PBB)	Polybrominated Diphenyl Ethers (PBDE)
PCBAs	X	O	O	O	O	O
Mechanical Hardware	O	O	O	O	O	O
Sheet Metal	O	O	O	O	O	O
Plastic Parts	O	O	O	O	O	O
Cable Assemblies	X	O	O	O	O	O
Display	O	O	O	O	O	O
Power Supply	O	O	O	O	O	O
Fans	O	O	O	O	O	O
Batteries	O	O	O	O	O	O
Power Cord	O	O	O	O	O	O
Ext Power Supply (if present)	X	O	O	O	O	O
Probes (if present)	X	O	O	O	O	O
Fuse (if present)	O	O	O	O	O	O
Product Case (if present)	O	O	O	O	O	O
Adapters/Modules (if present)	O	O	O	O	O	O
Mouse (if present)	O	O	O	O	O	O
O: Indicates that this toxic or hazardous substance contained in all of the homogeneous materials for this part is below the limit requirement specified in SJ/T11364-2014.						
X: Indicates that this toxic or hazardous substance contained in at least one of the homogenous materials used for this part is above the limit requirement specified in SJ/T11364-2014.						

EFUP (Environmental Friendly Use Period): 30 years.

Use Conditions: Refer to the environmental conditions stated in the User Manual.

EFUP for Probes: 10 years.