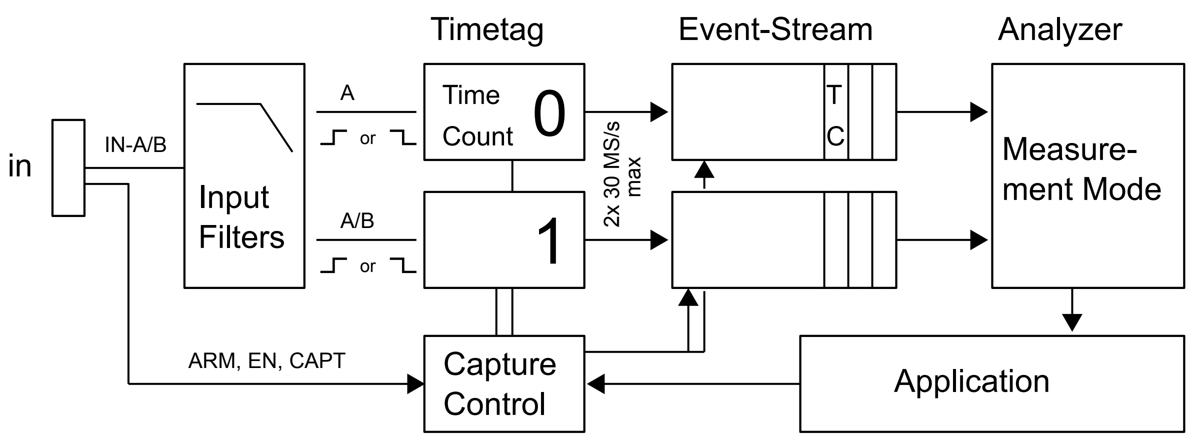
1 Overview

Figure 1: Timing Analyzer Architecture

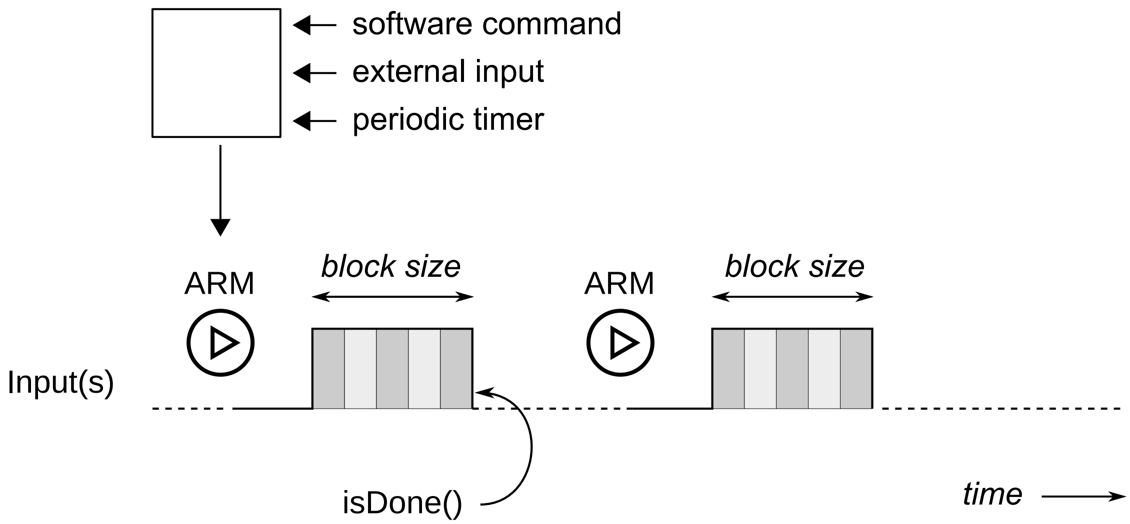


The TIC-8420 system is shown in Figure 1. After optionally low-pass filtering the inputs, the Timetag unit continuously counts edges and records the time of their appearance. At the same time, the Capture Control unit decides when to capture a count-time pair (event) into the Event-Stream.

The Analyzer functions consume the events from the Event-Stream in order to provide measurement results to the application program.

Due to the continuous event streaming architecture all measurements can be made on a zero-dead-time basis (back to back) without loss of data. Within the limits of bus speed (USB/PCI) events may be logged continuously.

Figure 2: Arming for Measurements



In any mode, the acquisition of timing data begins with the ARM signal. The ARM signal can be given "on-demand" by software control, external pulse, or on periodic timer intervals (Figure 2).

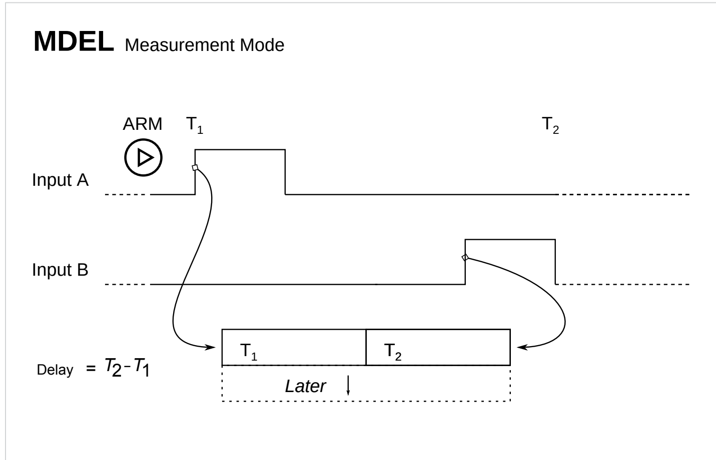
Block size
Per ARM block a pre-defined count of measurements can be taken. The default count is 1, for a single measurement per ARM pulse. Setting the block size to zero results in continuously taken measurements started at a single ARM pulse.

2 Basic Measurement Modes

2.1 MDEL - Delay (Start/Stop)

Delay is the time difference between the activation of the A and B channel. The delay is positive if the A channel is activated prior to the B channel. The delay is zero if the A and B channel are activated at the same time. The delay is negative if the B channel is activated prior the the A channel.

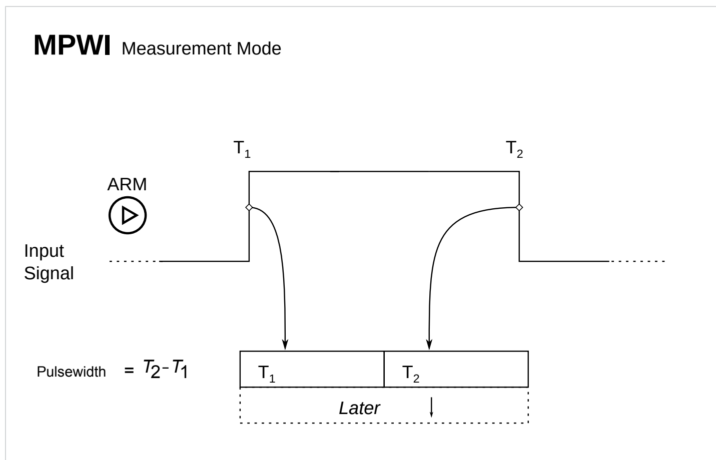
The tic_delay example demonstrates how to use this mode in an application program.



2.2 MPWI - Pulse Width

In positive edge mode, pulse width is the distance between the low-to-high and the subsequent high-to-low edge of a pulse. In negative edge mode, pulse width is the distance between the high-to-low and the subsequent low-to-high edge of a pulse. By definition, pulse width is always a non-zero positive value.

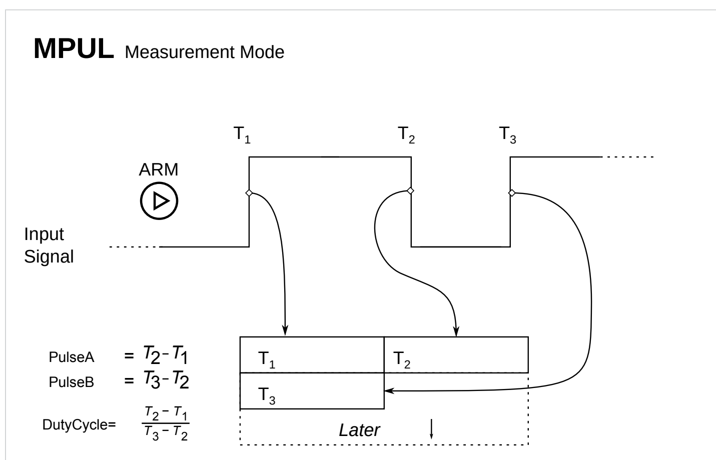
The tic_pulsewidth example demonstrates how to use this mode in an application program.



2.3 MPUL - Pulse

This mode measures the active as well as the inactive time of a pulse. The pulse period and the duty cycle are calculated. In positive edge mode, the measurement starts with the low-to-high edge of the input signal. In negative edge mode, the measurement starts with the high-to-low edge of the input signal. The duty cycle is the ratio between the first and the second half of the pulse. By definition, the duty cycle may approach but never reach the value of 0 or 1.

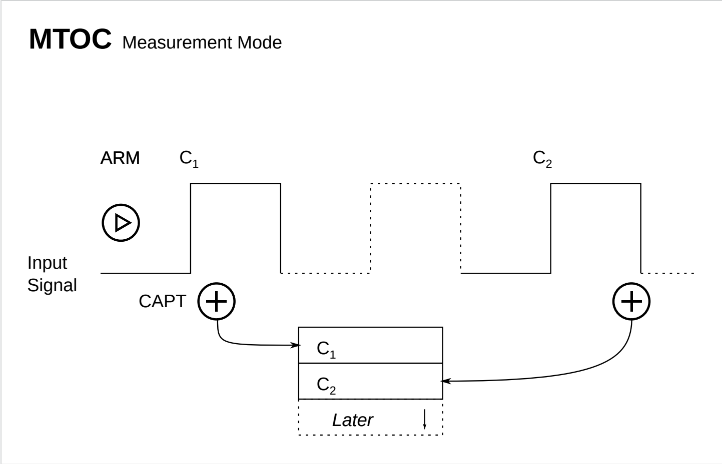
The tic_pulse example demonstrates how to use this mode in an application program.



2.4 MTOC - Totalizing Counter

In totalizing counter mode, the Timetag unit continuously counts the active edges of the input signal. Only on a CAPT internal pulse, the event is captured. The CAPT signal can be given "on-demand" by software control, external pulse, or on periodic timer intervals. The counter's values along with the respective capture times are available from the analyzer.

In positive edge mode, the low-to-high edges of the input signal are counted. In negative edge mode, the high-to-low edges of the input signal are counted.



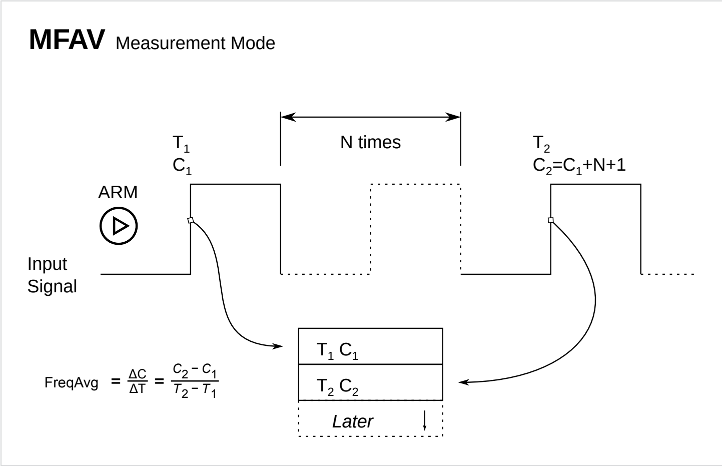
The tic_totalcount example demonstrates how to use this mode in an application program.

Note that, while this mode combined with periodic capture is able to provide frequency/period average measurements, a more precise, dedicated mode MFAV is provided (see below).

2.5 MFAV - Frequency Average

For frequency/period average measurements an interval *N* must be selected. An event is captured for every *N*th leading signal edge. The selection of *N* affects measurement time and resolution, see Section 5.

The tic_freq example demonstrates how to use this mode in an application program.



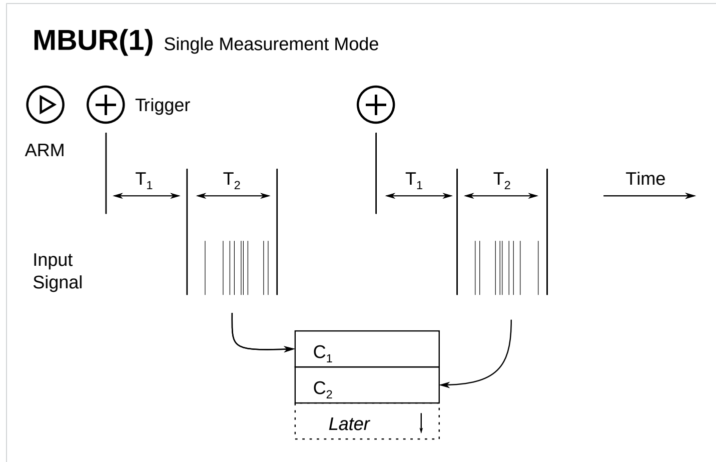
3 Advanced Measurement Modes

3.1 MBUR - Burst Mode

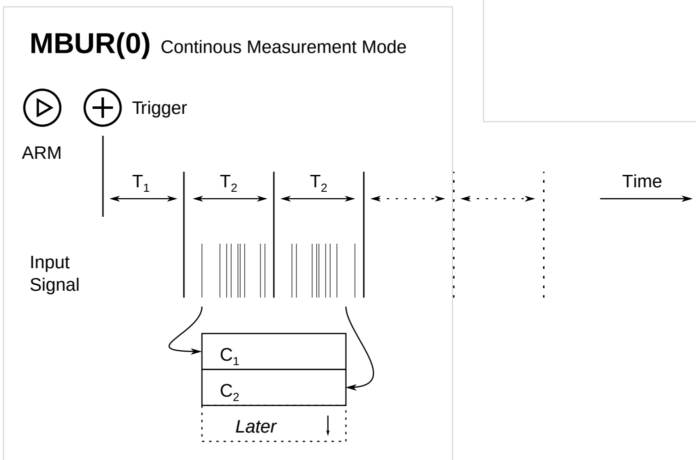
The burst mode implements capture of the counters with programmable timing schemes.

Single Mode

In the Single Mode the counter captures a single interval per trigger event. First, the counter waits until the trigger event occurs. After trigger, the T_1 parameter specifies the delay from the trigger event until the begin of the counting window. The length of the counting window is defined by the T_2 parameter.



Continous Mode



In the Continous Mode the counter takes samples continously after the trigger event occured once. First, the counter waits until the trigger event occurs. After trigger, the T_1 parameter specifies the delay from the trigger event until the begin of the counting window. The length of the counting window is defined by the T_2 parameter.

Parameters

TRM	Trigger Mode(integer): ASCII character code s (115 dec, 0x73 hex) for software command, b (98 dec, 0x62 hex) IO12 input leading edge starts window.
CTM	Capture Timing Mode(integer): 1 selects Single Mode, 0 selects Continues Mode.
CCH	Channel Count(integer): 1 selects channel A only, 2 selects A+B.
T1	timing window delay (floating point, seconds)
T2	timing window length (floating point, seconds)

4 Input Filters

The TIC-8420 provides low-pass (debouncing) filters at each digital input. The filter setting for each input can be configured independently. The table lists the Filter code to set the filter by the application program, the pulse width guaranteed to pass the filter (T_{pass}), the pulse width guaranteed to not pass the filter (T_{glitch}), and the maximum frequency able to pass the filter (f_{max}).

Input Filters

Code	T_{pass}	T_{glitch}	f_{max}
0	2.78 ns	(unfiltered)	180 MHz
31	5.56 ns	2.78 ns	90.0 MHz
33	11.1 ns	5.56 ns	45.0 MHz
35	22.2 ns	11.1 ns	22.5 MHz
37	44.4 ns	22.2 ns	11.3 MHz
8	200 ns	167 ns	2.50 MHz
9	400 ns	333 ns	1.25 MHz
10	800 ns	667 ns	625 kHz
11	1.60 µs	1.33 µs	313 kHz
12	3.20 µs	2.67 µs	156 kHz
13	6.40 µs	5.33 µs	78.1 kHz
14	12.8 µs	10.7 µs	39.1 kHz
15	25.6 µs	21.3 µs	19.5 kHz
16	51.2 µs	42.7 µs	9.77 kHz
17	102 µs	85.3 µs	4.88 kHz
18	205 µs	171 µs	2.44 kHz
19	410 µs	341 µs	1.22 kHz
20	819 µs	683 µs	610 Hz
21	1.64 ms	1.37 ms	305 Hz
22	3.28 ms	2.37 ms	153 Hz
23	6.55 ms	5.46 ms	76.3 Hz
24	13.1 ms	10.9 ms	38.1 Hz

5 Measurement Resolution

The measurement resolution (the number of significant digits of the result) depends on the measurement time only. For example, a pulse of 2.8µs cannot be measured with more than 3 decimal digits resolution by the TIC-8420, whereas a pulse of 2.8ms is measured with 6 digits.

While the resolution is determined by the input signal in most modes, the frequency/period average mode MFAV allows to trade off measurement time against resolution. The is done by selecting a larger *N* to increase resolution (along with measurement time). To speed up measurements, *N* must be decreased at the cost of resolution.

Sign.	Measurement	Sample Freq	Range
2 ½	0.56 µs	1.79 MHz	200
3	2.8 µs	359 kHz	1 k
3 ½	5.6 µs	178 kHz	2 k
4	28 µs	35.9 kHz	10 k
4 ½	56 µs	17.8 kHz	20 k
5	280 µs	3.59 kHz	100 k
5 ½	560 µs	1.78 kHz	200 k
6	2.8 ms	359 Hz	1 M
6 ½	5.6 ms	178 Hz	2 M
7	28 ms	35.9 Hz	10 M
7 ½	56 ms	17.8 Hz	20 M
8	280 ms	3.59 Hz	100 M
8 ½	560 ms	1.78 Hz	200 M
9	2.78 s	0.359 Hz	1 G

6 Programming

The instrument is controlled by applications using the `tic` programming interface (API) which is part of the ines Instruments Driver Library (iDil).

System Initialization

Before use of any other function the measurement system must be initialized by the tic_init() function. If the system is no longer used, resource can be released tic_fini() function.

Measurement Setup

The measurement task must be configured by calling the matching tic_mode_xxx function. Once the measurement mode has been selected, properties for the mode can be modified, using tic_set_property(). If the measurement is armed by software control (i.e. not externally or timer based) tic_arm() must be called.

Transfer Loop

The transfer loop transfers raw measurement data from the instrument's hardware to the analyzer functions implemented in software. The transfer loop must be operated as long as measurements are required. Within the transfer loop tic_loop() is called.

Fetching Results

Measurement results are always stored in a queue by the analyzer functions. Whenever the queue is not empty, elements may be fetched from that queue. tic_count_samples() returns the number of unread queue elements. The element's data can be read by the corresponding functions:

MFAV	tic_read_frequency_average() fetches the next queue element and returns the frequency average as floating point number.
MPWI	tic_read_pulse_width() fetches the next queue element and returns the pulse width as floating point number.
MPUL	tic_read_pulse_first() fetches the next queue element and returns the width of the first half as floating point number. tic_read_pulse_second() returns the width of the second half of the queue element previously fetched by tic_read_pulse_first() as floating point number. You must call tic_read_pulse_first() prior to tic_read_pulse_second(), even if the value of the first half is not required.
MTOC	tic_read_totalcount() fetches the next queue element and returns the count width as integer number.

7 Connecting External Signals

The input and output signals of the TIC-8420 are CMOS/TTL compatible, with a logic level L ≤ 0.8V and logic H ≥ 2.0V. A weak-pulldown at each input defines it when unconnected. For input characteristics see Figure 10.

The list explains the input signals to the TIC-8420. GND is the ground-reference for all lines.

IN-A/B	The input waveform signal. For MDEL measurement the start signal is connected to IN-A, the stop signal to IN-B.
EN-A/B	The input enable signal if external enable is selected. Ignored if external enable is not selected.
ARM-A/B	The arm signal if external arm is selected. Ignored if external arm is not selected.
CAPT-A/B	The external capture signal if external capture is selected. Ignored if external capture is not selected.

TIC-8420 Signal by Pin

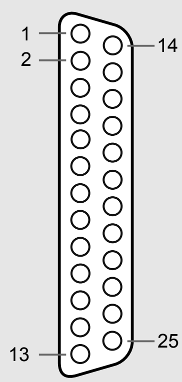
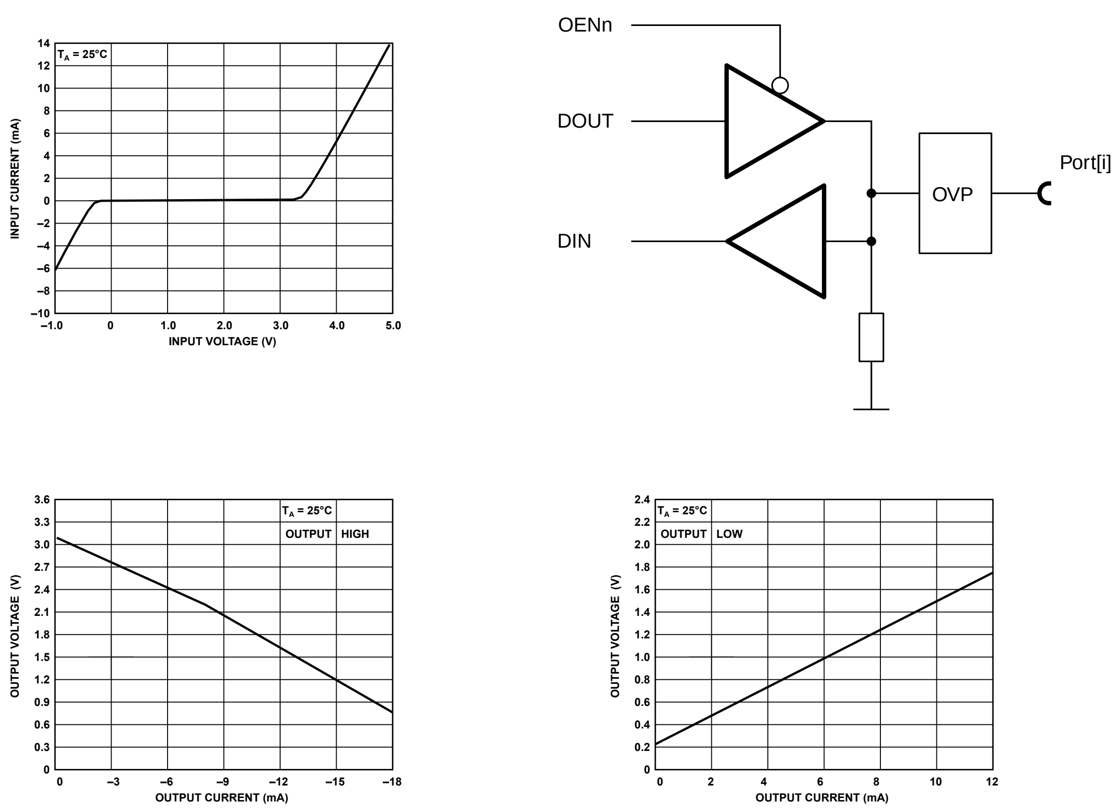
Signal	Pin	Connector View	Pin	Signal
3,3V	1		14	GND
IN-A/IO15	2		15	IN-B/IO14
GND	3		16	EN-B/IO13
EN-A/IO12	4		17	GND
ARM-B/IO11	5		18	ARM-A/IO10
GND	6		19	CAPT-B/IO09
CAPT-A/IO08	7		20	GND
IO07	8		21	IO06
GND	9		22	IO05
IO04	10		23	GND
IO03	11		24	IO02
GND	12		25	IO01
IO00	13			

Figure 10: I/O Characteristics

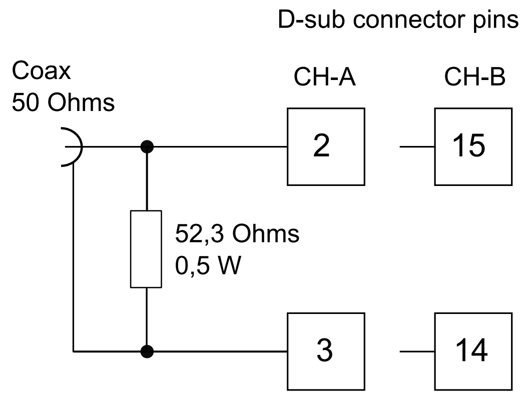


7.1 Signal Termination

Some applications may need proper termination of 50 Ohms transmission lines (e.g. coaxial cables). While the TIC-8420 inputs are CMOS/TTL compatible (technologies without impedance specification) they can easily made compatible with 50 Ohms transmission lines. A resistor and a coaxial connector can provide the required termination (see Figure 11 for signals IN-A and IN-B).

The outputs of the TIC-8420 can **not** drive directly into 50 Ohms.

Figure 11: 50 Ohms Termination Schematic



Important Notice

No references to the C++ API documentation are available in this standalone version of the manual. In order to access the C++ API documentation download and install the complete driver software package.

Limited warranty and liability

Information in this document is believed to be accurate and reliable. However, the publisher does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. The publisher takes no responsibility for the content in this document if provided by an information source outside of the publisher. In no event shall the publisher be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, the publishers' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of the publisher.

Trademarks

Product, service, or company names used in this document are for identification purposes only and may be either trademarks or registered trademarks of the relevant trademark owners. LabView, NI-488.2, LabWindows, PXI, DASYLab, DIAdem are trademarks or registered trademarks of National Instruments Corp., USA, in the United States and/or other countries. Microsoft, Windows, Windows NT, Windows CE, Windows 2000, Windows ME, Windows XP, Windows Vista, Visual Basic, Visual-C++ are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Specifications

All specifications are subject to change without prior notice.

B2B only

Our range of products and services is directed exclusively at commercial customers, institutes and public authorities, as well as other natural or legal persons or partnerships with legal capacity, who, when concluding a legal transaction, act in the exercise of their commercial or independent professional activity. Any business to consumers (i.e. natural persons who conclude a legal transaction for purposes which can predominantly be attributed neither to their commercial nor to their self-employed professional activity) is excluded.

Software

ALL SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.