# TDR5000

Version of Product V1.0

programming guide

## 1 Contents

programming guide

## 2 Registers and XPort Adresses

The registers of the TDR5000 are accessed by the Tiger-BASIC functions XIN24, XOUT24, XSET24 or XRES24.

The address used with this instructions are register addresses plus a 24 bit base address. The bits 0..15 of the base address can be selected by the four HEX switches. The bits 16 to 23 of the base address are always 0.

Please take a look at the description of the XSETUP function to see, how 24 bit addresses are enabled in your software. If you use IO-modules using 8 bit addresses then note that the used addresses of this modules are mirrored in the 24 bit address room every 256 addresses.

Make sure that used addresses do not overlap with addresses used by other modules.

Depended on the software version of your Tiger-BASIC or Tiger C it may not be possible to use 24 bit addresses with the XIN24, XOUT24, XSET24 or XRES24 functions. You can update your Software or if you don't want to do this then you can set the high byte of the base address to 0 and use the 8 bit functions XIN, XOUT, XSET or XRES.

The best performance will be reached by using 8 bit addresses and switching the high byte of the address to 0. If doing so make sure that XIN24, XOUT24, XSET24 or XRES24 functions or XIN16, XOUT16, XSET16 or XRES16 functions are not used in your application anywhere. Make also sure that L87 and L77 remain to low level output state.

Please use XIN24 or XIN functions with a delay parameter of at least 14 as shown in the following example:

```
EncMatch = &
XIN24(BASEADDRESS+TDR5000_ENC_MATCH,14)
```

The register names are defined in the file DEFINE_A.INC. We recommend to use the names instead of the Register Address to ensure that your software will be compatible to later versions of the TDR5000 even if the Addresses changes. The Register names are marked green in this programming guide.

Note: You can download the latest version of the DEFINE_A.INC file from the WEB, see http://www.wilke.de/

## 3 Reset Register

Setting bit 0 of the RESET register will reset all functions of the TDR5000 module.
Bit 1 of this register act as interrupt acknowledge.

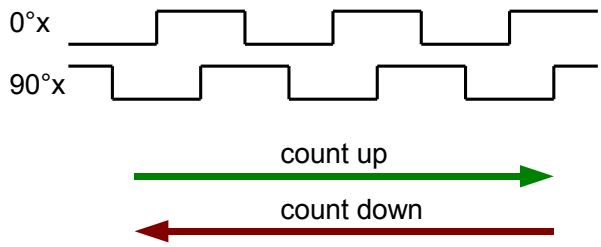
| Register Address | Register |
|---|---|
| 0 dec.<br>00 hex | TDR5000_RESET<br><br>REET register<br>write only<br><br>bit 0:<br>Setting bit 0 of this register will reset all functions. After REST the bit is cleared internally. Resetting the bit is not necessary and has no effect.<br><br>bit 1:<br>Setting bit 1 of this register will set the interrupt state to acknowledged. The bit is cleared internally. Resetting the bit is not necessary and has no effect. |

programming guide

# 4 Encoder Inputs

## 4.1 Normal Counter Mode

The TDR5000 includes inputs for 6 encoders. An encoders generates two 90° phase shifted signals. The frequency of this signals is proportional to the rotation speed at rotary encoders or to the movement speed at linear encoders. The position of the encoder can be detected by counting the edges of the signals. The movement direction can be detected due to the 90° phase shift between the 2 signals.




## 4.2 Single Counter Mode

In single counter mode only the pulse signal 0°x is neccessary. The counting direction is always counting up at each rising edge of the signal 0°x. The signal 90°x is ignored in this mode.
Each counter can be set individually to single or normal counter mode.

## 4.3 Both Counter Modes

Each of the 6 encoder inputs controls a 24bit up-down counter which can be read by the Tiger software. The counter can be enabled or disabled by an enable signal or under software control. The counter value can be set synchronous to a rising or falling edge of a reference signal or independent of this signal.

The counter value is latched into a buffer by reading the low byte of the counter value. So a correct value is read, even if the counter value changes between reading the low byte and the high byte.
You can alternatively latch the counter values by setting bits of the Synchronous latch register. So it's possible to latch the values of two or more counters absolute synchronously.

Before getting access to the counter value or the control register of a counter, the counter to must be selected by writing the counter no. to the counter select register.

programming guide

### 4.4 Used Register Addresses

| Register Address | Register |
|---|---|
| 1 dec.<br>01 hex | TDR5000_ENC_SYNC_LATCH<br>Synchronous latch<br>R/W<br>Setting a bit of this register will latch the counter value of the corresponding up down counter.<br>bit 0: Up-Down counter 0 is latched<br>bit 1: Up-Down counter 1 is latched<br>bit 2: Up-Down counter 2 is latched<br>...<br>bit 5: Up-Down counter 5 is latched<br>If the corresponding bit of the up down counter is '0' then the counter value is latched reading the low byte of the counter value. |
| 2 dec.<br>02 hex | TDR5000_ENC_SELECT<br>Encoder Select<br>(R/W)<br>Writing a value between 0...5 will select the register set of the up down counter 0...5 |
| 3 dec.<br>03 hex | TDR5000_ENC_LOW<br>R:<br>Up- Down counter low byte<br>W:<br>Preset value low byte. The low byte of the selected counter is set to this value, if the reference signal is active. |
| 4 dec.<br>04 hex | TDR5000_ENC_MID<br>R:<br>Up- Down counter mid byte<br>W:<br>Preset value mid byte. The mid byte of the selected counter is set to this value, if the reference signal is active. |
| 5 dec.<br>05 hex | TDR5000_ENC_HIGH<br>R:<br>Up- Down counter high byte<br>W:<br>Preset value high byte. The high byte of the selected counter is set to this value, if the reference signal is active. |
| 6 dec.<br>06 hex | TDR5000_ENC_CONTROL<br>Counter Control<br>R/W<br>bitmap to control the counter behavioral, see table below |
| 45 dec.<br>2D | TDR5000_ENC_SINGLEMODE<br>Single Mode Select<br>(R/W):<br>bitmap to set the corresponding counter in single counter mode.<br>bit 0 controls counter 0,<br>bit 1 controls counter 1,<br>bit 2 controls counter 2,<br>...<br>bit 5 controls counter 5<br>bit value = 0: normal counter mode<br>bit value = 1: single counter mode |

programming guide

### 4.5 Counter Controll Register

| Bit | Description |
|---|---|
| 0 | enable bit<br>if software enable is active:<br>0: counter is stopped<br>1: counter is enabled |
| 1 | external enable bit<br>0: counter is enabled using the enable bit<br>1: counter is enabled using the hardware signal EC_EN_x |
| 2 | invert hardware enable<br>0: hardware enable signal is high active<br>1: hardware enable signal is low active |
| 3 | software reference:<br>If software reference is selected then writing a 1 to this bit will cause that the counter is loaded with the preset value or set to 0.<br>This bit will be cleared after loading the counter value regardless of the reference mode. |
| 4 | reference mode<br>0: software reference is selected<br>1: hardware reference is selected<br>If hardware reference is selected then the selected edge of the hardware signal EC_REF_x will cause that the counter is loaded with the preset value or set to 0.<br>This bit will be cleared after loading the counter. |
| 5 | invert hardware reference signal<br>0: a rising edge of EC_REF_x is selected<br>1: a falling edge of EC_REF_x is selected. |
| 6 | Counter Clear<br>0: At the selected change of the reference signal the counter is set to the preset value.<br>1: At the selected change of the reference signal the counter is set to 0. |
| 7 | Overflow mode<br>0: an overflow will occur if the counter value rises above FFFFFF hex. If the counter value shrinks below 0, then the counter is set to FFFFFF hex.<br>1: an overflow will occur if the counter value rises above the preset value. If the counter value shrinks below 0, then the counter is set with the preset value. |

## 5 Pulse Generators

The digital open drain outputs can be used as general purpose outputs or as pulse outputs.

There are 6 pulse generators. Each of them have a pulse output and a direction output. The outputs of the generators are connected to the Open Drain Output Channels as shown in the table below:

| Pulse Generator Chanel | Output Type | Open Drain Output Chanel |
|---|---|---|
| 0 | pulse | 0 |
| 0 | direction | 1 |
| 1 | pulse | 2 |
| 1 | direction | 3 |
| 2 | pulse | 4 |
| 2 | direction | 5 |
| 3 | pulse | 6 |
| 3 | direction | 7 |
| 4 | pulse | 8 |
| 4 | direction | 9 |
| 5 | pulse | 10 |
| 5 | direction | 11 |

### 5.1 Puls rate

The puls rate is set by a global precounter and the individual on and off pulse width of each channel.

To get the base frequency $\mathbf{f_b}$ the precounter divides a frequency of **187.5kHz** trough a 16 bit value which is written to the **precounter value**.
If you write **0** to the precounter registers then the precounter stops counting and consequently all puls outputs are **stoped**.

The on time is the time when the FET in low impedance state and the off time is the time when the FET is in high impedance state. The on and the off time are set by writing the time in to the Pulse Width On and Pulse Width Off registers.

The On Time $T_{ON}$ is calculated by
$T_{ON}$ = value in Pulse Width On Register / $f_b$

The Off Time $T_{OFF}$ is calculated by
$T_{OFF}$ = value in Pulse Width Off Register / $f_b$

### 5.2 Number Of Pulses

The number of pulses is set to the No Of Pulses register.
After writing the high byte of the No Of Pulses register the pulse width on, the pulse width off and the number of pulses are stored into a double buffer if enabled and the TP5000 starts with putting out the pulses.

If the double buffer is enabled you can write the next values (Pulse Width On, Pulse With Off, No Of Pulses) to the registers of the same channel during the first pulses are putted out as soon as the "Latch Possible" bit of the Pulse Control Register is set. After all pulses with the first parameters are out, the next values are loaded into the double buffer and the TP5000 starts to put out the next pulses without any delay.

If the double buffer is disabled then writing new values will cause that the new parameters are used immediately after the pulse which is running.

The double buffer can be enabled or disabled by clearing or setting the "disable double buffer" bit of the Pulse Control Register

No Of Pulses expect a signed 16 bit value. If the value is negative, then direction signal is active and the absolute value of pulses are send to the output.

### 5.3 Direction Output

The direction output is controlled with the No Of Pulses register. The value in this register is used as a signed 16 bit word. If a negative value is written to the No Of Pulses register, then the absolute number of pulses is putted out with the direction output switched on. At positive values the direction output is switched off.

### 5.4 Synchronous Output of sets of pulses

You can sync sets of pulses of two or more channels by setting the "sync mode bit" of the Pulse Control register.

If the "sync mode bit" is set then the pulse generator will wait after putting out all pulses until all channels with a set "sync mode bit" have putted out all pulses, too.

After all pulses of all generators with a set "sync mode bit" are out the next set of parameters is loaded into the double buffer of each generator which is in sync mode and these channels start with the next set of pulses synchronously.

The sync mode only works with generators where the double buffer is enabled. Do not set the "sync mode bit" to generators where the double buffers are disabled.

### 5.5 Used Register Adresses

| Register Address | Register |
|---|---|
| 7 dec.<br>07 hex | TDR5000_PULSE_SELECT<br>Pulse Generator Select<br>(R/W)<br>Writing a value between 0...5 will select the register set of the Pulse Generator 0...5 |
| 8 dec.<br>8 hex | TDR5000_PULSE_CONTROL<br>Pulse Control Register<br>(R/W) |
| 9 dec.<br>09 hex | TDR5000_PULSE_WIDTH_ON_LOW<br>Pulse Width On (low byte)<br>(write only) |
| 10 dec.<br>0A hex | TDR5000_PULSE_WIDTH_ON_HIGH<br>Pulse Width On (high byte)<br>(write only) |
| 11 dec.<br>0B hex | TDR5000_PULSE_WIDTH_OFF_LOW<br>Pulse Width Off (low byte)<br>(write only) |
| 12 dec.<br>0C hex | TDR5000_PULSE_WIDTH_OFF_HIGH<br>Pulse Width Off (high byte)<br>(write only) |
| 13 dec.<br>0D hex | TDR5000_PULSE_NO_LOW<br>No Of Pulses (low byte)<br>(write only) |

programming guide

programming guide

| Register Address | Register |
|---|---|
| 14 dec.<br>0E hex | TDR5000_PULSE_NO_HIGH<br>No Of Pulses (high byte)<br>(write only)<br>No Of Pulses expect a signed 16 bit value. If the value is negative, then direction signal is active and the absolute value of pulses are send to the output. |
| 15 dec.<br>0F hex | TDR5000_PULSE_PREC_LOW<br>Pulse Precounter (low Byte) |
| 16 dec.<br>10 hex | TDR5000_PULSE_PREC_HIGH<br>Pulse Precounter (high Byte) |

### 5.6 Pulse Control Register

| Bit | Description |
|---|---|
| 0 | enable (write)<br>0: will reset the selected pulse generator, the buffer and double buffer.<br>the pulse generator is disabled<br>1: the pulse generator is enabled |
| 1 | sync mode bit<br>0: the selected counter runs independent of the others<br>1: generator will wait loading the next values to the double buffer until all counters with set sync bit are ready.<br>Activate the double buffer to use sync mode |
| 2 | disable double buffer<br>0: the double buffer is enabled<br>1: the double buffer is disabled. |
| 3 | non stop mode<br>0: if the double buffer is empty and all pulses are ready the pulse output will stop.<br>1: The pulse buffer is not cleared reading it. Therefore the values of the buffer will be reloaded after all pulses are done. The puls out generator will run non stop until 0 is written to No Of Pulses register or this bit is cleared. |
| 4 | IO Mode<br>0: The pulses generated with the puls generator are visible at the outputs.<br>1: The bit of Output register is visible at the outputs. |
| 5 | not used, (read only) |
| 6 | All Done (read only)<br>0: there are some pulses to output in the puls buffer or in the double buffer.<br>1: All pulses are done. The puls buffer and the double buffer are empty. |
| 7 | Latch Possible (read only)<br>0: The double buffer contains values wich are not read, yet. Writing a new value to the high byte of No Of Pulse register will overwrite these values.<br>1: The double buffer is read into the puls buffer and is ready to get new values |

**Note**: Writing to the read only bits will have no effect. Nevertheless please only write 0 to these bits to be compatible to future Versions of the TDR5000.

programming guide

## 6 Open Drain Outputs

The Open drain Outputs are controlled by the Pulse Generators as well as by the Open Drain Output register. Each bit of the Open Drain Output register controls one channel of the open drain outputs:

bit 0 controls channel 0,
bit 1 controls channel 1,
bit 2 controls channel 2,
...
bit 11 controls channel 11.

### 6.1 Open Drain Output with enabled pulse generator

If a pulse generator is enabled then a set bit of the Open Drain Output register will cause that the output state of the FET will be inverted.

### 6.2 Open Drain Output with disabled pulse generator

If a pulse generator is disabled then a set bit of the Open Drain Output register will cause that the FET will be in low impedance state. Clearing the bit will set the FET in High impedance state.

### 6.3 Used Register Adresses

| Register Address | Register |
|---|---|
| 17 dec.<br>11 hex | TDR5000_OD_OUT_LOW<br>Open Drain Output (low byte)<br>(R+W)<br>0: The FET is in high impedance state. If the Output is used by the pulse generator then the output of the pulse generator is not inverted<br>1: The FET is in low impedance state If the Output is used by the pulse generator then the output of the pulse generator is inverted. |
| 18 dec.<br>12 hex | TDR5000_OD_OUT_HIGH<br>Open Drain Output (high byte)<br>(R+W)<br>The bits 4...7 of the high byte are not used |

## 7 Opto Isolated Digital Inputs

The Opto Isolated Digital input can be read at Register Address 19

| Register Address | Register |
|---|---|
| 19 dec.<br>13 hex | TDR5000_OID_IN<br>Opto Isolated Digital Input<br>bit 0: Din 0<br>bit 1: Din 1<br>bit 2: Din 2<br>bit 3: Din 3<br>bit 4...bit 7: not used |

programming guide

## 8 Reading Encoder Inputs Directly

The digital states of the Encoder Inputs can be read directly by reading the following register addresses:

| Register Address | Register |
|---|---|
| 20 dec.<br>14 hex | TDR5000_ENC01_DIRECT<br>Encoder 0 and 1 direct<br>Encoder 0<br>bit 0: Ei_0_0<br>bit 1: Ei_90_0<br>bit 2: Ei_EN_0<br>bit 3: Ei_Ref_0<br><br>Encoder 1<br>bit 4: Ei_0_1<br>bit 5: Ei_90_1<br>bit 6: Ei_EN_1<br>bit 7: Ei_Ref_1 |
| 21 dec.<br>15 hex | TDR5000_ENC23_DIRECT<br>Encoder 2 and 3 direct<br>Encoder 2<br>bit 0: Ei_0_2<br>bit 1: Ei_90_2<br>bit 2: Ei_EN_2<br>bit 3: Ei_Ref_2<br><br>Encoder 3<br>bit 4: Ei_0_3<br>bit 5: Ei_90_3<br>bit 6: Ei_EN_3<br>bit 7: Ei_Ref_3 |
| 22 dec.<br>16 hex | TDR5000_ENC45_DIRECT<br>Encoder 4 and 5 direct<br>Encoder 4<br>bit 0: Ei_0_4<br>bit 1: Ei_90_4<br>bit 2: Ei_EN_4<br>bit 3: Ei_Ref_4<br><br>Encoder 5<br>bit 4: Ei_0_5<br>bit 5: Ei_90_5<br>bit 6: Ei_EN_5<br>bit 7: Ei_Ref_5 |

## 9 Analog Inputs

Sampling an analog signal is started by writing a command to the TDR5000_ADC_LOW register.

The converted value can be read 15µs later. To wait about 15us you can use the loop, endloop instructions:

```
loop 4   'wait until ADC is ready
endloop
```

### 9.1 Commands

| Bits | Description |
|---|---|
| 7...3 | 01010: Start sample 0...10V or 0..20mA<br>01000: Start sample 0...5V or 0..10mA |
| 2...0 | analog channel 0...7 |

### 9.2 Used Register Adresses

| Register Address | Register |
|---|---|
| 23 dec.<br>17 hex | TDR5000_ADC_LOW<br>write:<br>Writing a Command to the ADC<br>read:<br>reading the low byte of the result |
| 24 dec.<br>18 hex | TDR5000_ADC_HIGH<br>read only:<br>reading the high byte of the result<br>bit 4... bit 7 are not used |

programming guide

## 10 Analog Outputs

| Register Address | Register |
|---|---|
| 25 dec.<br>19 hex | TDR5000_DAC_CONTROL<br>write:<br>DAC control register<br>read:<br>DAC state |
| 26 dec.<br>1A hex | TDR5000_DAC_LOW<br>low byte of the output value<br>(write only)<br>Bits 0...Bit 3 is unused and should be set to 0 |
| 27 dec.<br>1B hex | TDR5000_DAC_HIGH<br>high byte of the output value.<br>(write only)<br>Writing to this register will cause sending the control word and the output value to the DAC |

### 10.1 DAC control register

| Bits | Description |
|---|---|
| 3...0 | Channel select<br>0000: AnOut0<br>0001: AnOut1<br>0010: AnOut2<br>0011: AnOut3<br>0100: all channels |
| 7...4 | command<br>0000: write to DAC register n<br>0001: update (power up) DAC register n<br>0010: write to DAC register n, update all<br>0011: write and update DAC register n<br>0100: power down n<br>1111: no operation |

### 10.2 DAC status register

| Bits | Description |
|---|---|
| 0 | busy<br>'1': The DAC is busy.<br>Don't write to the high byte of the output value.<br>'0': The DAC is ready to accept new values |
| 7...1 | not used |

programming guide

## 11 Serial Ports

The RS232 ports support handshake signals (CTS, RTS)
The inputs and outputs are buffered. Each buffer has an size of 2047 bytes.

Writing to a serial port is done by writing to the data register. All bytes will be buffered until they are send out. The number of free bytes in the send buffer can be read by reading the registers TDR5000_SERA_FREE_LOW and TDR5000_SERA_FREE_HIGH for channel A or TDR5000_SERB_FREE_LOW and TDR5000_SERB_FREE_HIGH for channel B

| Register Address | Register |
|---|---|
| 28 dec.<br>1C hex | TDR5000_SERA_CONTROL<br>write<br>Control for SER A<br>read<br>RS232 state of channel A |
| 29 dec.<br>1D hex | TDR5000_SERA_DATA<br>Read- and Write Data SER A<br>By reading this register you'll get the next value from the input buffer<br>By writing you will put the value to the output buffer. |
| 30 dec.<br>1E hex | TDR5000_SERA_FREE_LOW<br>number of free bytes in write buffer A (low byte)<br>read the low byte first and then the high byte |
| 31 dec.<br>1F hex | TDR5000_SERA_FREE_HIGH<br>number of free bytes in write buffer A (high byte)<br>read the low byte first and then the high byte |
| 32 dec.<br>20 hex | TDR5000_SERA_FILL_LOW<br>number of bytes in read buffer A (low byte)<br>read the low byte first and then the high byte |
| 33 dec.<br>21 hex | TDR5000_SERA_FILL_HIGH<br>number of bytes in read buffer A (high byte)<br>read the low byte first and then the high byte |
| 34 dec.<br>22 hex | TDR5000_SERB_CONTROL<br>write<br>Control for SER B<br>read<br>RS232 state of channel B |
| 35 dec.<br>23 hex | TDR5000_SERB_DATA<br>Read- and Write Data SER B<br>By reading this register you'll get the next value from the input buffer<br>By writing you will put the value to the output buffer. |
| 36 dec.<br>24 hex | TDR5000_SERB_FREE_LOW<br>number of free bytes in write buffer B (low byte)<br>read the low byte first and then the high byte |
| 37 dec.<br>25 hex | TDR5000_SERB_FREE_HIGH<br>number of free bytes in write buffer B (high byte)<br>read the low byte first and then the high byte |
| 38 dec.<br>26 hex | TDR5000_SERB_FILL_LOW<br>number of bytes in read buffer B (low byte)<br>read the low byte first and then the high byte |
| 39 dec.<br>27 hex | TDR5000_SERB_FILL_HIGH<br>number of bytes in read buffer B (high byte)<br>read the low byte first and then the high byte |

programming guide

### 11.1 RS232 Control register

| Bits | Description |
|---|---|
| 2..0 | select baud rate:<br>000: 2400<br>001: 4800<br>010: 9600<br>011: 19200<br>100: 38400<br>101: 57600<br>110: 115200<br>111: not used |
| 3 | '1': use handshake signals<br>'0': don't use handshake signals |
| 5..4 | no of data bits:<br>00: 7<br>01: 8<br>10: 9; transmit bit 9 = 0<br>11: 9; transmit bit 9 = 1 |
| 6 | enable parity bit:<br>0: no parity<br>1: parity bit is used |
| 7 | select even / odd parity:<br>0: even parity is used<br>1: odd parity is used |

### 11.2 RS232 state register

| Bits | Description |
|---|---|
| 0 | '1': output buffer is empty |
| 1 | '1': output buffer is full |
| 2 | '1': input buffer is empty |
| 3 | '1': input buffer is full |
| 4 | In 9 bit mode:<br>bit 9 of the next byte to read in the input buffer.<br><br>In 8 bit mode with parity enabled:<br>parity bit of the next byte to read in the input buffer.<br><br>Other modes:<br>undefined. |
| 5 | Error Bit<br>This bit is set, if a byte with a wrong parity bit is received.<br><br>It is cleared, if a byte with a correct parity bit is received. |

programming guide

## 12 Event registers

If an event occurs then it is stored in an event register. The event register is cleared automatically after reading it. So you can detect events which happened since the last reading of the register.

An event can also trigger an interrupt request in INTM1 to start an INTTASK if enabled. The corresponding bit of the Interrupt Mask Register must be set to enable the INTTASK.

It is necessary to acknowledge the interrupt request inside the INTTASK, otherwise no further interrupt request can be generated. To acknowledge an interrupt request write the value 2 to the RESET register.

Note: It's possible that more than one event occurs at the same time. So it's necessary to read and handle all event registers of all modules where you have enabled any interrupt.

If an event with enabled interrupt occurs while the INTTASK is pending, then the interrupt is latched and the INTTASK is started again after first INTTASK has finished.

| Register Address | Register |
|---|---|
| 40 dec.<br>28 hex | TDR5000_ENC_MATCH<br>Encoder Match Event<br>write:<br>Interrupt Mask register<br>read:<br>Encoder Event:<br>The bit is set if the corresponding counter value changes from a different value to the preset value.<br>Note: This can also happen, if the counter is set to the preset value due to an hard- or software reference signal. |
| 41 dec.<br>29 hex | TDR5000_PULSE_LATCH_POSS<br>Puls Out Latch possible<br>write:<br>Interrupt Mask register<br>read:<br>The bit is set if the double buffer of the puls generator can get a new value. |
| 42 dec.<br>2A hex | TDR5000_PULSE_READY<br>Puls Out ready<br>write:<br>Interrupt Mask register<br>read:<br>The bit is set if all pulses are out. |
| 43 dec.<br>2B hex | TDR5000_SERA_EVENT<br>RS232 Event for SER A<br>write:<br>Interrupt Mask register<br>read<br>The bit is set if the following event has happend:<br>bit 0: SER A: Send buffer is empty<br>bit 1: SER A: Send buffer is full<br>bit 2: SER A: Receive buffer changes from empty to not empty<br>bit 3: SER A: Receive buffer is full<br>bit 4: not used<br>bit 5: SER A: Parity error |
| 44 dec.<br>2C hex | TDR5000_SERB_EVENT<br>RS232 Event for SER B<br>write:<br>Interrupt Mask register<br>read<br>The bit is set if the following event has happend:<br>bit 0: SER B: Send buffer is empty<br>bit 1: SER B: Send buffer is full<br>bit 2: SER B: Receive buffer changes from empty to not empty<br>bit 3: SER B: Receive buffer is full<br>bit 4: not used<br>bit 5: SER B: Parity error |

## 13 Document History

| Document Version | Product Version | Description |
|---|---|---|
| V001 | V1.0 | first Version |
| V002 | V1.0 | Description of Single Counter Mode added.<br>Chapter “Registers and XPort Adresses” updated. |