

---

## Wireless Communication with IrDA

Gunther Zielosko

### 1. Basics of infrared communication

After the breakthrough of infrared remote controls in electronic devices, there had been endeavors at the beginning of the 90s to use this technology for bi-directionally communicating between devices. Although individual computers with an infrared interface already existed, there was no general standard. Finally in 1993 a committee of approximately 30 companies was formed, the Infrared Data Association (IrDA). In 1994 it created a standard for cross device infrared communication, which was named IrDA 1.0 or standard IR (SIR). In 1995 IrDA 1.1 followed, sometimes known as Fast IrDA or FIR. Both versions work with the same technology as far as the transceiver (a combination of IR-transmitter and receiver) is concerned; the communication however is completely different:

- IrDA 1.0 works asynchronous, analogue to the known RS232-interface (COM-port at the PC), with data rates up to 115.2 kbit/s. The transmission takes place at half duplex operation with a start bit, a stop bit and no parity bit. UARTs serve, which are also used for serial RS232 interfaces, as the hardware basis.
- IrDA 1.1 works synchronous (clock required) and allows 4 Mbit/s. The price for the higher speed is that no UART can be used here, but a special controller is required.

These fundamental definitions about the working method, such as baud rates, impulse lengths etc. form the so-called physical basis (physical layer). This aspect can be studied in more detail on:

[http://www.vishay.com/docs/irdc\\_physical\\_layer.pdf](http://www.vishay.com/docs/irdc_physical_layer.pdf)

If you, however, wish to communicate with other devices via infrared, it is necessary to take a look at the world of different software protocols. Similar to a cordless phone the individual devices have “addresses”. There are definitions which device will communicate first (Master / Slave), how the communication will take place etc. Finally, unlike wired data exchange, IrDA allows several partners to be present in the environment. Simply reversing the data direction by putting a specific level on one of the two lines is not possible. True IrDA operation simply requires more “discipline”.

The family of IrDA-compatible devices is large; the application of the available technology on the other hand is rather limited. Where do we find the IrDA interfaces?

- On almost all laptops (fully assembled plus existing driver)
- On almost all PDA's (Psion, Palm and Co.)
- On many mobile phones
- On many PC's (often only prepared as connector on the motherboard; only a IrDA transceiver chip is missing)

---

Is IrDA interesting for the BASIC-Tiger® and how is it possible to make the BASIC-Tiger® IrDA-compatible? What problems can occur?

Concerning to the first question – IrDA is wireless, electrically nearly undisturbed and can be used in “dangerous” surroundings. When a BASIC-Tiger® works e.g. in high voltage equipment (even 220V!), data can only be exchanged with high effort (Optocouplers, touch protection, electrical isolation etc.). Via IrDA this works without any problem over a distance exceeding 1 m, through insulated glass windows or transparent casting compound as well as electronic, chemical or even biologically critical space. Likewise it is possible to communicate with hermetically closed systems – it only requires a transparent window. Just think of operation under water! It is just as interesting that in contrast to other interfaces, connections can be made and dissolved in active operation. Although this is also propagated for newer wired interfaces such as USB, experience shows that it does not always work. A further aspect is the fact that many devices are already equipped with the IrDA-interface – so the BASIC-Tiger® could e.g. immediately start communicating with a mobile phone via IrDA.

Concerning to the second question – the relationship of IrDA 1.0 to the RS232 interface enables relatively simple solutions with the BASIC-Tiger®. In this application note we will therefore exclusively use IrDA 1.0.

Finally – what are the problems? The “true” IrDA communication runs according to a strict protocol. A standard IrDA device searches its surroundings for another IrDA device before the data transfer takes place. The functions are allocated (Host / Client), addresses are stated and much more. But basically it works – indeed we will not deal with real communication to any IrDA devices in this application note, but rather implement a data transfer analogous to the RS232 interface.

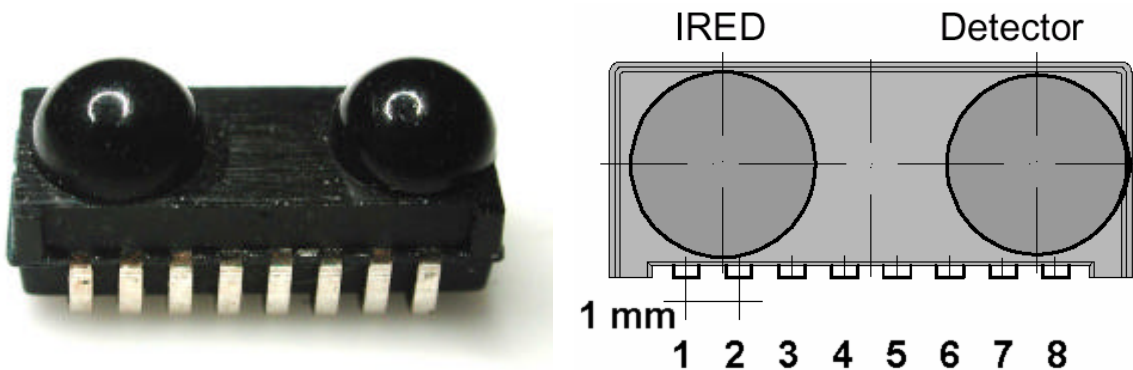
Although the signals of IrDA are related to those of the RS232-interface, they are not identical! With normal UART data, which is sent to the serial interface, “1” is represented by an impulse. With IrDA it is the other way around, a light impulse equals “0”. This inversion saves electricity, as a “high” applied to RS232 also is the idle state. The transmission of a byte only begins with the start bit, with IrDA this is a “low” level. Like this the IrDA interface only uses energy when it is really sending data, important for all battery operated devices. The standard pulse length for IrDA 1.0 is 3/16 of the “bit length” of a normal RS232 interface. At 115.2 kbit/s the length of an infrared pulse is 1.6 µs, at a transmission rate of 9600 bit/s it is about 20 µs. Therefore we need to transform the signals of the SER interface of our BASIC-Tiger® before the conversion into light signals.

## **2. A complete Transceiver-module, Vishay’s TFDU4100**

The Vishay Company manufactures different IrDA-Transceivers; we have chosen the TFDU4100, which combines an infrared transmitter and an infrared receiver in one case. A data sheet can be found on:

<http://www.vishay.com/document/89000/89000.pdf>

There are different case versions; ultimately it is not relevant for our usage which design we choose. The author has chosen the version shown in figure 1 and/or 2. An advantage in comparison with solutions with single components of this module is the fact that a gain control amplifier for sensitivity control has already been built in.



*Fig. 1 looks sexy: Vishay's TFDU4100 as Fig. 2 Pin connections of the TFDU4100 IR-Transceiver viewed from the front onto the lenses*

Despite its good looks, the TFDS4100 module cannot carry out the required impulse transformation from a regular RS232 signal all alone, for this an additional pulse shaper is needed, which is also offered by Vishay.

### 3. The coupled link for the SER interface, Vishay' TOIM3232

The TOIM3232 contains a programmable baud rate generator, with which different standard baud rates from 1200 to 115200 baud can be deduced from a quartz frequency of 3.6864 MHz. Furthermore it shortens the RS232 pulses to either the standard length of 3/16 of a RS232 pulse or to the power saving minimal value of 1.627  $\mu$ s, which can also be programmed. Unfortunately the TOIM3232 is no longer available. Although the replacement type TOIM4232 is identical regarding functionality and casing, it operates with the more "modern" supply voltage of 3.3 V. At the input the TOIM4232 can take 5 V levels and the Tiger also accepts the 3.3 V level from the TOIM4232. There is only the problem that an additional voltage regulator is required when connecting the TOIM4232 to the BASIC-Tiger<sup>®</sup>, in order to achieve stable 3.3 V.

One more thing: the TOIM3232 as well as the TOIM4232 do not work with real RS232 levels, but with TTL levels, disregarding their pin descriptions, on RS232 side. This is more of an advantage when using a BASIC-Tiger<sup>®</sup> without integrated RS232 level converters. If one wishes to use a real RS232-Tiger or the Plug-and-Play-Lab, it is necessary to interconnect a MAX232 once again. Datasheets for the TOIM4232 or TOIM3232 can be obtained on:

<http://www.vishay.com/document/82546/82546.pdf>  
<http://www.ida.ing.tu-bs.de/academics/labs/ist2/praktikumWS01/files/comports/toim3232.pdf>

Depending on the available version (TOIM4232 or TOIM3232) you must restrict the operation voltage to 3.3V, for example with a specific 3.3V regulator (TOIM4232). If you are able to get hold of a TOIM3232, simply connect it to the VCC of the BASIC-Tiger®.

As already mentioned, the TOIM3232 can be programmed and in this way adjusted to the applied case. Programming is done directly from the BASIC-Tiger® via the two logical connections RESET and BR/D (see the complete circuit diagram of our IrDA adapter, figures 3 and 4). The programming process is done at normal TTL levels as follows:

Step	RESET (Pin 1)	BR/D (Pin 2)	Description
1	High	X	Reset all registers and set the data rate to 9600 bits/s, IrDA pulse width to 1.627 µs, power saving mode
2	Low	X	End reset, then wait at least 7 µs
3	Low	High	Activate programming mode, wait at least 7 µs
4	Low	High	Now send the control word YZ* (1 byte) serially to pin 3, RD_232 (Programming data see tables 2 and 3) <ul style="list-style-type: none"> <li>• After RESET data rate 9600 bit/s, IRDA pulse width 1.267 µs</li> <li>• On programming repetition most recent parameters wait at least 1 µs</li> </ul>
5	Low	Low	Activate data communication mode. RESET and BR/D both stay low during data transfer. The following data transfer must from now on be handled with the recently set parameters! A new change of the interface parameters can be done by following steps from 3 onwards.

*Table 1 Programming sequence of the TOIM3232*

The Y in the control byte YZ contains the higher 4 bits with the pulse length S0 (Bit 4) as well as the values for the programmable user outputs S1 and S2 (Bits 5 and 6) in accordance with table 2. The lower 4 bits (Z) determine the data rate (table 3).

Bit 7	Bit 6 (S2)*	Bit 5 (S1)*	Bit 4 (S0)	1 <sup>st</sup> character	IrDA pulse length
X	X	X	0	0	3/16 bit length
X	X	X	1	1	1.627 µs

\* Within the programming the outputs S1 and S2 (Pins 12 and 13) can be randomly set to Low or High by the user. In our circuit we use S2 to change the sensitivity (SC = Sensitivity Control) of the TFDU4100.

*Table 2 TOIM3232 – The interpretation of the 1st programming character Y (pulse length)*

Bit 3	Bit 2	Bit 1	Bit 0	2 <sup>nd</sup> character	Baud rate
0	0	0	0	0	115,200
0	0	0	1	1	57,600
0	0	1	0	2	38,400
0	0	1	1	3	19,200
0	1	0	0	4	14,400
0	1	0	1	5	12,800
0	1	1	0	6	9,600
0	1	1	1	7	7,200
1	0	0	0	8	4,800
1	0	0	1	9	3,600
1	0	1	0	A	2,400
1	0	1	1	B	1,800
1	1	0	0	C	1,200

Table 3 TOIM3232 –the interpretation of the 2nd programming character Z (Baud rate)

**Example:**

The data byte YZ in binary form is: **11010110b**, which is **D6h** hexadecimal or **214** decimal.  
This means:

- S2 is statically set to 1
- S1 is statically set to 0
- From now on the pulse width is 1.627  $\mu$ s
- From now on the data rate is 9600 bit/s (Baud)

Now we know almost everything about both IC's with which we want to make our BASIC-Tiger® IrDA-compatible. In the following section we will deal with the connection to the BASIC-Tiger®.

**4. Circuit BASIC-Tiger® / IrDA interface**

On the hardware side there is not much to be done, we connect to the corresponding serial lines of the BASIC-Tigers® first the TOIM3232 and then the TFDU4100 (Figure 3). The presented circuit applies to the Tiger without built in RS232 level converter. When using Tigers with integrated RS232- level converters or the Plug-and-Play-Lab a MAX232 must be inserted between the Tiger and the rest of the circuit (Figure 4)!

The already described characteristic of the TOIM4232 with its operation voltage of 3.3 V still requires its own regulator, which will conduct this voltage from the VCC of the BASIC-Tiger®. For the sake of simplicity we have designed the circuit for a TOIM3232, which does not require this. Nevertheless quite a lot must be considered with regard to the voltage supply. The operation voltage of the TFDU4100 should be well buffered via a filter section  $47\Omega$  /

100nF, 4.7µF. As far as electrolytic capacitors are concerned only tantalum capacitors are worth considering. Equally critical is the supply of the sender diode at the pin IR-anode. Due to the considerable pulse currents that arise there (0.2 A!), the supply should actually take place via a second, independent 5V regulator. If you should still wish to take this voltage from the “normal” VCC, it is first important to ensure the short-term running pulse current via a RC-combination; secondly it must be guaranteed that the supply of the BASIC-Tiger<sup>®</sup> does not occasionally “bend its knees” while doing so. The suggested combination is a compromise (see piece of advice at the end of this application note!)

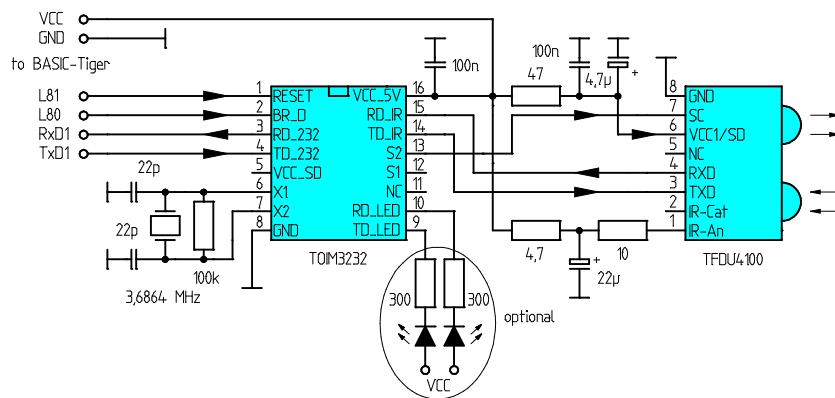


Fig. 3 Circuit of the simple IrDA-Adapter for BASIC-Tiger<sup>®</sup> without RS232 interface

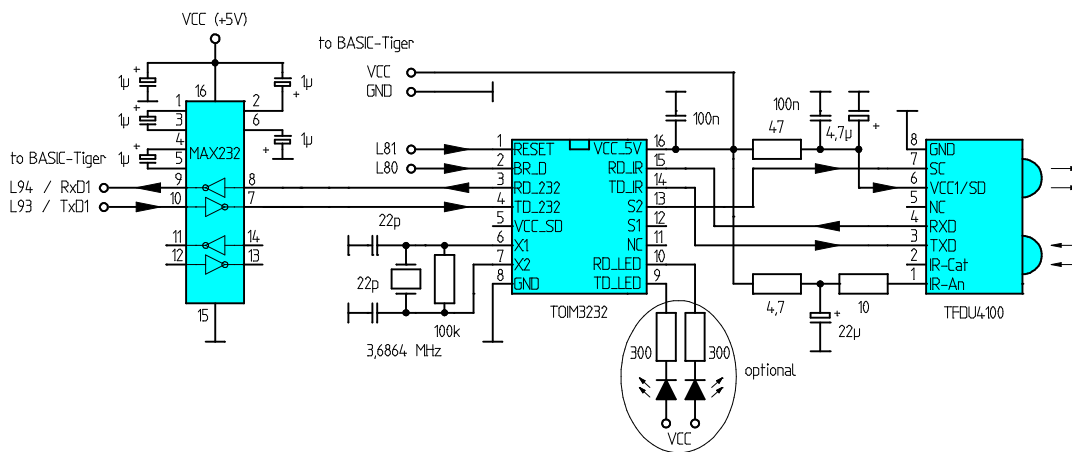


Fig. 4 The same for BASIC-Tiger<sup>®</sup> with built-in RS232 interface

The TOIM4232/3232 as well as the TFDU4100 have a SMD casing (the TOIM4232/3232 a particularly wide one, the TFDU4100 a specific one with 1 mm screen); this is the reason why the construction of the circuit requires a little time. Either one is satisfied with a “wire entanglement” or one manufactures a professional printed circuit board (e.g. by photographic means). As an idea here is the layout of the printed circuit board, manufactured by the author (Figure 5) and the finished adapter (Figure 6).

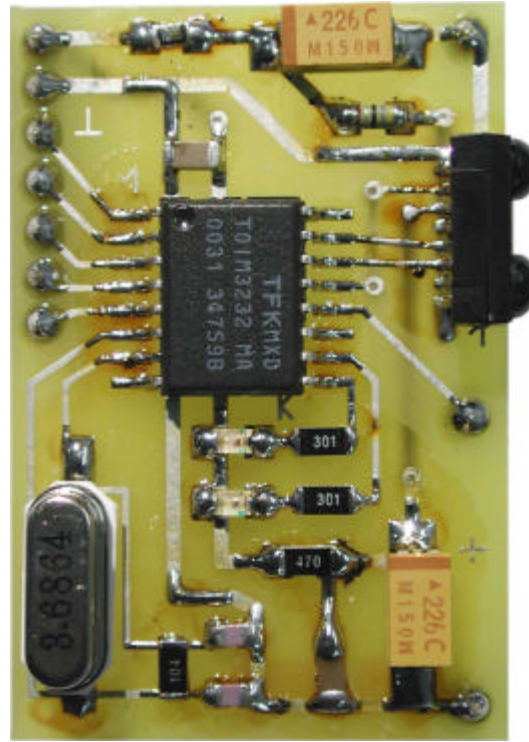
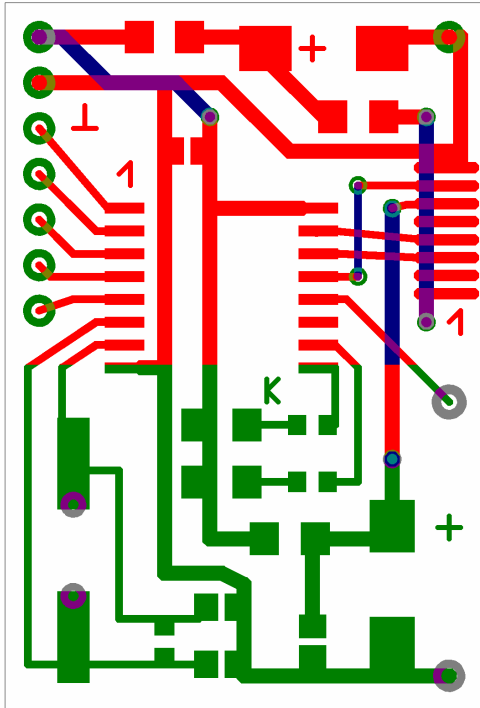


Fig. 5 Layout of the IrDA adapter (red BE- Side, blue conductor side) Fig. 6 fully equipped IrDA adapter

For initial testing of the adapter switch the module successively as follows:

1. Idle current
 

VCC	to + 5 V	no LED should light up, the power consumption of the circuit should be a maximum of ca. 5mA
GND	to GND	
RESET	to + 5 V	
BR_D	to GND	
TD_232	to GND	
  
2. Active sending (only differences to 1. listed here!)
 

RESET	to GND	TD_LED should now light up; power consumption is now ca. 25mA
-------	--------	---
  
3. Change TD\_232 (only differences to 2. listed here!)
 

TD_232	to + 5 V	TD_LED off
TD_232	to GND	TD_LED on
  
4. Test reception (only differences to 3. listed here!)
 

TD_232	to+ 5 V	TD_LED off
IrDA signal connected from outside		RD_LED blinking

---

If everything has worked up until now, the odds are looking pretty good for your IrDA module.

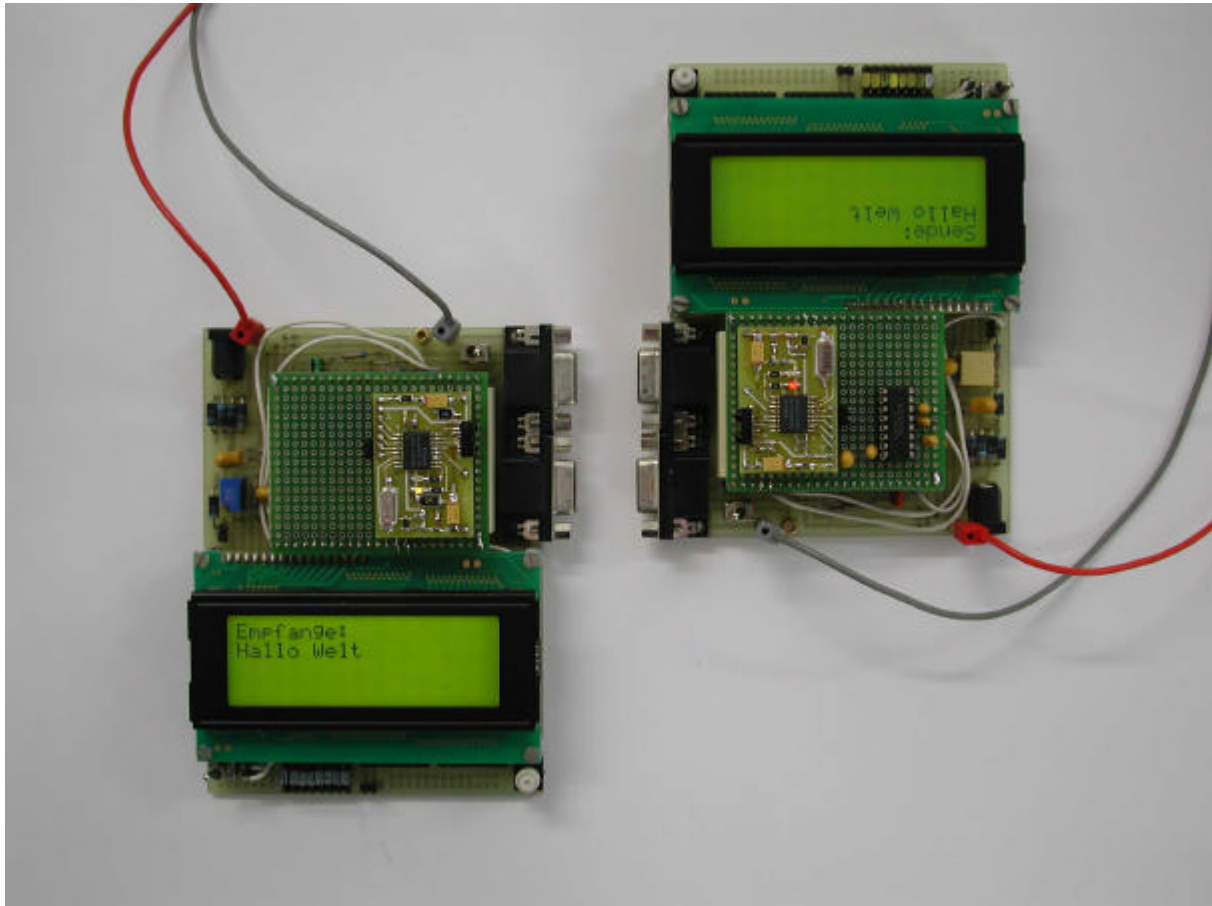
So how is it possible to connect other IrDA devices? There are 2 fundamental ways:

- You either buy a standard IrDA adapter or you build your own for the serial interface (e.g. for the PC). Like this the data transfer will take place almost exactly as you would expect it of a RS232. With a terminal program or other suitable software a data transfer from or to the BASIC-Tiger® should be possible.
- You walk the stony path of the “real” IrDA communication and write programs for the BASIC-Tiger®, which will then at least communicate according to protocol with IrDA-1.0-devices.

## **5. Software**

Do you require software actually for data transfer on the “lower” level? Actually the data transfer really takes place totally identical to the usual RS232 interface. Nevertheless we need a few additional BASIC lines – besides the installation of the serial device driver (initially with the standard baud rate of 9600 baud) we at least need to initialize the TOIM3232 and then set the new baud rate, which the “remote device” understands (for example 19200 baud). This baud rate must now be set in the device driver.

Again, as remote device we are using an IrDA compatible BASIC-Tiger®, because here we have access to all parameters of the interface and all possibilities of manipulation. The program IRDA01.TIG shows exemplary how the adaptation of the baud rate and the setting of other IrDA parameters can take place. Here ASCII characters are sent and received without considering any protocols. The differentiation is done in the last lines (Replace commentary symbols!), so that it is possible to setup one BASIC-Tiger® as sender and another as receiver. Figure 7 shows how 2 such systems, here in a mini lab and with plugged-on IrDA adapter, work. Initial experiments showed a range of 3m and more without any effort, even with low sensitivity (S2 to “0”).



*Fig. 7 Two BASIC-Tigers<sup>®</sup>, each in a mini lab, communicate via IrDA (Receiver on the left – yellow LED, sender on the right – red LED). The right-hand device has a BASIC-Tiger<sup>®</sup> with built-in RS232 interface and therefore requires a MAX232 for “level reconversion”.*

**Another important note:**

**We already know that our IrDA sender requires a considerable amount of power when active. The most unfavorable case for power requirement is when the baud rate is extremely low (e.g. 1,200 baud) and the IrDA pulse length is set to 3/16 of the bit length at the same time. It is then possible that, due to the chosen circuit (Supply voltage of the IrDA-adapter from the Tiger’s VCC), this VCC crashes below its minimum value. This causes a RESET of the BASIC-Tiger, perhaps even worse! If very slow data rates are really required, the pulse length should be set to 1.627  $\mu$ s (minimal pulse length). If this is not possible either, the adapter circuit must have its own efficient voltage regulator!**

Are you interested?

If you are, there are a number of possibilities waiting for us... Good luck!