BASIC-Tiger® sends SMS

Gunther Zielosko

1. Entering the modern world of mobile phones

Almost everyone today owns a mobile phone; some even have more than one. Many telephone services offer a new phone mostly free of charge in exchange for the old one once the contract has expired, so that a number of old mobile phones a lying seemingly useless in people's cupboards. Those of you that are familiar with the BASIC-Tiger® can use such mobiles for far more than simply telephoning. Why should the BASIC-Tiger® not be able to use the mobile as a sort of emergency switchboard – compared to other radio systems this would work over any distance you like, even worldwide!
A precondition for our plan is that mobiles must be capable of data transmission, i.e. they must be able to send an SMS (Short Message Service) and have a data connection to the outside world. Many mobiles even have two such connectors, enabling them to communicate via an infrared interface on the one hand and with a so-called data cable via RS232-interface on the other hand. The remote device is usually a PC or a PDA. It is possible with these connections to transfer telephone numbers quickly and straightforward from the PC to the phone, to dial automatically, to query the status of the telephone and much more. If the BASIC-Tiger® should control the telephone instead of the PC, the connection should be established via the data cable.
In this application note we want to find out a little more about the technology of such mobile phones, to discuss their coupling with the BASIC-Tiger® and to try out an important application using this combination:

Following the occurrence of certain conditions (open or closed contact, measured value above the limit, battery voltage too low, etc.) the BASIC-Tiger® should automatically "call" a specified number with the mobile and send an SMS there (alarm system principle).

Naturally not all mobile phones and their interfaces are the same, which is why the overall solution presented here is not always usable without certain modifications. All details in this application refer to the classical and somewhat older Siemens mobile S25, the manufacturer of which has standardized its data coupling over a longer period of time allowing later models to make use of this system. In comparison it will be difficult if your mobile is only able to communicate via infrared or USB.

2. The Siemens-mobile phone S25

The S25 has gotten on in years, though the technology and equipment was at its peak for the time of its development. It is almost ideal for our application. Figure 1 shows the valuable

piece in its entirety and figure 2 the most important detail for us – the data interface through a plug.

Fig. 1 The Siemens S25 as a gate to the SMS world

Fig. 2 Connection of data cable to Siemens S25

The technical details of the S25 are only important to us within the framework of the planned coupling with the BASIC-Tiger®. Therefore we have here initially the pin assignment of the 12-pole socket, which is accessible on the front face of the device (Table 1). The pins 1 (GND), 5 (DATA OUT) and 6 (DATA IN) specifically serve for data communication of the S25. It is important to know that the data traffic normally runs according to the regularities of the RS232-interface (19200 baud, 8 data bits, no parity, 1 stop bit, no handshake). But modified levels are used here (0 V and 3,3 V), with the consequence that for a data transfer to a real RS232-interface, i.e. on a PC, firstly always a level converter (such as a MAX232) must be interposed and secondly the signal cable that leads to the mobile phone must be restricted to ca. +3 V. The standard data cable for the S25 automatically realizes this; it has the complete electronics built into its 9 pole Sub-D-plug. In order to complicate matters, here directly the hint that the level converter that is built into the data cable is supplied with current from the RS232-interface of the PC. This current supply through not needed signal lines of the RS232-interface is for example also common practice for the computer mouse. Unfortunately the RS232-interfaces of the BASIC-Tigers® (both SER0 and SER1) are not compatible in this connection, so that “special conditions” become necessary.

Pin	Description	Function	Comments	Type
1	GND	Telephone-Mass		
2	SB	Recognition or rather control for the battery charger	LOW = 150mA-battery charger HIGH = 1A-battery charger NC = 400mA-charger at 35/45	in / out
	AUD01	SL45: Audio channel 1		out
3	CHARGE	Input charger voltage	U = 6.1V – 8.0V (to max. 1 A)	in
4	BATT	Voltage supply for accessories	U = 3,0V – 3.9V Umin = 2.6V Imax = 100mA	out
5	DATA OUT	Data outgoing	PullUp built in telephone	out
6	DATA IN	Data arriving	PullDown designated in end device	in
7	Z_CLK	Clock line for accessory bus		
8	Z_DATA	Data line for accessory bus		
9	MICG	CTS	Only used in data mode	
10	MIC	GND for microphone		
11	AUD	Microphone	U = 1.5Vpp	in
	AUD02	Speaker	U = 1.5Vpp	out
	AUDG	SL45: Audio channel 2		
12	AUDG	4xxx: GND for speaker		
	AUD	GND for speaker		
		4xxx: speaker/DC offset	Signal is not uncoupled!	

Tab. 1 Pin designation of the Siemens connector (Colors in the home made cable of the author, only the four colored lines are used for our project)

3. Hardware

3.1. Coupling possibilities Tiger – Mobile phone S25

Depending on the initial situation there are different solutions for the coupling of the S25 with the BASIC-Tiger®:

- You want to use the Plug-and-Play-Lab. This generally has “true” RS232-interfaces SER0 and SER1 regardless of the used Tiger module type. The 9-pole SUB-D-connectors (jacks) are also there. You therefore use the original data cable for the S25. You then need a “male / male” adapter for the connection of the two “female” connectors on the Plug-and-Play-Lab and the data cable. Additionally you connect the unused pin 4 of a 9-pole SUB-D-plug of adapter from the Plug-and-Play-Lab via 10 Ω with +5 V (VCC) to supply the cable electronics.

- You wish to have a stand-alone solution. Your Tiger comes from a “better family” and has a built in RS232-interface. Its levels for communication are at ca. +10 V and -10 V respectively. In this case you can use the standard data cable of the S25 here as well. This cable has a 9 pole SUB-D-jack for direct coupling with the RS232-interface of the PC. We therefore sensibly use a 9-pole SUB-D-plug on the Tiger-side for SER0 with an additional power supply connection as mentioned above (Figure 3).

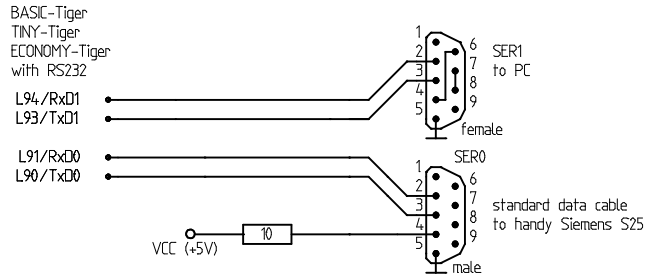

Fig. 3 Solution for Tiger with built in RS232-interface

- You want to have a stand-alone construction using a Tiger without RS232-interface, but still use the standard data cable for the S25. You then have to build in a MAX232 as level converter into your device (Figure 4).

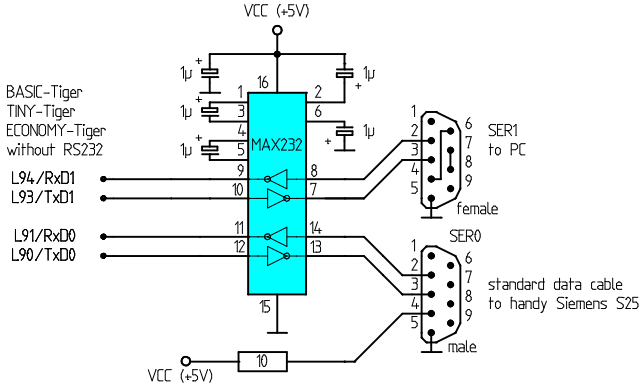

Fig. 4 Solution with topped level converter for Tigers without RS232-interface

- If you want to use a Tiger without built in RS232-level converter (the cheapest solution!), you can also spare yourself the expensive data cable with its internal electronics. You put the TTL-levels of the SER0 directly on the mobile phone, but have to make sure that the outgoing signal is restricted to ca. 3 V (Figure 5). The problem with this solution is simply the plug on the side of the mobile phone. It costs quite a lot and you must carry out filigree solder work. If you are lucky you can purchase such a plug in an accessory shop. Another solution is to remove a plug from no longer needed accessory parts – the simple headset with earphones and microphone for example includes a fully equipped plug for soldering and is usually even cheaper than a simple plug.

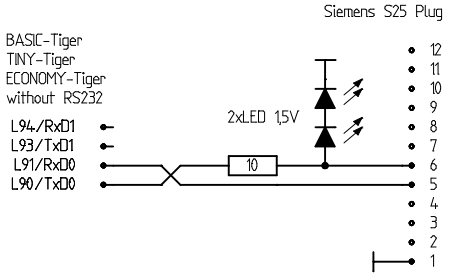

Fig. 5 Circuit for Tiger without RS232 directly to the S25-plug

- You wish to save space and costs and use an Economy-Tiger without built in RS232-interface. Despite the limited pins the functionality is sufficient for the planned purpose. The author favors this solution, the circuit and sample programs fit in with it (Figure 6).

3.2. The circuit

To operate the alarm system we use a mini circuit with Economy-Tiger. The programming of the controller takes place in the Plug-and-Play-Lab via the two adapters from Wilke Technology (BASIC-Tiger® to Tiny-Tiger® and Tiny-Tiger® to Economy-Tiger®). The serial interface SER0 operates the phone with TTL-levels. The connection takes place via a 4-pole plug and a self-made direct cable without any electronics. Port 6, as well as the two control lines L36 and L37, can be used for the connection of a (optional) display. The program HANDY_01.TIG supports the presentation via SER1 and Terminal program as well as via the LC display, if required both must be realized on the hardware side. On the other hand, if you do not need a display, you can make use of the multifunctional pins as analog inputs – useful for different warning installations with analog measurement data. We use the I/O pins L80 – L84 and possibly L85 to select different options or rather to send switching signals. The complete circuit is shown in figure 6. Even the inner storage battery can be charged via pin 3 of the mobile phone plug from our VCC through a 10 Ω resistor, the proper regulation is done in the phone itself. The system is now ready to run in continuous operation.

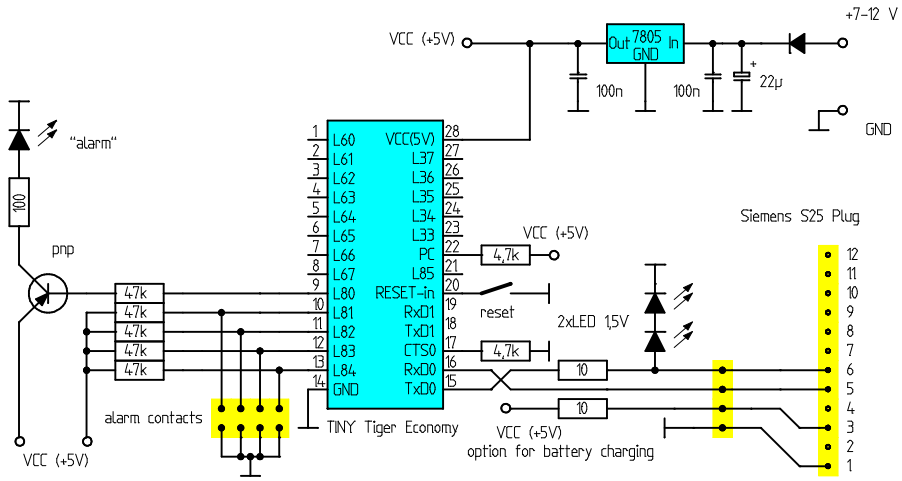

Fig. 6 Complete circuit with Tiny-Tiger® Economy and minimal effort

Both LEDs with 1.5 V forward voltage each (measure and combine with normal diodes if necessary!) serve to limit the level of signals going out to the mobile phone to ca. 3 V. Of course you can also use suitable Z-Diodes or similar here. The lines L81 – L84 are connected to VCC through pull-up resistors and are used as inputs, pulled down to Low they will set off the alarm. Pin L80 is used as an output and switches an LED via a transistor (signals an occurred case of alarm continuously).

4. Software

In order for you to check your circuit straight away, with HANDY_01.TIG we deliver a program which operates several functions of the phone from the BASIC-Tiger® and calls up all kinds of data. But before that a chapter about the communication with a mobile phone.

4.1. The command language of mobile phones

Almost all devices, which have a data interface, either per wire or per infrared, are responsive to the so-called AT-commands for GSM-devices. A quality of these commands is that they can be sent to the mobile phone as normal ASCII-character strings with a prefixed “AT”. There are commands, which are universally applicable, and such that are valid for a special device or device family. Some commands cause something to happen on the mobile phone, others deliver information from the mobile back to the PC or to our Tiger. From the multitude of possible commands only a few select ones will be named here, those who wish to know more can find them at

http://www.nobbi.com/phones.htm

in the internet.
Other than that the use of a terminal program is highly recommendable for first examinations. Following the selection of the interface and its parameters (19200 baud, 8 data bits, 1 stop bit, no parity, no protocol or Xon/Xoff) you are able to enter commands directly here and watch the reaction of the phone. Do not forget to hit <Enter> after each command!

Note:

We have generally laid out our programs in such a way that the data transfer with the mobile phone runs on SER0. SER1 and the LC display serve as an option for observing the data transfer (“Questions” and “Answers” are additionally displayed here). For this observation there must be a provision that SER1 is equipped with the appropriate levels for the PC-connection and/or that a LC display is connected. Once you are certain that the mobile phone control is working you can certainly leave out the hard-and software for monitoring via SER1 as well as LC display if you wish to (in which case the minimal version according to the circuit in figure 6 is sufficient).
Following a short line up of typical AT-commands (commands with blue background change something on the telephone, the others only request something!):

Command	Function	Example of an answer	
atd...;	Dials a number Valid chars for the dial string are: 1234567890*#+ABC. The semicolon directly generates a voice connection.		
at+chup	Separates an existing connection		
at^sms0	Switches off telephone		
at^srtc	Switches ringing on or off (command again)		
atd+cbc	Querying the battery condition (here e.g. 0=Battery, 80%	+CBC: 0,80	
at+cclk?	Query time and date (14.11.2002, 14:23:00 hours)	+cclk: "02/11/14,14:23:00"	
at+cgmi	Queries manufacturer (here Siemens)	SIEMENS	
at+cgmm	Model (here S25)	S25	
at+cgsn	Serial number		
at^scid	Issue SIM-serial number		
at^srtc=?	Show possible ring tones (here Nos. 0-42, volume 0-4)	^SRTC: (0-42),(0-4)	
at+cmgs=x	Sends a SMS. The data of the SMS must be handed over in PDU format. x indicates the length of the handed over data of the PDU in which the (optional) data of the SMSC to be used is not counted. Once the command is sent the telephone answers with a prompt '>', after which the PDU can be handed over to the phone. The PDU is ended with CTRL-Z.		
at+cmgl=4	Puts out all SMS saved in the device to the terminal		

Many commands allow for inquiries (sometimes with a question mark), with some parameters these commands set the respective parameters or change telephone settings:

at^srtc switches the ringing on and (when repeated) off
at^srtc=? shows available ring tone settings. Answer: ^SRTC: (0-42),(0-4) for S25
at^srtc=a,b sets one of the possible ring tones
a=0 no tone
a=1 first melody
a=2 second melody
a=42 forty-second melody

b=0 slowly increasing volume
b=1-4 different volumes

4.2. The mobile phone test program “HANDY_01.TIG”

With the help of this simple program we will now extend our knowledge of the mobile phone command language. We will also test, whether our hardware and software settings (i.e. terminal program etc.) are working. If a Siemens mobile phone S25 or similar is connected to the BASIC-Tiger®, a lot should happen on the mobile phone after the program start, i.e. it should start ringing and then switch it off again. The current sequence of events can be displayed via SER1 and a terminal program and/or via a connected LC display if one of the two is connected. The chosen example AT-commands are at no expense, meaning we of course do not dial a number. Nevertheless you should be careful in the experimenting phase and closely observe the mobile, its settings and possible deviations from its normal behavior.

4.3. How to create a SMS from a simple telephone number and a plain text

The example program SMS_01.TIG uses the mobile phone to send a message for the first time. For making any sense of it we design SMS_01.TIG as a small alarm system. When closing any of four different contacts it should send another SMS to another telephone. To test this you first need to give the string N$ a real number in the program:

With N$ you determine the number which the alarm-mobile should dial. For the first few attempts you must have this (second) mobile close by and switched on, so that you can control the success or failure immediately. If need be you can of course send a SMS to your own phone. Generally please choose the international spelling in the following form (fill in all chars without a space in the program!):

+49xxxyyyyyyyy	with +49 for Germany with xxx for the mobile net or rather the area code without the leading 0 (e.g. 170) with yyyyyyyy for the actual calling number.

T$ determines the SMS text to be sent. You may use all printable ASCII characters in your text. In the program “SMS_01.TIG” T$ is preset with “Alarm vom Handy” (“Alarm from mobile”) and with text parts that are created by the program itself.

If you now believe that the BASIC-Tiger® program simply puts both of these messages together, you are wrong. The life of a SMS is far more complicated. Here now a few explanations, which will surely not be adequate, those that wish to know more can get more advice and support from the internet link found on page 6. The PC program PDUspy can also be useful. It can be used to write a SMS on the PC and study the all the details of number and text transformation. The most important are briefly listed here:

There are at least two ways of proceeding when sending SMS,

- Text mode — is only mastered by few mobile phones, but would be a godsend for our project!)
- PDU mode — all SMS-able mobile phones are able to do this, but it requires an incredible large effort to use it.

In order to be universal we must choose the PDU mode, whether we want to or not. That means the number to be dialed as well as the simple text must be reworked in complicated way. What conversions are we talking about?

The Text:

Our example text is “Alarm vom Handy”. You certainly know that such text are usually stored or sent as ASCII characters. With this

Alarm vom Handy

is first converted to the hexadecimal character sequence

41 6C 61 72 6D 20 76 6F 6D 20 48 41 6E 64 79 (all in hex!)

Each one of these characters normally has 8 bits, of which only the lower 7 are used by pure ASCII characters, more or less as follows:

"A" = 41h = 0100 0001
"l" = 6Ch = 0110 1100
"a" = 61h = 0110 0001
etc. until

"n" = 6Eh = 0110 1110
"d" = 64h = 0110 0100
"y" = 79h = 0111 1001

When these characters are written or sent after each other, the following bit chain is created:

"A"	"l"	"a"		"d"	"y"
0100 0001	0110 1100	0110 0001	....	0110 0100	0111 1001

The PDU inventors likely wanted to save this unused 7th bit and have therefore simply removed it. Hence all characters are shortened by one bit, as follows:

"A" = 41h = 100 0001
"l" = 6Ch = 110 1100
"a" = 61h = 110 0001
etc. until

"n" = 6Eh = 110 1110
"d" = 64h = 110 0100
"y" = 79h = 111 1001

Here too these 7 bit long bit chains are fit together again, so that the following arises at the end of our chain:

	"n"	"d"	"y"
....	110 1110	110 0100	111 1001

Now imagine making 8-bit characters from this again (simply arrange the gaps differently!):

.....	1 1011	1011 0010	0111 1001
		B2h	79h

That would be completely different hex-data. Besides, so many zeros are filled up at the beginning until the front byte is filled containing also 8 bit. Just for harassment the new bytes are now also distributed in reverse order one after the other. Those who wish to run through all of this once can compare the result with the yellow string (that should come out!).

41 6C 61 72 6D 20 76 6F 6D 20 48 41 6E 64 79 becomes

```
41 76 58 DE 06 D9 DF 6D 10 32 EC 26 E7 01
```

This is now the new text message, which our mobile phone should send.

The telephone numbers to be dialed (Target no.)

Even the simple telephone number of the device to which the SMS should be sent must be converted first.

We have agreed to use the international format for the target number, which could look as follows (only an example, please do not call the poor person at this number!!):

+49 160 12345678	**+49**	Prefix for Germany
	160	Net prefix (here T D1)
	12345678	the actual calling number

Unfortunately here again there has been worked rather mysteriously. The character "+" is converted to 91h (Why?), 81h would be used e.g. for national numbers or numbers without a prefix (Service numbers). The other digits are painstakingly swapped in blocks of two, starting from the beginning (Again why?). Finally, in the case of the last digit having no partner to swap, an **F** is inserted and then the last digits are also swapped. Now our number looks as follows:

```
+  49 16 01 23 45 67 8F
↓   X  X  X  X  X  X  X   (X stands for swapping of both digits)
91 94 61 10 32 54 76 F8
```

Isn't that brilliant?

Aside from encoded **telephone number** and modified **text** a "real" SMS also requires two **length specifications** (Original number without "+" - here 0Dh = 13, original text – here 0Fh = 15) as well as a few **identifiers** and **flags**, which we will not further dwell on but build into our PDU-text as constants. All this will then be put together properly and sent as PDU string Z$.

```
000100 0D 91946110325476F8 0000 0F 417658DE06D9DF6D1032EC26E701
```

In case of an alarm the program generates a complete PDU string Z$, which is sent to the mobile phone after some preparations.

- The AT-command "at+cmgs=x" (x is the decimal length of the text part of the PDU string) is sent to the mobile phone (delete spaces in the program!)
- The mobile responds with a prompt ">" (we ignore it, but instead wait long enough!)
- Following that the prepared PDU string Z$ is sent to the mobile phone.
- Finally our program sends a "CTRL Z" ("<1Ah>")

If you wish to go deeper into the world of PDU, you should take a look at the web sites of

http://www.nobbi.com

There you can find help and information for all questions on mobile phones, telephone nets, hard- and software etc. You can also find the PC program PDUspy at

http://www.nobbi.com/download/pduspy.zip

with which you can immediately check the results of your own SMS programming on the BASIC-Tiger® professionally.

Practical Tips

The program SMS_01.TIG is built up in such a way that in principle you can experiment without a mobile phone. For this you only have to connect a terminal program to SER1 (with 19200 baud) and can see and possibly check all reactions (alarms, the individual steps of the text-string conversion, number-encoding and length calculations of single parts). Do not forget the RS232-levels, which you need for the PC! You can also see and print the messages, which are ultimately sent to the mobile phone. You can also test complicated facts with the PC program PDUspy. Only when all seems to be working, the mobile phone should be connected; otherwise several tries may cost a lot of money...

Before experimenting with the remote control of your “Alarm mobile phone” you should first check that you are able to send a SMS “by hand” from this “Alarm mobile phone” to the chosen second mobile phone, which should receive your message. Only when this works and all settings of both mobile phones allow trouble-free SMS transfer, you should employ the BASIC-Tiger®. You must now enter the target number and, if needed, the changed SMS text in the program. In order for the TINY-Tiger® Economy to really become the alarm centre, you must obviously establish alarm conditions, e.g. when one of the pins L81 – L84 of the Tiger is switched to ground. This is provided in the program SMS_01.TIG! The Tiger only calls for help, when you put such a pin to ground. If that works, you obviously have all the freedom to program alarm conditions yourselves...

You now have a data capable mobile phone, an additional module for automatic sending and receiving of SMS as well as the suitable software for the BASIC-Tiger®. To put the system into operation, you obviously need access to a mobile network. You could already try this out with your “normal” SIM card. In the long run this is not very helpful – as you need your mobile phone for telephoning. So for continuous operation with the automatic SMS system you need to register the “data mobile phone”. As this is really used only in exceptional cases (alarm), a prepaid contract is recommended here with a mobile network. Apart from the installation costs you only pay the costs for sending a SMS from the alarm system. In the other case (as a remote control installation) there are only costs for the sending telephone (exception international calls!). But you have to take care that the amount on the prepaid card must be topped up if need be. Furthermore these cards usually have a limited term – you must

see that the card is extended from time to time! Incidentally you can accept calls and SMS in most cases even when the card is empty. Your network operator will be able to tell you how long this is possible.

Outlook

The topic SMS from the BASIC-Tiger® via mobile phone has not been that simple and still not been fully dealt with. Nevertheless there are enormous possibilities already now:

- Complicated alarm systems with different SMS texts and/or telephone numbers
- Telemetry data, e.g. sent via SMS
- Surveillance of small children or helpless people and sending of SMS in critical situations

We have not yet spoken of the other direction, e.g. to cause the BASIC-Tiger® to take some action via SMS (depending on the SMS text). There is another opportunity for a new application note...

Best of luck with your experiments with Tiger and mobile phone!