TOIM3000/ 3232
Design Notes

Table of Contents

IrDA Port and Serial Port Time Multiplexing

The TOIM3xxx is a low-cost device that allows the systems designer to integrate IrDA capability into a notebook computer using the UART NSC16550/16450 or equivalent. At one end, the TOIM3xxx interfaces directly with an IrDA port and a serial RS232 port, while at the other end, the 16550. Figure 2 provides an example of a how to time multiplex the 16550 to an IrDA port and a serial RS232 port:

The OUT1 signal from the NSC16550 plays the role of selecting either the IrDA port or the RS232 port. When RESET=0, the TOIM3xxx communicates with the IrDA port, while data communication from/to the RS232 port is blocked. On the other hand, when RESET=1, the TOIM3xxx communicates with the RS232 port while data communication from/to the IrDA port is blocked.

Please note that the TOIM3xxx always operates in half-duplex mode, as specified by the IrDA standard. That is, whenever the TOIM3xxx transmits, the receiver is blocked and is inactive, and vice versa.

Operation Description

Features only for TOIM3000

The TOIM3000 uses two clocks from the UART: the 1.8432 MHz clock and the Baud_out clock for its internal timing. Both are connected to XIN and B_CLK, respectively. The B_CLK is used as a reference for pulse stretching whereas XIN is used as a time base for pulse shortening to 1.627 µs and noise filtering.

Single clock operation

TOIM3000 can be operated with only a single clock. In this case, the B_CLK and XIN are tight together and connected to the Baudout pin of the UART. The pulse width is then shortened to 3/16 of the bit length and noise filtering is deactivated. S1 is to be connected to V_{CC} and S0 to GND.

We strongly recommend not to use this mode in battery-operated systems because the 3/16 pulse length at lower bit rates consumes more power than the shorter pulses. At a baud rate of 9600 bit/s, the ratio of power consumption of both modes is a factor of 12 (!).

The TOIM3000 interfaces to an RS232 level converter through two pins, RD_232 and TD_232. These two pins provide the extra function that a single TOIM3000 IC can time share with both an infrared IrDA port and an RS232 port. Whenever RESET = 0, the TOIM3000 links to the infrared transceiver TFDS3000 through RD_IR and TD_IR pins. On the other hand, when RESET = 1, the TOIM3000 links to the RS232 port through RD_232 and TD_232 pins.

Features only for TOIM3232

The baud rate at which an RS232 serial port communicates with the external adapter is programmable inside the TOIM3232. This programmable baud rate should be used when the baud clock and the UART oscillator clock are not available. When BR/D = 0, the TOIM3232 inter–prets the signals at RD_232 and RD_IR pins as data to be transmitted and received data. On the other hand, whenever BR/D = 1, the TOIM3232 interprets the seven LSBs at the RD_232 input as the control word. The operating baud rate will change to its supposedly new baud rate when the BR/D returns back to LOW (“0”).

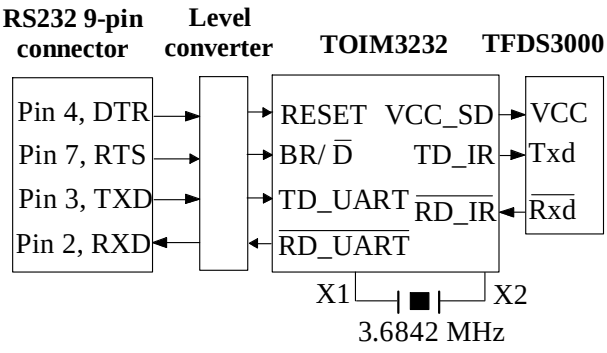


Figure 1. TOIM3232 – RS232 port interface (external infrared adapter)

Control Byte (8 bit)

<-----First char.---->				<---Second char.---->			
X	S2	S1	S0	B3	B2	B1	B0

where

X: Don't care
S1, S2: User-programmable bit
S0: IrDA pulse select
= (1) 1.627 µs output pulses
= (0) 3/16 bit time pulses *)
B0 .. B3: Baud rate selects, B0 = LSB

*) not recommended

Baud Rate Select Words

Note: IrDA standard only supports 2.4, 9.6, 19.2, 57.6, and 115.2 kbit/s.

B3	B2	B1	B0	Second Char.	Baud Rate
0	0	0	0	0	**115.2 k**
0	0	0	1	1	**57.6 k**
0	0	1	0	2	**38.4 k**
0	0	1	1	3	**19.2 k**
0	1	0	0	4	14.4 k
0	1	0	1	5	12.8k
0	1	1	0	6	**9.6k**
0	1	1	1	7	7.2 k
1	0	0	0	8	4.8 k
1	0	0	1	9	3.6 k
1	0	1	0	A	**2.4 k**
1	0	1	1	B	1.8 k
1	1	0	0	C	1.2 k

IrDA-Compliant 1.63 µs Pulse Hardwiring

The IrDA standard specifies the minimum pulse width as 1.41µs, regardless of the baud rate, and nominal as 3/16 bit time. Setting the output pulse as 1.63 µs for all baud rates saves LED current for those baud rates below 115.2 Kb/s. For the baud rate = 115.2 Kb/s, the 1.63 µs pulse mode is equivalent to 3/16 bit time mode, so there is no power saving at this baud rate for either scheme. However, for those baud rates below 115.2 Kb/s, the 3/16 bit time pulses becomes larger as the baud rate decreases. At 9600 bit/s communication, the 3/16 bit time pulse is 19.5 µs, which is 12 times more consumption current than a 1.63 µs pulse per output pulse. For notebook applications, since power saving is critical, we therefore recommend setting the pulse width at 1.63 µs.

The 1.63 µs pulses are generated by counting 3 clocks from the 1.8632 MHz clock. It is assumed that the 1.8632 MHz clock is readily available as the input clock driving the NSC 16550 and or output thereof. The designer must make sure the clock output from the NSC 16550:XOUT is 1.8632 MHz. If the 1.8632 MHz clock is not available, the designer must use an extra crystal to generate this clock. The 1.8632 MHz clock does not have to be synchronized to the 16 x baud rate clock.

The additional and important advantage of using the 1.63 µs mode is the digital filtering capability of the TOIM3000. The TOIM3000 uses the 1.8632 MHz clock to drive the digital filtering circuitry. Input infrared pulses of length less than one clock should then be filtered out. The following specifications must be set for this mode: S1=GND and S0=GND (see figure 3 for exact circuit connections).

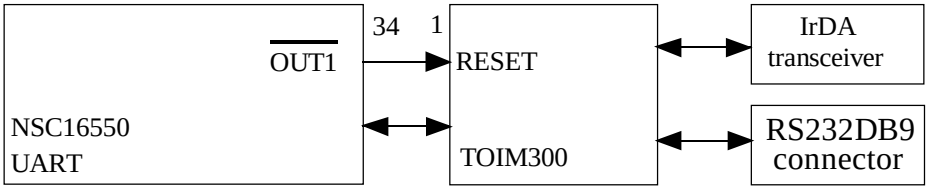


Figure 2. Using RESET to select communication port

IrDA-Compliant 3/16 Bit Time Hardwiring

If a 1.8632 MHz clock isn't available, the 3/16 bit time pulse can be used. The TOIM3000 can be hardwire-configured to output 3/16 bit time pulses.

Mode 3/16 bit time consumes more current than the 1.63 µs mode at baud rates slower than 115.2 Kb/s. At 115.2 Kb/s, current consumption is identical regardless whether the 3/16 bit time or 1.63 µs mode is used. For notebook applications, we strongly recommend using the 1.63 µs mode as explained in "IrDA-Compliant 1.63 µs Pulse Hardwiring". There are two reasons for this:

- Power saving
- Infrared signal input noise digital filtering.

Mode 3/16 bit time DOES NOT have digital filtering. Therefore, the TOIM3000 is vulnerable to noise at the infrared signal input into the TOIM3000.

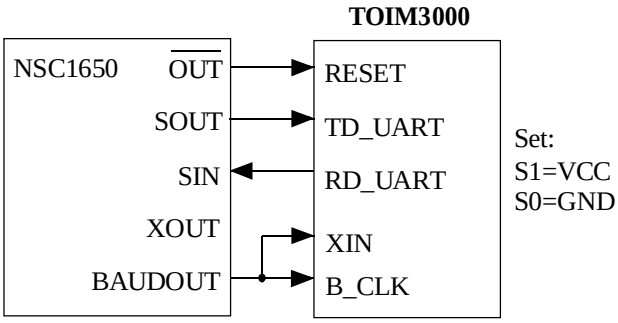


Figure 3. 3/ 16 bit time hardwiring

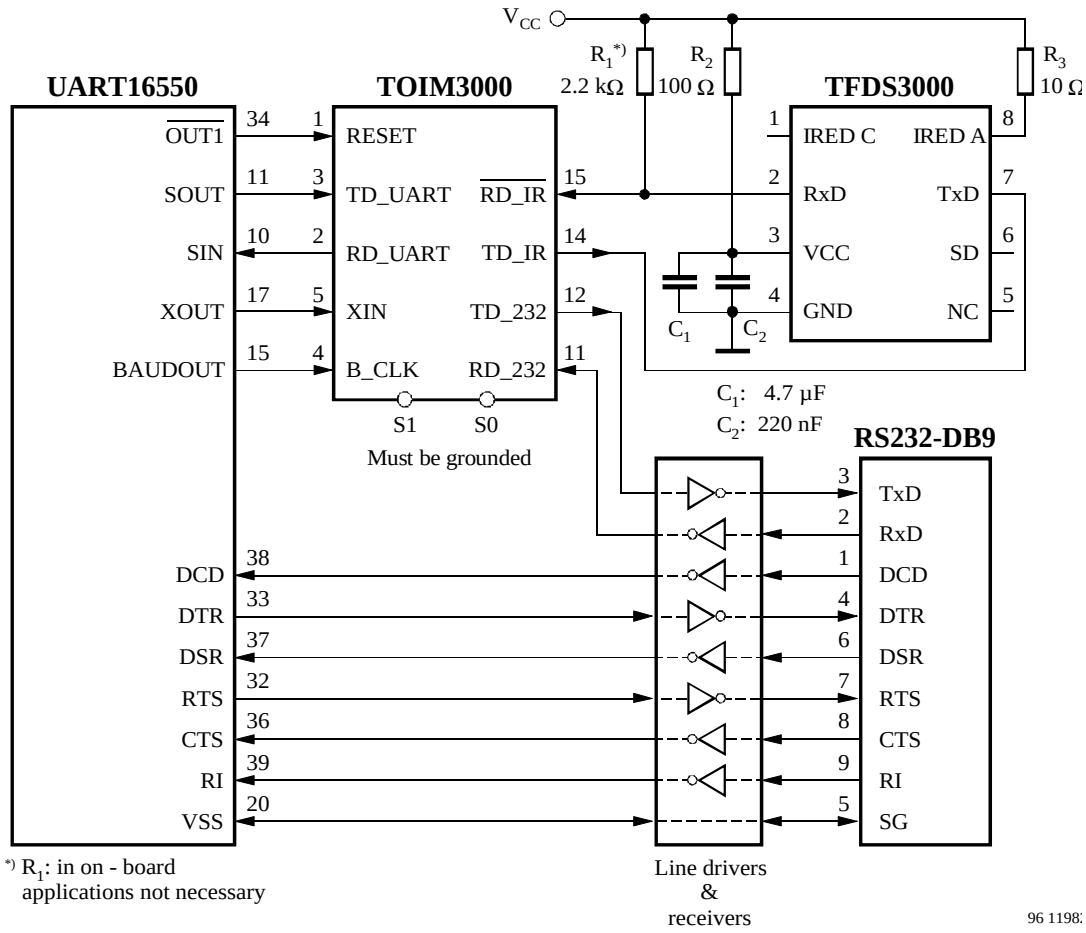


Figure 4. TOIM300/ TFDS300 with NSC UART 16550

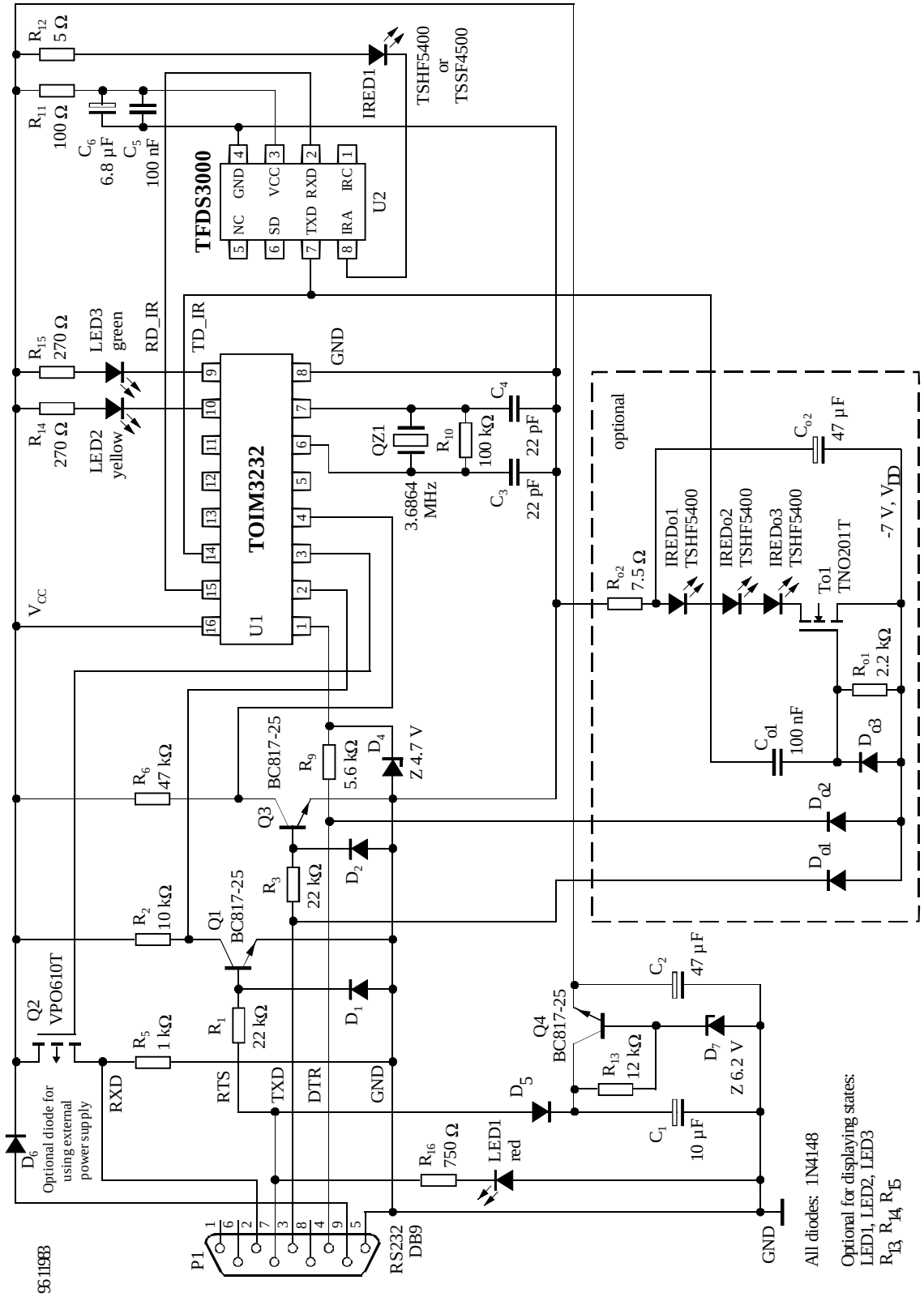


Figure 5. IrDA at RS232 port solution with discrete interface to RS232 port, PCB information, component placement, bill of material (see Appendix)

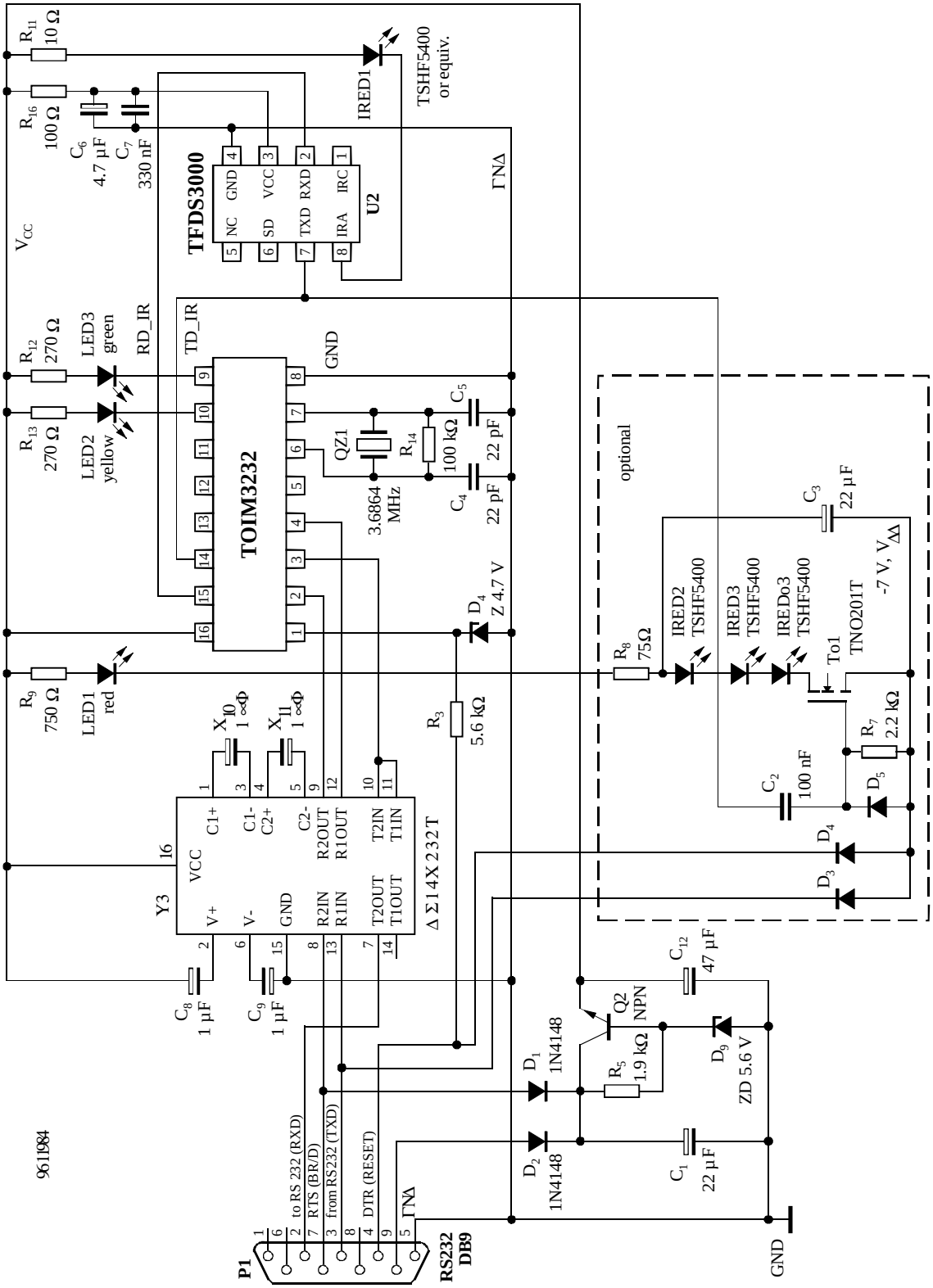


Figure 6. Solution with integrated interface to RS232 port

Appendix

Bill of Materials IrDA RS232 Adapter

Item	Quantity	Reference	Part
1	**1**	**U_1**	**TOIM3232**
2	**1**	**U_2**	**TFDS3000**
3	1	C_1	10 µF, 16 V
4	1	C_2	47 µF, 10 V
5	1	C_3	22 pF
6	1	C_4	22 pF
7	1	C_5	100 nF
8	1	C_6	6.8 µF, 6.3 V Tantalum
9	**1**	**D_1**	**1N4148**
10	**1**	**D_2**	**1N4148**
11	**1**	**D_3**	**Z4.7 V**
12	**1**	**D_5**	**1N4148**
13	**1**	**IRED1**	**TSHF5400**
14	**1**	**D_7**	**Z6.2 V**
15	1	P_1	Connector, DB9
16	1	Q_1	BC817-25
17		**Q_2**	**VP0610T**
18	1	Q_3	BC817-25
19	1	Q_4	BC817-25
20	1	Q_{Z1}	Quartz, 3.6864 MHz or ceram. resonator CSAC3.68MGC-TC, Murata
21	1	R_1	22 k
22	1	R_2	10 k
23	1	R_3	22 k
24	1	R_4	
25	1	R_5	1 k
26	1	R_6	47 k
27	1	R_7	
28	1	R_8	
29	1	R_9	5.6 k
30	1	R_{10}	100 k
31	1	R_{11}	100
32	2	R_{12}	(2×10 in parallel) =5
33	1	R_{13}	12 k
34	**1**	**PCB**	**TEMIC demo board**

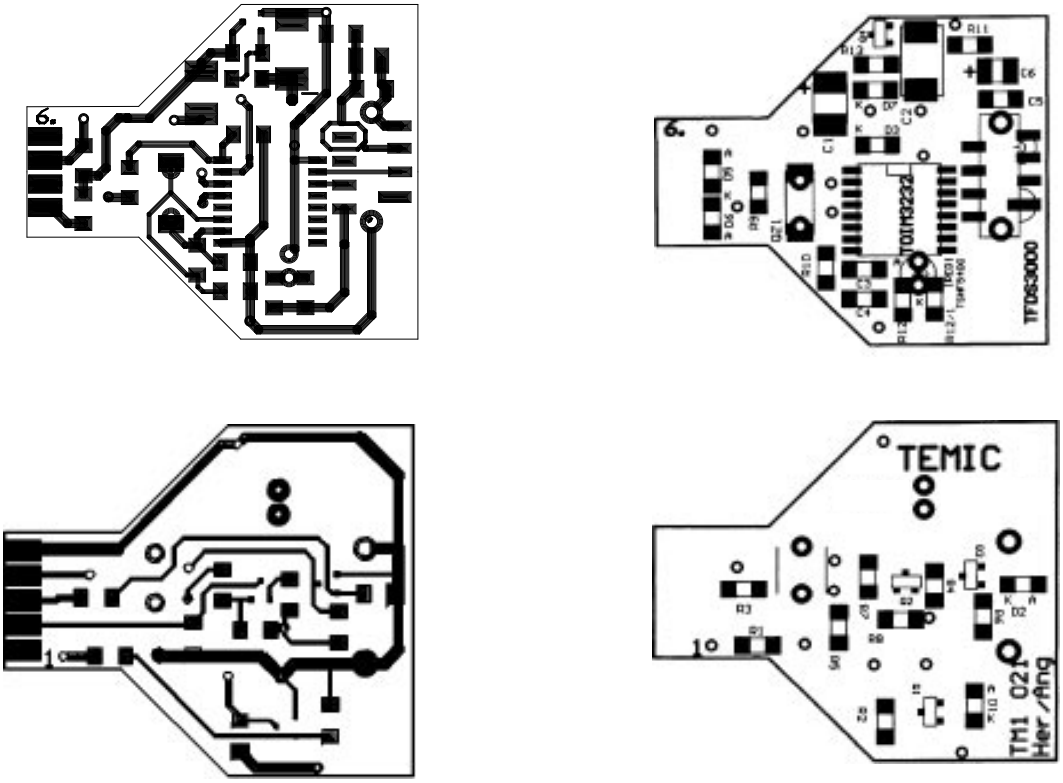
Optional Display (recommended only with external power supply)			
35	**1**	**LED1**	**LED, red, power on state display**
36	**1**	**LED2**	**LED, yellow, status**
37	**1**	**LED3**	**LED, green, status**
38	1	R_{16}	750, power-on state display
39	1	R_{14}	270, status
40	1	R_{15}	270, status

Optional Power Booster			
41	1	Ro_1	2.2 k
42	1	Ro_2	7.5
43	**1**	**To_1**	**TN0201T**
44	1	Co_1	100 nF
45	1	Co_2	47 µF 16 V
46	**1**	**Do_1**	**1N4148**
47	**1**	**Do_2**	**1N4148**
48	**1**	**Do_3**	**1N4148**
49	**1**	**D_6**	**1N4148 for external power supply**
50	**1**	**IREDo1**	**TSHF5400**
51	**1**	**IREDo2**	**TSHF5400**
52	**1**	**IREDo3**	**TSHF5400**

(bold = TEMIC parts)

Board Layout and Component Placement

Component placement, 9 Pin RS232 connector not shown.



Software for the TOIM3232

The control word is composed of two characters, written in hexadecimal, in format: YZ

UART Programming

For proper operation, the RS232 must be programmed to send a START bit plus an 8 bit data word, YZ and no STOP bit for every word sent. The transfer rate for programming must be identical with the formerly programmed data rate, or after resetting the TOIM3232, the default rate of 9600 bit/s is used.

Software Algorithm

STEP	RESET	BR/D	RD_UART	TD_UART	RD_IR	TD_IR	COMMENTS
1	HIGH	X	X	X	X	X	Resets all internal registers. Resets IrDA default baud rate of 9600 bit/s.
2	LOW	X	X	X	X	X	Wait at least 7 µs.
3	LOW	HIGH	X	X	X	X	Wait at least 7 µs. The TOIM3232 now enters the control word (programming) mode.
4	LOW	HIGH	YZ with Y = 1 for 1.627 µs Y = 0 for 3/16 bit length	X	X	X	Sending the control word YZ. Send ‘1Z’ if 1.627 µs pulses are used. Otherwise send ‘0Z’ if 3/16 bit pulses are used. ‘Y6’ keeps the 9.6 kbit/s data rate, whereas the ’0Z’ selects the 3/16 bit time pulses. Z = 0 sets to 115.2 kbit/s. Then wait at least 1 µs for hold–time.
5	LOW	LOW	DATA	DATA	DATA	DATA	Data communication between the TOIM3232 and the RS232 port has been established by BR/D LOW. The TOIM3232 now enters the data transmission mode. Both RESET and BR/D must be kept LOW (‘0’) during data mode. Software can re-program a new data rate by re-starting from step 3. The UART also must be set to the correct data rate ***).

***) For programming the UART, refer to e.g., National Semiconductor's data sheet of PC 16550 UART

Examples for Programming the RS232 Port and TOIM3232

Demo Program for Transmitting ASCII Characters

```
DECLARE SUB dtrreset ()
DECLARE SUB rtsbrdline ()
DECLARE SUB progtoim ()
DECLARE SUB proguart ()
DECLARE SUB progtoimsh ()
DECLARE SUB dataout ()
DECLARE SUB proof ()
CLS
DIM SHARED pro AS INTEGER
DIM SHARED sig AS INTEGER
DIM SHARED asci AS INTEGER
DIM SHARED rts$
DIM SHARED bdrate AS LONG

' Programmed for COM1, base address 3f8.
' In case of COM3 use 3e instead of 3f
' In case of COM2 use 2f instead of 3e
' In case of COM4 use 2e instead of 3e

'apply power supply voltage to dongle
   rts$ = "LO"              'Vcc high, pin2 LOW
   CALL rtsbrdline          'sets rts=br/d - switch to VCC

' Reset UART and PLD to default conditions
     CALL dtrreset          'In demo board the reset line is not(!) inverted
                            'Resets the TOIM3232 to 9600 bd
     bdrate = 9600
     CALL proguart          'sets the UART to 8,n,0,9600, default

'Read the speed from keyboard:
datin:
   INPUT "BAUDRATE"; bdrate
   CALL proof
     IF pro = 0 THEN GOTO datin

     INPUT "ASCII character to be transmitted (0-255)"; asci
     IF asci < 0 OR asci > 255 THEN GOTO datin

'Send the control word to the TOIM3232
sig = 0
PRINT "You can choose the different IrDA standards:"
INPUT "1.6æs pulses (1) or 3/16 bit (0) length"; sig
'Set rts active
   rts$ = "HI"              'pin2 = High
   CALL rtsbrdline          'sets rts=br/d - switch to Vcc
```

```
IF sig = 0 THEN CALL progtoim     'sets the toim to the given baudrate (3/16)
IF sig = 1 THEN CALL progtoimsh   'sets the toim to the given baudrate (1.6 µ)
FOR n = 1 TO 100           'some delay is necessary here
NEXT n
rts$ = "LO"               'at PLD = GND
CALL rtsbrdline           'sets rts=br/d - switch to GND
CALL proguart             'sets the baudrate to the right speed
FOR i = 1 TO 1000
NEXT i
FOR n = 1 TO 10000             'PROGRAM DURATION
FOR m = 1 TO 11500
NEXT m
CALL dataout
NEXT n
CLS
END

SUB dataout

OUT &H3F8, asci

END SUB

SUB dtrreset
OUT &H3FC, (INP(&H3FC) AND &HFE)
OUT &H3FC, (INP(&H3FC) OR &H1)
OUT &H3FC, (INP(&H3FC) AND &HFE)
END SUB

SUB progtoim
OUT &H3FB, (INP(&H3FB) AND &H7F)         'disable access to divisor latch bits
                          'enables data transfer to out/input register

IF bdrate = 115200 THEN OUT &H3F8, (&H0)  'LSB   0000
IF bdrate = 57600 THEN OUT &H3F8, (&H1)  'LSB   0001
IF bdrate = 38400 THEN OUT &H3F8, (&H2)  'LSB   0010
IF bdrate = 19200 THEN OUT &H3F8, (&H3)  'LSB   0011
IF bdrate = 14400 THEN OUT &H3F8, (&H4)  'LSB   0100
IF bdrate = 12800 THEN OUT &H3F8, (&H5)  'LSB   0101
IF bdrate = 9600 THEN OUT &H3F8, (&H6)   'LSB   0110
IF bdrate = 7200 THEN OUT &H3F8, (&H7)   'LSB   0111
IF bdrate = 4800 THEN OUT &H3F8, (&H8)   'LSB   1000
IF bdrate = 3600 THEN OUT &H3F8, (&H9)   'LSB   1001
IF bdrate = 2400 THEN OUT &H3F8, (&HA)   'LSB   1010
IF bdrate = 1800 THEN OUT &H3F8, (&HB)   'LSB   1011
IF bdrate = 1200 THEN OUT &H3F8, (&HC)   'LSB   1100
```

```
END SUB

SUB progtoimsh
OUT &H3FB, (INP(&H3FB) AND &H7F)          'disable access to divisor latch bits
                                          'enables data transfer to out/input register

IF bdrate = 115200 THEN OUT &H3F8, (&H10)  'LSB   0000
IF bdrate = 57600 THEN OUT &H3F8, (&H11)  'LSB   0001
IF bdrate = 38400 THEN OUT &H3F8, (&H12)  'LSB   0010
IF bdrate = 19200 THEN OUT &H3F8, (&H13)  'LSB   0011
IF bdrate = 14400 THEN OUT &H3F8, (&H14)  'LSB   0100
IF bdrate = 12800 THEN OUT &H3F8, (&H15)  'LSB   0101
IF bdrate = 9600 THEN OUT &H3F8, (&H16)   'LSB   0110
IF bdrate = 7200 THEN OUT &H3F8, (&H17)   'LSB   0111
IF bdrate = 4800 THEN OUT &H3F8, (&H18)   'LSB   1000
IF bdrate = 3600 THEN OUT &H3F8, (&H19)   'LSB   1001
IF bdrate = 2400 THEN OUT &H3F8, (&H1A)   'LSB   1010
IF bdrate = 1800 THEN OUT &H3F8, (&H1B)   'LSB   1011
IF bdrate = 1200 THEN OUT &H3F8, (&H1C)   'LSB   1100

END SUB

SUB proguart

OUT &H3FB, (&H3)                          '8bit, 1stop, no parity
OUT &H3FB, (INP(&H3FB) OR &H80)           'Enable access to divisor latch register

IF bdrate = 9600 THEN OUT &H3F8, (&HC)   'LSB   send divisor: 12 for 9600 baud
IF bdrate = 9600 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 19200 THEN OUT &H3F8, (&H6)  'LSB   send divisor: 6 for 19.20 kbaud
IF bdrate = 19200 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 38400 THEN OUT &H3F8, (&H3)  'LSB   send divisor: 3 for 38.4 kbaud
IF bdrate = 38400 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 57600 THEN OUT &H3F8, (&H2)  'LSB   send divisor: 2 for 57.6 kbaud
IF bdrate = 57600 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 115200 THEN OUT &H3F8, (&H1)  'LSB   send divisor: 1 for 115.2 kbaud
IF bdrate = 115200 THEN OUT &H3F9, (&H0)  'MSB = 0

IF bdrate = 7200 THEN OUT &H3F8, (&H10)  'LSB   send divisor: 16 for 7.20 kbaud
IF bdrate = 7200 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 4800 THEN OUT &H3F8, (&H18)  'LSB   send divisor: 24 for 4800 baud
IF bdrate = 4800 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 3600 THEN OUT &H3F8, (&H20)  'LSB   send divisor: 32 for 3.6 kbaud
IF bdrate = 3600 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 2400 THEN OUT &H3F8, (&H30)  'LSB   send divisor: 48 for 2.4 kbaud
IF bdrate = 2400 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 2000 THEN OUT &H3F8, (&H3A)  'LSB   send divisor: 58 for 2 kbaud
```

```
IF bdrate = 2000 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 1800 THEN OUT &H3F8, (&H40)  'LSB   send divisor: 64 for 1.8 kbaud
IF bdrate = 1800 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 1200 THEN OUT &H3F8, (&H60)  'LSB   send divisor: 96 for 1.2 kbaud
IF bdrate = 1200 THEN OUT &H3F9, (&H0)   'MSB = 0
'PRINT INP(&H3f8), INP(&H3f9)
'PRINT INP(&H3fB)
OUT &H3FB, (INP(&H3FB) AND &H7F)         'disable access to divisor latch bits
'PRINT INP(&H3fB)
END SUB

SUB proof
pro = 0
IF bdrate = 115200 THEN pro = 1
IF bdrate = 57600 THEN pro = 1
IF bdrate = 38400 THEN pro = 1
IF bdrate = 19200 THEN pro = 1
IF bdrate = 9600 THEN pro = 1
IF bdrate = 7200 THEN pro = 1
IF bdrate = 4800 THEN pro = 1
IF bdrate = 3600 THEN pro = 1
IF bdrate = 2400 THEN pro = 1
IF bdrate = 1800 THEN pro = 1
IF bdrate = 1200 THEN pro = 1
IF bdrate = 14400 THEN pro = 1
IF bdrate = 12800 THEN pro = 1
END SUB

SUB rtsbrdline
'Be aware: the signal is inverted by the MAX 232
IF rts$ = "LO" THEN OUT &H3FC, (INP(&H3FC) OR &H2)
'IF rts$ = "HI" then PRINT rts$
IF rts$ = "HI" THEN OUT &H3FC, (INP(&H3FC) AND &HFD)
'IF rts$ = "LO" then PRINT rts$
END SUB

'Demo program for receiving ASCII characters
DECLARE SUB dtrreset ()
DECLARE SUB rtsbrdline ()
DECLARE SUB progtoim ()
DECLARE SUB proguart ()
DECLARE SUB progtoimsh ()
DECLARE SUB dataout ()
DECLARE SUB datain ()
DECLARE SUB proof ()
DECLARE SUB wa ()
CLS
```

```
DIM SHARED pro AS INTEGER
DIM SHARED wai AS INTEGER
DIM SHARED sig AS INTEGER
DIM SHARED rts$
DIM SHARED bdrate AS LONG

' Programmed for COM3, base address 3e8.
' In case of COM1 use 3f instead of 3e
' In case of COM2 use 2f instead of 3e
' In case of COM4 use 2e instead of 3e
' GOTO start
' Reset UART and TOIM3232 to default conditions

RES:

'apply power supply voltage to dongle
   rts$ = "LO"              'Vcc high, pin2 LOW
   CALL rtsbrdline          'sets rts=br/d - switch to Vcc

'RESET:
CALL dtrreset            'In demo board the reset line is not(!) inverted
                         'Resets the PLD to 9600 bd
       bdrate = 9600
       CALL proguart           'sets the UART to 8,n,0,9600, default

start:
'Read the speed from keyboard:
datin:
   INPUT "BAUDRATE"; bdrate
   CALL proof
     IF pro = 0 THEN GOTO datin
'PRINT rts$
'SLEEP
'Send the control word to the TOIM3232
sig = 1
PRINT "You can choose the different IrDA standards:"
INPUT "1.6æs pulses (1) or 3/16 bit (0) length"; sig
'Set rts active
   rts$ = "HI"              'Vcc switched off, pin2 high
   CALL rtsbrdline          'sets rts=br/d - switch to Vcc

IF sig = 0 THEN CALL progtoim     'sets the toim to the given baudrate (3/16)
IF sig = 1 THEN CALL progtoimsh   'sets the toim to the given baudrate (1.6 µ)

wai = .01                '10 ms: some delay is necessary here, depending on the
CALL wa                  'actual data rate
```

```
rts$ = "LO"              'at PLD = GND
CALL rtsbrdline          'sets rts=br/d - switch to GND
'PRINT rts$
'SLEEP
CALL proguart            'sets the baudrate to the right speed
wai = .01
CALL wa
CALL datain

END

SUB datain
beg:
a = (INP(&H3FD) AND (&H1))
IF a = 1 THEN b = INP(&H3F8)
IF a = 1 THEN PRINT b; "  ";
GOTO beg
END SUB

SUB dataout
OUT &H3F8, (&H0)
END SUB

SUB dtrreset
'OUT &H3fC, (INP(&H3fC) AND &HFE)
FOR n = 1 TO 10
OUT &H3FC, (INP(&H3FC) OR &H1)
OUT &H3FC, (INP(&H3FC) AND &HFE)
NEXT n
OUT &H3FC, (INP(&H3FC) AND &HFE)
END SUB

SUB progtoim
OUT &H3FB, (INP(&H3FB) AND &H7F)        'disable access to divisor latch bits
                        'enables data transfer to out/input register

IF bdrate = 115200 THEN OUT &H3F8, (&H0)  'LSB   0000
IF bdrate = 57600 THEN OUT &H3F8, (&H1)  'LSB   0001
IF bdrate = 38400 THEN OUT &H3F8, (&H2)  'LSB   0010
IF bdrate = 19200 THEN OUT &H3F8, (&H3)  'LSB   0011
IF bdrate = 14400 THEN OUT &H3F8, (&H4)  'LSB   0100
IF bdrate = 12800 THEN OUT &H3F8, (&H5)  'LSB   0101
IF bdrate = 9600 THEN OUT &H3F8, (&H6)   'LSB   0110
IF bdrate = 7200 THEN OUT &H3F8, (&H7)   'LSB   0111
IF bdrate = 4800 THEN OUT &H3F8, (&H8)   'LSB   1000
IF bdrate = 3600 THEN OUT &H3F8, (&H9)   'LSB   1001
IF bdrate = 2400 THEN OUT &H3F8, (&HA)   'LSB   1010
```

```
IF bdrate = 1800 THEN OUT &H3F8, (&HB)   'LSB   1011
IF bdrate = 1200 THEN OUT &H3F8, (&HC)   'LSB   1100

END SUB

SUB progtoimsh
OUT &H3FB, (INP(&H3FB) AND &H7F)         'disable access to divisor latch bits
                                         'enables data transfer to out/input
                                         'register

IF bdrate = 115200 THEN OUT &H3F8, (&H10)  'LSB   0000
IF bdrate = 57600 THEN OUT &H3F8, (&H11)  'LSB   0001
IF bdrate = 38400 THEN OUT &H3F8, (&H12)  'LSB   0010
IF bdrate = 19200 THEN OUT &H3F8, (&H13)  'LSB   0011
IF bdrate = 14400 THEN OUT &H3F8, (&H14)  'LSB   0100
IF bdrate = 12800 THEN OUT &H3F8, (&H15)  'LSB   0101
IF bdrate = 9600 THEN OUT &H3F8, (&H16)   'LSB   0110
IF bdrate = 7200 THEN OUT &H3F8, (&H17)   'LSB   0111
IF bdrate = 4800 THEN OUT &H3F8, (&H18)   'LSB   1000
IF bdrate = 3600 THEN OUT &H3F8, (&H19)   'LSB   1001
IF bdrate = 2400 THEN OUT &H3F8, (&H1A)   'LSB   1010
IF bdrate = 1800 THEN OUT &H3F8, (&H1B)   'LSB   1011
IF bdrate = 1200 THEN OUT &H3F8, (&H1C)   'LSB   1100

END SUB

SUB proguart

OUT &H3FB, (&H3)                         '8bit, 1stop, no parity
OUT &H3FB, (INP(&H3FB) OR &H80)          'Enable access to divisor latch register

IF bdrate = 9600 THEN OUT &H3F8, (&HC)   'LSB   send divisor: 12 for 9600 baud
IF bdrate = 9600 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 14400 THEN OUT &H3F8, (&H8)  'LSB   send divisor: 8 for 14400 baud
IF bdrate = 14400 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 12800 THEN OUT &H3F8, (&H9)  'LSB   send divisor: 9 for 12800 baud
IF bdrate = 12800 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 19200 THEN OUT &H3F8, (&H6)  'LSB   send divisor: 6 for 19.20 kbaud
IF bdrate = 19200 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 38400 THEN OUT &H3F8, (&H3)  'LSB   send divisor: 3 for 38.4 kbaud
IF bdrate = 38400 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 57600 THEN OUT &H3F8, (&H2)  'LSB   send divisor: 2 for 57.6 kbaud
IF bdrate = 57600 THEN OUT &H3F9, (&H0)  'MSB = 0
IF bdrate = 115200 THEN OUT &H3F8, (&H1)  'LSB   send divisor: 1 for 115.2 kbaud
IF bdrate = 115200 THEN OUT &H3F9, (&H0)  'MSB = 0

IF bdrate = 7200 THEN OUT &H3F8, (&H10)  'LSB   send divisor: 16 for 7.20 kbaud
IF bdrate = 7200 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 4800 THEN OUT &H3F8, (&H18)  'LSB   send divisor: 24 for 4800 baud
IF bdrate = 4800 THEN OUT &H3F9, (&H0)   'MSB = 0
```

```
IF bdrate = 3600 THEN OUT &H3F8, (&H20)  'LSB   send divisor: 32 for 3.6 kbaud
IF bdrate = 3600 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 2400 THEN OUT &H3F8, (&H30)  'LSB   send divisor: 48 for 2.4 kbaud
IF bdrate = 2400 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 2000 THEN OUT &H3F8, (&H3A)  'LSB   send divisor: 58 for 2 kbaud
IF bdrate = 2000 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 1800 THEN OUT &H3F8, (&H40)  'LSB   send divisor: 64 for 1.8 kbaud
IF bdrate = 1800 THEN OUT &H3F9, (&H0)   'MSB = 0
IF bdrate = 1200 THEN OUT &H3F8, (&H60)  'LSB   send divisor: 96 for 1.2 kbaud
IF bdrate = 1200 THEN OUT &H3F9, (&H0)   'MSB = 0
'PRINT INP(&H3f8), INP(&H3f9)
'PRINT INP(&H3fB)
OUT &H3FB, (INP(&H3FB) AND &H7F)         'disable access to divisor latch bits
'PRINT INP(&H3fB)
END SUB

SUB proof
pro = 0
IF bdrate = 115200 THEN pro = 1
IF bdrate = 57600 THEN pro = 1
IF bdrate = 38400 THEN pro = 1
IF bdrate = 19200 THEN pro = 1
IF bdrate = 14400 THEN pro = 1
IF bdrate = 12800 THEN pro = 1
IF bdrate = 9600 THEN pro = 1
IF bdrate = 7200 THEN pro = 1
IF bdrate = 4800 THEN pro = 1
IF bdrate = 3600 THEN pro = 1
IF bdrate = 2400 THEN pro = 1
IF bdrate = 1800 THEN pro = 1
IF bdrate = 1200 THEN pro = 1
END SUB

SUB rtsbrdline
'Be aware: the signal is inverted by the MAX 232
IF rts$ = "LO" THEN OUT &H3FC, (INP(&H3FC) OR &H2)
'IF rts$ = "HI" then PRINT rts$
IF rts$ = "HI" THEN OUT &H3FC, (INP(&H3FC) AND &HFD)
'IF rts$ = "LO" then PRINT rts$
END SUB

SUB wa
t1 = TIMER
DO UNTIL (t2 - t1) > wai
t2 = TIMER
LOOP
END SUB
```