HDQ – 1-Wire

The HDQ and the 1-Wire interface both are interfaces using only 1 single pin. The HDQ interface has been developed by Texas Instruments and is used in many of their applications. Dallas developed the 1-Wire bus, which is slightly more complex because of its network addresses, but in return enables using several participants on the bus that can be individually addressed.

With the Tiger up to 8 of such interfaces can be driven simultaneously and parallel. So you can connect 8 different devices to 8 Tiger pins and change the functionality from HDQ to 1-Wire and vice versa even during run time.

Important: At the HDQ / 1-Wire pin a ***pull-up resistor*** to Vcc must be connected, otherwise the communication will not work. For example, for HDQ a 10k resistor and for 1-Wire a 4.7k resistor can be used. Both the GND lines certainly must be connected, too.

At the beginning a setup has to be executed for each channel, only then this pin can be used as interface. The following functions are available for communication:

HDQ_1WIRE_SETUP	=> defines port and pin of channel
HDQ_READ	=> reads byte through HDQ
HDQ_WRITE	=> writes byte through HDQ
ONEWIRE_RESET	=> issues reset signal through 1-Wire
ONEWIRE_READ	=> reads byte through 1-Wire
ONEWIRE_WRITE	=> writes byte through 1-Wire
ONEWIRE_READ_BIT	=> reads bit through 1-Wire (net addresses)
ONEWIRE_WRITE_BIT	=> writes bit through 1-Wire (net addresses)

Important: At 1-Wire, the basic communication is implemented. Realization of the communication protocol has to be written in BASIC. For the appropriate protocol please refer to the device's datasheet. At first the net address command has to be sent, which is followed by the function command with parameters etc. In BASIC you can read and write byte by byte through the bus. Should you intend to make use of the search algorithms for net addresses, additional functions for reading and writing bit by bit are available for you.

HDQ_ONEWIRE_SETUP

Flag = HDQ_ONEWIRE_SETUP (Port, Pin, Channel)

Function: One port and one pin for the HDQ and 1-Wire line is determined. Additionally a channel can be given, which enables to communicate on several channels and so the connect various devices.

Parameters:

	B	W	L	S	F	
Port	●	●	●	-	-	This port is used for the interface
Pin	●	●	●	-	-	This pin of above port is used for the interface
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
Flag	●	●	●	-	-	0 = Parameter OK -1 = Channel invalid -2 = Port invalid -3 = Pin invalid

Note: HDQ_1WIRE_SETUP must be executed at the beginning, before the interface can be used. There is no default pin for this interface.

Note: If more than one interface is to be served, a channel should be set. If only one interface is used, setting the channel can be left out – it will be set to 0 by default.

Important: At the HDQ / 1-Wire pin a ***pull-up resistor*** to Vcc must be connected, otherwise the communication will not work. For example, for HDQ a 10k resistor and for 1-Wire a 4.7k resistor can be used. Both the GND lines certainly must be connected, too.

Example for setting up pin L80 (as channel 0):

```
'FLAG =  HDQ_1WIRE_SETUP(Port, Pin, Channel)
  res =  HDQ_1WIRE_SETUP(  8,   0)       ' End task MAIN
```

HDQ_READ

RES = HDQ_READ (Address, Channel)

Function: Reads one byte through the HDQ interface from *Address*. Devices that are accessed through HDQ have addresses that can be read out or written to, mostly registers or a RAM/ROM area.

Parameters:

	B	W	L	S	F	
Address	●	●	●	-	-	Address of connected device that is read out
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
RES	●	●	●	-	-	Read value stored in low byte of numerical variable: 0...255: Content of 'Address' -1 = Address too high -2 = Channel invalid Error codes only available with LONG variables!

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

Note: There is an R/W (Read or Write) bit in the HDQ protocol determining if data is to be written or read and that bit is added to the 7-bit address. This is covered by the function HDQ_READ, so please do *not* specify this bit yourself!

Program example:

```
task main
  byte addr
  long res
  string hdq1$(02Ch)
  hdq1$ = ""

  ' 1. without Channel ( Channel = 0 ) !!!

'FLAG =  HDQ_1WIRE_SETUP(Port, Pin, Channel)
  res =  HDQ_1WIRE_SETUP(  8,   0)

  if res >= 0 then
    for addr=0 to 02Ch
'     res = HDQ_READ(Address, Channel)
      res = HDQ_READ(addr)
      if res >= 0 then
        hdq1$ = hdq1$ + chr$(res)
      endif
    next
  endif



  ' 2. with Channel (Channel = 7)
'FLAG =  HDQ_1WIRE_SETUP(Port, Pin, Channel)
  res =  HDQ_1WIRE_SETUP( 8,    0,     7)

  if res >= 0 then
    for addr=0 to 02Ch
'     res = HDQ_READ(Address, Channel)
      res = HDQ_READ(   addr,    7)
      if res >= 0 then
        hdq1$ = hdq1$ + chr$(res)
      endif
    next
  endif


end
```

HDQ_WRITE

FLAG = HDQ_WRITE (Address, Data, Channel)

Function: A byte is written through the HDQ interface to *Address*. Devices that are accessed through HDQ have addresses that can be read out or written to, mostly registers or a RAM/ROM area.

Parameters:

	B	W	L	S	F	
Address	●	●	●	-	-	Address in connected device to which is written to (max. 7-bit address)
Data	●	●	●	-	-	Data byte that is written (max. 255)
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
FLAG	●	●	●	-	-	0 = OK -1= Address too high -2= Data byte too high -3= Channel invalid

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

Note: There is an R/W (Read or Write) bit in the HDQ protocol determining if data is to be written or read and that bit is added to the 7-bit address. This is covered by the function HDQ_WRITE, so please do *not* specify this bit yourself!

ONEWIRE_RESET

FLAG = ONEWIRE_RESET (Channel)

Function: Sends a 1-Wire signal. The bus is pulled to LOW for 480µs. A reset has to be sent before each (command) sequence. For the precise method of the 1-Wire protocol and the valid commands for the device, please refer to the appropriate datasheet.

Parameters:

	B	W	L	S	F	
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
FLAG	●	●	●	-	-	0= No device connected to bus ›0= At least 1 device connected to bus -1= Channel invalid

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

ONEWIRE_READ

RES = ONEWIRE_READ (Channel)

Function: Reads 1 byte through the 1-Wire interface.

Parameters:

	B	W	L	S	F	
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
RES	●	●	●	-	-	Read value stored in low byte of numerical variable: 0...255: Content of 'Address' -1 = Channel invalid **ATTENTION:** 0FFh (-1) could be real data when using BYTE or WORD variable.

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

Note: Before a byte can be read, the slave has to know that he should transmit a byte, so the correct commands have to be written to the device first. These commands are e.g. 0CCh for skipping the net address (if only 1 slave is connected), then the command for reading (069h), followed by the address – now this address can be read out.

Program example:

```
#define NETADDRESS_SKIP 0CCh
#define IWIRE_FUNC_READ 69h
#define STARTADDR 0

task main
  byte x
  long res
  string onewire$(02Ch)
  onewire$ = ""

  ' 1. without Channel ( Channel = 0 ) !!!

'FLAG =  HDQ_1WIRE_SETUP(Port, Pin, Channel)
  res =  HDQ_1WIRE_SETUP(  8,   0)

  if res >= 0 then
'-------------------------------------------------------------------------
' 1. Send reset pulse
'-------------------------------------------------------------------------
'  FLAG = ONEWIRE_RESET(Channel)
    res = ONEWIRE_RESET()
    if res > 0 then
'-------------------------------------------------------------------------
' 2. Send net address command
'-------------------------------------------------------------------------
'    FLAG = ONEWIRE_WRITE(Value,          Channel)
     res = ONEWIRE_WRITE(NETADDRESS_SKIP)
     if res >= 0 then
'-------------------------------------------------------------------------
' 3. Function Command + Address
'-------------------------------------------------------------------------
'      FLAG = ONEWIRE_WRITE(Value,         Channel)
       res = ONEWIRE_WRITE(IWIRE_FUNC_READ)
       if res >= 0 then
'        FLAG = ONEWIRE_WRITE(Value,    Channel)
         res = ONEWIRE_WRITE(STARTADDR)
         for x = 0 to 1Ah
'-------------------------------------------------------------------------
' 4. Read out data
'-------------------------------------------------------------------------
'           RES = ONEWIRE_READ(Channel)
           res = ONEWIRE_READ()
           onewire$ = onewire$ + chr$(res)
         next
       endif
     endif
   endif
  endif


end
```

ONEWIRE_WRITE

FLAG = ONEWIRE_WRITE (Value, Channel)

Function: Writes 1 byte through the 1-Wire interface.

Parameters:

	B	W	L	S	F	
Value	●	●	●	-	-	Value to be sent through 1-Wire interface Value range: 0...255
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
FLAG	●	●	●	-	-	0: OK -1: Channel invalid -2: Value too high

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

Note: Writing does not refer only to writing to the chip, but is generally the sending of data. When an address on the chip is to be (re)written, special commands must be sent first – just like when reading. These commands are issued using the ONEWIRE_WRITE function.

Program example:

```
#define NETADDRESS_SKIP 0CCh
#define IWIRE_FUNC_READ 69h
#define STARTADDR 0

task main
  byte x
  long res
  string onewire$(02Ch)
  onewire$ = ""

  ' 1. without Channel ( Channel = 0 ) !!!

'FLAG =  HDQ_1WIRE_SETUP(Port, Pin, Channel)
  res =  HDQ_1WIRE_SETUP(  8,   0)

  if res >= 0 then
'-------------------------------------------------------------------------
' 1. Send reset pulse
'-------------------------------------------------------------------------
'  FLAG = ONEWIRE_RESET(Channel)
    res = ONEWIRE_RESET()
    if res > 0 then
'-------------------------------------------------------------------------
' 2. Send net address command
'-------------------------------------------------------------------------
'    FLAG = ONEWIRE_WRITE(Value,          Channel)
     res = ONEWIRE_WRITE(NETADDRESS_SKIP)
     if res >= 0 then
'-------------------------------------------------------------------------
' 3. Function Command + Address
'-------------------------------------------------------------------------
'      FLAG = ONEWIRE_WRITE(Value,          Channel)
       res = ONEWIRE_WRITE(IWIRE_FUNC_READ)
       if res >= 0 then
'        FLAG = ONEWIRE_WRITE(Value,    Channel)
         res = ONEWIRE_WRITE(STARTADDR)
         for x = 0 to 1Ah
'-------------------------------------------------------------------------
' 4. Daten auslesen
'-------------------------------------------------------------------------
'           RES = ONEWIRE_READ(Channel)
           res = ONEWIRE_READ()
           onewire$ = onewire$ + chr$(res)
         next
       endif
     endif
   endif
  endif


end
```

ONEWIRE_READ_BIT

RES = ONEWIRE_READ_BIT (Channel)

Function: Reads one single bit through the 1-Wire interface. Should the search algorithms for net addresses be used, reading and writing bit by bit are needed.

Parameters:

	B	W	L	S	F	
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
RES	●	●	●	-	-	Result (0 or 1) or -1: Channel invalid

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.

ONEWIRE_WRITE_BIT

FLAG = ONEWIRE_WRITE_BIT (Value, Channel)

Function: Writes one single bit through the 1-Wire interface. Should the search algorithms for net addresses be used, reading and writing bit by bit are needed.

Parameters:

	B	W	L	S	F	
Value	●	●	●	-	-	Value to be sent through 1-Wire interface Value range: 0 or 1
Channel	●	●	●	-	-	Channel 0...7 This parameter is *optional*; if it is left out it is set to 0 by default!
						Function value:
FLAG	●	●	●	-	-	0: OK -1: Channel invalid -2: Value invalid (too high)

Note: Before this function can be used, once the HDQ_1WIRE_SETUP has to be executed for the appropriate channel.