One Wire Library Functions for the XCSB compiler.

The one wire library functions consists of two separate files.

One_wire.bas - Basic one wire functions

One_wire_utils.bas – Extended one wire functions.

The one_wire.bas is available as a free download from www.xcprod.com/cdb .

Note this library file is **not** supported by xcprod, problems and queries should be addressed to cdb@barnard.name.

The library files have been tested using v1.07 and 1.08.1pro of XCSB. Available from www.xcprod.com/titan/xcsb .

The One_wire_utils.bas will be available for a nominal charge. Further details on the above webpage.

Functions covered by the two files.

Function	***Library File***	***Purpose***
m_1W_reset	One_wire	Performs reset and presence detect of device
m_1W_r_byte	One_wire	Reads a byte from a one wire device
m_1W_wr_byte	One_wire	Writes a byte to a one wire device
skip_rom	One_wire	Performs a reset and sends the skip rom command
readrom_1W	One_wire_utils	Performs a reset, reads the rom and crc checks
device_search	One_wire_utils	Multiple device search where the SN's are unknown
find_family **	One_wire_utils	Future function **not available as yet**
match_1W_rom	One_wire_utils	Searches for a device specified by a ram variable
match_1W_rom_code	One_wire_utils	Searches for a device specified in program code
crc8_1W	One_wire_utils	CRC checksum 8 bit version – used by serial #'s
crc16_1W **	One_wire_utils	Future function **not available as yet**
strong_1W_pup	One_wire_utils	Strong pullup, required by heavy current devices

How to implement the functions.

It is suggested that the file(s) are saved in the XCSB library directory.

An example of how to include the standard file is shown below.

include "LIB/one_wire.bas"

To include the extra functions file you only need to declare

include "LIB/one_wire_utils.bas" , this file automatically includes one_wire.bas and an error will occur if both are declared.

Usage Method:

Interrupts must be disabled whenever any of the one wire functions are used. They can be re-enabled when control returns to the main code.

The one wire port and data bit must be declared in the main program file as follows.

hwreg M_1W_PORT = &PORTx , where 'x' is the port letter assigned to the one wire buss.
const M_1W_BIT = x, where 'x' is the bit number connected to the one wire buss.

Example code

```
hwreg M_1W_PORT = &PORTB      //Port 1Wire bus is connected to
const M_1W_BIT = 3            //Port bit 1Wire device is connected to
```

Available Functions.

m_1W_reset

Library - one_wire.bas

Description:

This function sends the reset signal to the one wire device and receives the presence detect signal from the slave. The function must be called prior to accessing a device and again at the end of communication. Some of the one wire functions automatically call m_1W_reset to commence communication, but a call to m_1W_reset must **always** be written into the main code to terminate a communication session. All functions that automatically call m_1W_reset will return to the main code immediately if an error condition exists.

Definition:

```
ubyte m_1W_reset()
```

Usage.

On entry there are no parameters

On exit

Returns the following codes for further manipulation if required

D_SHORT = -1, data line is shorted permently low.
DEV_OK = 0, one wire device replied with a detect pulse.
NO_DEV = 1, the line has stayed high, no one wire device connected.

Example code

```
status = m_1W_reset()

if status == NO_DEV then

    set_bit(RB7)  //The line is high

endif
```

skip_rom

Library - one_wire.bas

Description:

This function sends the reset pulse to the slave device and then send the skip rom code (0xCC) to the device

Definition:

ubyte skip_rom()

Usage:

On entry there are no parameters.

On exit returns the m_1W_reset() codes. See above!

Example code

```
if (status = skip_rom()) == NO_DEV then

    do something exciting here

 endif
```

m_1W_r_byte

Library one_wire.bas

Description:

This function reads in one 8 bit byte from a one wire device. If called explicitly by the main code, m_1W_reset() needs to be called before and after the call to this function if no further access is required.

Definition:

ubyte m_1W_r_byte()

Usage:

On entry there are no parameters.

On exit returns the byte just read.

Example code

```
rec_byte = m_1W_r_byte()

USART_send_poll(rec_byte)
```

m_1W_wr_byte

Library - one_wire.bas

Description:

Writes one 8 bit byte to a one wire device. If called explicitly from the main code then a m_1W_reset() call is required as mentioned above.

Definition:

m_1W_wr_byte(ubyte txbyte)

Usage:

On entry **txbyte** contains the byte to be sent.

On exit there is no return

Example code

```
m_1W_wr_byte(RDTIME) //send readtime code 0x66
```

readrom_1W

Library - one_wire_utils.bas

Description:

This function sends a reset signal to the device and then reads in the 8 byte serial number and performs a CRC8 check. This function must is used if there is only one 1-Wire slave directly connected at any one time. If multiple 1 wire devices are connected then use either the match_rom or search_rom functions.

Definition:

ubyte readrom_1W(ubyte *serial_no)

Usage:

On entry *serial_no contains a pointer to an 8 byte array where the serial number is to be stored.

On exit *serial_no will have the 64bit rom serial number, and the return byte, the crc or reset exit code. If the CRC is correct then the return will read false (0), any other number will either be one of the reset failure codes 0x01 or 0xFF, any other result means that the crc check has failed. Probably due to transmission problems.

Example code

ubyte rom_num[8] or the FIFO could be used

```
error_byte = readrom_1W(&rom_num)
if error_byte == ERROR CODE  then
   panic here
endif
```

device_search

Library – one_wire_utils.bas

Description:

This function searches the one wire buss for multiple devices and returns the serial numbers of all devices found. It is mainly used when there is more than one device permanently connected / hard wired to the one wire buss, and the serial numbers of the devices are not known in advance or can be used in conjunction with the find_family function. See below.

Definition:

device_search(ubyte *devices)

Usage:

On entry *devices contains a pointer to a default array of 64 bytes. The array must be declared in the main program. The default search is for 8 devices. It is suggested that for the lower end 16 series Pics a maximum of 4 one wire devices are physically connected to the one wire buss which requires 32 bytes. This is achieved by altering the constant MAX_DEV in one_wire.bas.

device_search calls search_rom(ubyte *devices)

The array would be declared in the following manner.

section DEVICES_SECT:1000

ubyte DEVICES_SECT devices_1W[64] // For 8 devices. Change to 32 for 4 devices.

The most compact code will be produced if the number of devices searched for, are powers of 2. IE. 2,4,8,16 etc.

On exit the array will contain 8 complete serial numbers.

Example code

section DEVICES_SECT:1000

ubyte DEVICES_SECT devices_1W[64]

device_search(&devices_1W)

find_family

Library – one_wire_utils.bas

Description:

Function to search for specific families of one wire devices. The function is used where only certain types of one wire devices are to be recognised.

Definition:

This function has yet to be implemented

Usage:

Example code

match_1W_rom

Library - one_wire_utils.bas

Description:

Match rom takes a one wire serial number that is held in ram and searches for the device on the buss.

If found, it checks that the serial number matches, and then returns to the main program so that further communication can be carried out with the called device. NOTE: If the serial number received differs from that sent, a reset signal is sent to the device and communication is closed. Therefore the reset status must be tested for this condition.

Definition:

ubyte match_1W_rom(ubyte *device_sn)

Usage:

On entry *device_sn is a pointer to an 8 byte ram array which contains the serial number of the device to be accessed.

On exit the reset status is returned for further processing.

Example code

if (status = match_1W_rom(&sn_variable)) == NO_DEV then

error code here

endif

match_1W_rom_code

Library - one_wire_utils.bas

Description:

This function works exactly as above, except the serial number(s) have been written into rom at compile time or by using the FLASH_write rom functions found in the XCSB flash_lib.bas files.

Definition:

ubyte match_1W_rom_code(const *device_sn)

Usage:

On entry *device_sn points to a table address in rom.

On exit the return code from the reset function is available for further processing.

Example code

status = match_1W_rom_code(&serial_num)

if status == NO_DEV then

error code here

endif

crc8

Library - one_wire_utils.bas

Description:

This function is primarily used by the read_rom routine and checks the integrity of the serial number of the device. This function could be called independently in the main program, it calculates the crc on the fly and therefore does not take up any rom space with table lookups.

Definition:

ubyte crc8_1W(ubyte *serial_no)

Usage:

On entry *serial_number is a pointer to an array holding the serial number to be verified.

On exit returns the crc calculated in hex. This should be zero if the code is verified. Any other number means verification failure.

Example code

crc_checked = crc8_1W(&serial_no)

if crc_checked != 0 then

some error code here //crc was not zero

endif

crc16

Library - one_wire_utils.bas

Description:

The crc16 is used by devices that contain EEPROMS and the SHA security iButton. Like crc8 the crc returns zero if the correct byte information has been received. The crc16 is calculated on the fly, so no rom space is taken up with large table lookups.

Definition:

This function has yet to be implemented

Usage:

Example code

Deliberately Blank

strong_1W_pup

Library - one_wire_utils.bas

Description:

Function to source high current to certain one wire devices that require higher than standard current either during a temperature conversion or are located further from the microcontroller than the one wire standard allows. Any one wire devices that require more than 20mA will need to be powered separately or one of the Maxim 1-wire bus drivers should be used.

Definition:

strong_1W_pup()

Usage:

On entry there are no parameters.

On exit there are no parameters. Note the one wire bus is returned to normal high Z after 255mS

Example code

```
if (status = skip_rom()) == NO_DEV then

    panic here

 endif

m_1W_wr_byte(DO_CONVERSION)
```

strong_1W_pup()

```
rec_byte = m_1W_r_byte()

status = m_1W_reset()

do something with rec_byte
```