modbus\_send\_raw\_request(3)

==========================

NAME

----

modbus\_send\_raw\_request - send a raw request

SYNOPSIS

--------

\*int modbus\_send\_raw\_request(modbus\_t \*'ctx', uint8\_t \*'raw\_req', int 'raw\_req\_length');\*

DESCRIPTION

-----------

The \*modbus\_send\_raw\_request()\* function shall send a request via the socket of

the context \_ctx\_. This function must be used for debugging purposes because you

have to take care to make a valid request by hand. The function only adds to the

message, the header or CRC of the selected backend, so \_raw\_req\_ must start and

contain at least a slave/unit identifier and a function code. This function can

be used to send request not handled by the library.

The public header of libmodbus provides a list of supported Modbus functions

codes, prefixed by `MODBUS\_FC\_` (eg. `MODBUS\_FC\_READ\_HOLDING\_REGISTERS`), to help

build of raw requests.

RETURN VALUE

------------

The function shall return the full message length, counting the extra data

relating to the backend, if successful. Otherwise it shall return -1 and set

errno.

EXAMPLE

-------

[source,c]

-------------------

modbus\_t \*ctx;

/\* Read 5 holding registers from address 1 \*/

uint8\_t raw\_req[] = { 0xFF, MODBUS\_FC\_READ\_HOLDING\_REGISTERS, 0x00, 0x01, 0x0, 0x05 };

int req\_length;

uint8\_t rsp[MODBUS\_TCP\_MAX\_ADU\_LENGTH];

ctx = modbus\_new\_tcp("127.0.0.1", 1502);

if (modbus\_connect(ctx) == -1) {

fprintf(stderr, "Connection failed: %s\n", modbus\_strerror(errno));

modbus\_free(ctx);

return -1;

}

req\_length = modbus\_send\_raw\_request(ctx, raw\_req, 6 \* sizeof(uint8\_t));

modbus\_receive\_confirmation(ctx, rsp);

modbus\_close(ctx);

modbus\_free(ctx);

-------------------

SEE ALSO

--------

linkmb:modbus\_receive\_confirmation[3]

AUTHORS

-------

The libmodbus documentation was written by Stéphane Raimbault

<stephane.raimbault@gmail.com>