```
<?
/**
 * @file modbus.inc
 * @author Akash Heimlich
 * @version 1.0
 * @date 09/04/2021
 * @description Modbus functions
 */

define('MB_ILLEGAL_FUNCTION',-1);
define('MB_ILLEGAL_ADDRESS',-2);
define('MB_ILLEGAL_VALUE',-3);
define('MB_SLAVE_FAILURE',-4);

/**
  * mb_get_error_string - get modbus exception error
  *
  * Return a human readable error string based on the exception code
  *
  * @param int $error_number - Modbus except (negative value)
  *         For exceptions not generated by the functions in this library,
send it as -$error_number
  * @return string Human readable description of the error
  */
function mb_get_error_string($error_number) {

    if ($error_number==MB_ILLEGAL_FUNCTION) {
        return "Illegal Modbus Function";
    }
    if ($error_number==MB_ILLEGAL_ADDRESS) {
        return "Illegal Modbus Register Address";
    }
    if ($error_number==MB_ILLEGAL_VALUE) {
        return "Illegal Modbus Value";
    }
    if ($error_number==MB_SLAVE_FAILURE) {
        return "Device Slave Failure";
    }
    return "";
}

/**
  * mb_set_int16_0x03 - set 16-bit register via function 03
  *
  * Set a modbus register using function code 03
  *
  * @param int $id - Modbus device ID
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $cmd - Value
```

```
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */

function mb_set_int16_0x03($id, $bus, $reg, $cmd, $retries)
{
    print("/scripts/tests/modbus.inc");
    global $DEBUG;
    $base_addr = $reg;
    $num_bytes = 2;
    $msg = array(
        $id,
        6
    );
    $msg[] = $base_addr >> 8;
    $msg[] = $base_addr & 0xFF;
    $msg[] = $cmd >> 8;
    $msg[] = $cmd & 0xFF;
    if ($DEBUG) {
        print ("<span class='info'>[MODBUS]</span>\r\nSet register -
Retries=" . $retries);
    }
    $res = 0;
    while ($retries) {
        $res = mb_send_command($bus, $msg);
        $retries--;
        if (is_array($res) && sizeof($res)) {
            break;
        }
        if ($DEBUG) {
            print("\r\n<span style='color:red'>Modbus Command
FAILED</span>\r\n");
        }
    }

    if (is_array($res)) {
        // if the result was successful
        if ($res[1]==6) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}

/**
  * mb_set_int16_0x10 - set 16-bit register via function 10h
  *
  * Set a modbus register using function code 10h
  *
  * @param int $id - Modbus device ID
```

```
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $cmd - Value
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */

function mb_set_int16_0x10($id, $bus, $reg, $cmd, $retries)
{
    global $DEBUG;
    $base_addr = $reg;
    $num_bytes = 2;
    $msg = array(
        $id,
        0x10
    );
    $msg[] = $base_addr >> 8;
    $msg[] = $base_addr & 0xFF;
    $msg[] = 0;
    $msg[] = 1;
    $msg[] = 2;
    $msg[] = $cmd >> 8;
    $msg[] = $cmd & 0xFF;
    $res=0;
    while ($retries) {
        $res = mb_send_command($bus, $msg);
        $retries--;
        if (is_array($res) && sizeof($res)) {
            break;
        }
        if ($DEBUG) {
            print("\r\n<span style='color:red'>Modbus Command
FAILED</span>\r\n");
        }

    }
    if (is_array($res)) {
        // if the result was successful
        if ($res[1]==0x10) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}

/**
  * mb_set_uint32_be_0x10 - set 32-bit uint big endian via function 10h
  *
  * Set a big endian UINT32 register via function 0x10h
```

```
  *
  * @param int $id - Modbus device ID
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $cmd - Value
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */
function mb_set_uint32_be_0x10($id, $bus, $reg, $cmd, $retries)
{
    global $DEBUG;
    $base_addr = $reg;
    $num_bytes = 4;
    $msg = array(
        $id,
        0x10
    );
    $msg[] = $base_addr >> 8;
    $msg[] = $base_addr & 0xFF;
    $msg[] = 0;
    $msg[] = 2;
    $msg[] = 4;
    $msg[] = $cmd >> 24 & 0xFF;
    $msg[] = $cmd >> 16 & 0xFF;
    $msg[] = $cmd >> 8 & 0xFF;
    $msg[] = $cmd & 0xFF;
    $res=0;
    while ($retries) {
        $res = mb_send_command($bus, $msg);
        $retries--;
        if (is_array($res) && sizeof($res)) {
        //    print_r($res);
            break;
        }
        if ($DEBUG) {
            print("\r\n<span style='color:red'>Modbus Command
FAILED</span>\r\n");
        }

        //print_r($res);
    }
    if (is_array($res)) {
        // if the result was successful
        if ($res[1]==0x10) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}
```

```
/**
  * mb_set_uint32_le_0x10 - set 32-bit uint little endian via function 10h
  *
  * Set a little endian UINT32 register via function 0x10h
  *
  * @param int $id - Modbus device ID
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $cmd - Value
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */
function mb_set_uint32_le_0x10($id, $bus, $reg, $cmd, $retries)
{
    global $DEBUG;
    $base_addr = $reg;
    $num_bytes = 4;
    $msg = array(
        $id,
        0x10
    );
    $msg[] = $base_addr >> 8;
    $msg[] = $base_addr & 0xFF;
    $msg[] = 0;
    $msg[] = 2;
    $msg[] = 4;
    $msg[] = $cmd >> 8 & 0xFF;
    $msg[] = $cmd & 0xFF;
    $msg[] = $cmd >> 24 & 0xFF;
    $msg[] = $cmd >> 16 & 0xFF;

    $res=0;

    while ($retries) {
        $res = mb_send_command($bus, $msg);
        $retries--;
        if (is_array($res) && sizeof($res)) {
        //    print_r($res);
            break;
        }
        if ($DEBUG) {
            print("\r\n<span style='color:red'>Modbus Command
FAILED</span>\r\n");
        }

        //print_r($res);
    }
    if (is_array($res)) {
```

```
        // if the result was successful
        if ($res[1]==0x10) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}

/**
  * mb_set_float_be_0x10 - set 32-bit big endian float via function 10h
  *
  * Set a big endian float register via function 0x10h
  *
  * @param int $id - Modbus device ID
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $val - Value
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */
function mb_set_float_be_0x10($id,$bus,$reg,$val,$retries) {
    global $DEBUG;
    $base_addr=$reg;
    $num_bytes=4;
    $msg=array($id,16);
    $msg[]=$base_addr >> 8;
    $msg[]=$base_addr & 0xFF;
    $msg[]=0;
    $msg[]=2;
    $msg[]=4;
    $val=floatval($val);
    $cmd = ieee754toint($val);



    $msg[]=$cmd >> 24 & 0xFF;
    $msg[]=$cmd >> 16 & 0xFF;

    $msg[]=$cmd >> 8 & 0xFF;
    $msg[]=$cmd & 0xFF;
    if ($DEBUG) {
      print("<span class='info'>[MODBUS]</span>\r\nSet Register Send:\r\n");
      print_r($msg);
      print ("Retries=" . $retries);
    }
    $res=0;
    while ($retries) {
        $res=mb_send_command($bus,$msg);
        $retries--;
```

```
        if (is_array($res) && sizeof($res)) {

            break;
        }
    }

    if (is_array($res)) {
        // if the result was successful
        if ($res[1]==0x10) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}

/**
  * mb_set_float_le_0x10 - set 32-bit little endian float via function 10h
  *
  * Set a little endian float register via function 0x10h
  *
  * @param int $id - Modbus device ID
  * @param int $bus - Modbus bus ID (0 for Rs-485, 1 for RS-482 2, 2-11 for
TCP channels 0-10)
  * @param int $reg - Register ID
  * @param int $val - Value
  * @oaran unt $retries - number of retries
  *
  * @return int 0 for failure, < 0 if modbus exception, 1 if successful
  */
function mb_set_float_le_0x10($id,$bus,$reg,$val,$retries) {
    global $DEBUG;
    $base_addr=$reg;
    $num_bytes=4;
    $msg=array($id,16);
    $msg[]=$base_addr >> 8;
    $msg[]=$base_addr & 0xFF;
    $msg[]=0;
    $msg[]=2;
    $msg[]=4;
    $val=floatval($val);
    $cmd = ieee754toint($val);
    $msg[]=$cmd >> 8 & 0xFF;
    $msg[]=$cmd & 0xFF;

    $msg[]=$cmd >> 24 & 0xFF;
    $msg[]=$cmd >> 16 & 0xFF;

    if ($DEBUG) {
      print("<span class='info'>[MODBUS]</span>\r\nSet Register Send:\r\n");
      print_r($msg);
      print ("Retries=" . $retries);
```

```
    }

    while ($retries) {
        $res=mb_send_command($bus,$msg);
        $retries--;
        if (is_array($res) && sizeof($res)) {

            break;
        }
    }
    if (is_array($res)) {
        // if the result was successful
        if ($res[1]==0x10) {
            return 1;
        } else return -$res[2]; // return modbus exception code
    }
    return $res;
}

?>
```