MODBUS registers read position via port 502.

Application Description:
This example shows the use of internal MODBUS registers 400-405 on the Universal-Robots to read the robot position from external device.

The application uses the MODBUS server running at port 502 on the Universal-Robots. The MODBUS server has a range of register available with dedicated content and register 128 to 255 are for general purpose use. In this example register 400 to 405 will be used because they contain the values of the robot position as

400 = TCP-x in tenth of mm (in base frame)
401 = TCP-y in tenth of mm (in base frame)
402 = TCP-z in tenth of mm (in base frame)
403 = TCP-rx in mrad (in base frame)
404 = TCP-ry in mrad (in base frame)
405 = TCP-rz in mrad (in base frame)

A list of the MODBUS and register can be found at this link UR Modbus page

The commands are send from a external host with Python code.
Function description:
The external host (PC) is connected to the UR via Ethernet and access the MODBUS registers on the UR on TCP port 502.

For the MODBUS protcol and meaning of the raw data there are more informations at this link MODBUS registers Input and Output handling via port 502.

Program description:
The external host reads the register 400-405 every 5 seconds and convert the raw data to readable data in mm.

(Further Below is the same program with more conversation data shown on the screen for evaluation purpose).

Program code:

No program on UR.

Host program in Python code.

```
# Echo client program

import socket

import time


HOST = "192.168.0.9" # The remote host

PORT_502 = 502


print "Starting Program"


count = 0

home_status = 0

program_run = 0
```

```
while (True):
 if program_run == 0:
  try:
   s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   s.settimeout(10)
   s.connect((HOST, PORT_502))
   time.sleep(0.05)
   print ""
   reg_400 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x90\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_400 = s.recv(1024)
   reg_400 = reg_400.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_400 = reg_400.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_400 == "":
     reg_400 = "0000"
     reg_400_i = int(reg_400,16)
   if reg_400_i < 32768:
     reg_400_f = float(reg_400_i)/10
   if reg_400_i > 32767:
     reg_400_i = 65535 - reg_400_i
     reg_400_f = float(reg_400_i)/10*-1
   print "X = ",reg_400_f


   reg_401 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x91\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_401 = s.recv(1024)
   reg_401 = reg_401.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_401 = reg_401.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_401 == "":
     reg_401 = "0000"
     reg_401_i = int(reg_401,16)
   if reg_401_i < 32768:
     reg_401_f = float(reg_401_i)/10
```

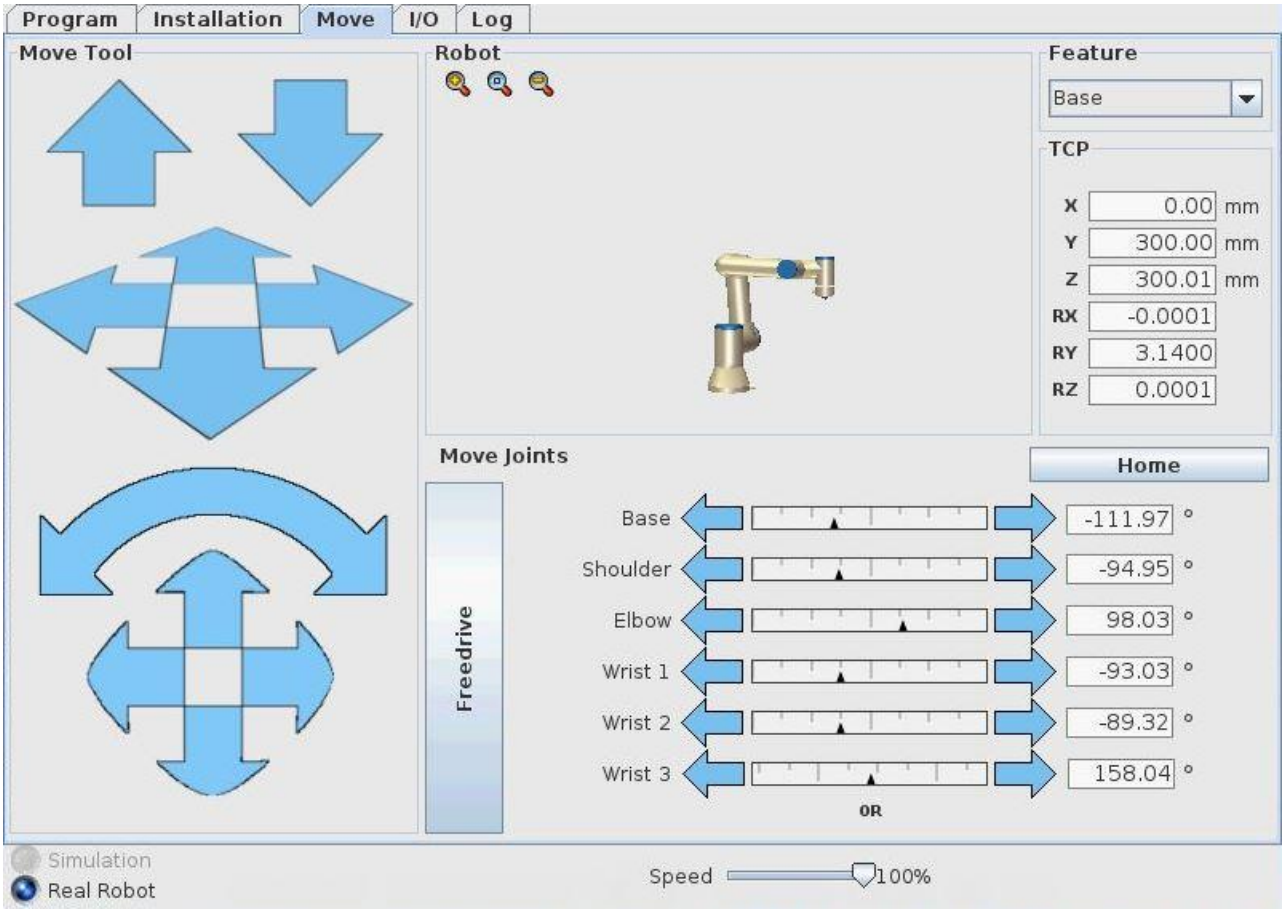
```
   if reg_401_i > 32767:
     reg_401_i = 65535 - reg_401_i
     reg_401_f = float(reg_401_i)/10*-1
   print "Y = ",reg_401_f


   reg_402 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x92\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_402 = s.recv(1024)
   reg_402 = reg_402.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_402 = reg_402.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_402 == "":
     reg_402 = "0000"
     reg_402_i = int(reg_402,16)
   if reg_402_i < 32768:
     reg_402_f = float(reg_402_i)/10
   if reg_402_i > 32767:
     reg_402_i = 65535 - reg_402_i
   reg_402_f = float(reg_402_i)/10*-1
   print "Z = ",reg_402_f


   reg_403 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x93\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_403 = s.recv(1024)
   reg_403 = reg_403.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_403 = reg_403.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_403 == "":
     reg_403 = "0000"
     reg_403_i = int(reg_403,16)
   if reg_403_i < 32768:
     reg_403_f = float(reg_403_i)/1000
   if reg_403_i > 32767:
     reg_403_i = 65535 - reg_403_i
     reg_403_f = float(reg_403_i)/1000*-1
   print "Rx = ",reg_403_f
```

```
   reg_404 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x94\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_404 = s.recv(1024)
   reg_404 = reg_404.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_404 = reg_404.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_404 == "":
     reg_404 = "0000"
     reg_404_i = int(reg_404,16)
   if reg_404_i < 32768:
     reg_404_f = float(reg_404_i)/1000
   if reg_404_i > 32767:
     reg_404_i = 65535 - reg_404_i
     reg_404_f = float(reg_404_i)/1000*-1
   print "Ry = ",reg_404_f


   reg_405 = ""
   s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x95\x00\x01") #request data from
register 128-133 (cartisian data)
   reg_405 = s.recv(1024)
   reg_405 = reg_405.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
   reg_405 = reg_405.encode("hex") #convert the data from \x hex notation to plain
hex
   if reg_405 == "":
     reg_405 = "0000"
     reg_405_i = int(reg_405,16)
   if reg_405_i < 32768:
     reg_405_f = float(reg_405_i)/1000
   if reg_405_i > 32767:
     reg_405_i = 65535 - reg_405_i
     reg_405_f = float(reg_405_i)/1000*-1
   print "Rz = ",reg_405_f


   time.sleep(5.00)
   home_status = 1
   program_run = 0
```

```
   s.close()
 except socket.error as socketerror:
   print("Error: ", socketerror)
print "Program finish"
```

Program run:

The robot is in this position – Notice the X, Y, Z, Rx, Ry, Rz values in the Move screen.



Below is the printout on the screen with the X, Y, Z, Rx, Ry, Rz values when the Python program is executed. The values are collected from the MODBUS registers and correspond with the actual values of the robot position. The X, Y, Z are in mm and Rx, Ry, Rz are in radians.

>>>
Starting Program

X = 0.0
Y = 300.0
Z = 300.0
Rx = 0.0
Ry = 3.14
Rz = 0.0

Notes:
For the purpose of monitor and evaluation of the data is below the same program, but with more data printout on the screen so it can be observed how the data are converted to readable data in mm and radians.

Program code:

No program on UR.

Host program in Python code with more data printout.

```
# Echo client program
import socket
import time

HOST = "192.168.0.9" # The remote host
PORT_502 = 502

print "Starting Program"

count = 0
home_status = 0
program_run = 0

while (True):
 if program_run == 0:
   try:
     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
     s.settimeout(10)
     s.connect((HOST, PORT_502))
     time.sleep(0.05)

     print ""
     print "Request reg 400"
     print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x90 x00 x01"
     reg_400 = ""
     s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x90\x00\x01") #request data fro
m register 128-133 (cartisian data)
     print "Modbus command send to read reg 400"
     print "Received"
     reg_400 = s.recv(1024)
     print repr(reg_400) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P
     print ""
     reg_400 = reg_400.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
```

```
     reg_400 = reg_400.encode("hex") #convert the data from \x hex notation to pla
in hex
     if reg_400 == "":
       reg_400 = "0000"
     print reg_400
     print ""
     reg_400_i = int(reg_400,16)
     print reg_400_i
     if reg_400_i < 32768:
       reg_400_f = float(reg_400_i)/10
     if reg_400_i > 32767:
       reg_400_i = 65535 - reg_400_i
       reg_400_f = float(reg_400_i)/10*-1
     print "X = ",reg_400_f


     print ""
     print "Request reg 401"
     print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x91 x00 x01"
     reg_401 = ""
     s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x91\x00\x01") #request data fro
m register 128-133 (cartisian data)
     print "Modbus command send to read reg 401"
     print "Received"
     reg_401 = s.recv(1024)
     print repr(reg_401) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P
     print ""
     reg_401 = reg_401.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
     reg_401 = reg_401.encode("hex") #convert the data from \x hex notation to pla
in hex
     if reg_401 == "":
       reg_401 = "0000"
     print reg_401
     print ""
     reg_401_i = int(reg_401,16)
     print reg_401_i
     if reg_401_i < 32768:
       reg_401_f = float(reg_401_i)/10
```

```
     if reg_401_i > 32767:
       reg_401_i = 65535 - reg_401_i
       reg_401_f = float(reg_401_i)/10*-1
     print "Y = ",reg_401_f


     print ""
     print "Request reg 402"
     print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x92 x00 x01"
     reg_402 = ""
     s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x92\x00\x01") #request data fro
m register 128-133 (cartisian data)
     print "Modbus command send to read reg 402"


     print "Received"
     reg_402 = s.recv(1024)
     print repr(reg_402) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P
     print ""
     reg_402 = reg_402.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
     reg_402 = reg_402.encode("hex") #convert the data from \x hex notation to pla
in hex
     if reg_402 == "":
       reg_402 = "0000"
     print reg_402
     print ""
     reg_402_i = int(reg_402,16)
     print reg_402_i
     if reg_402_i < 32768:
       reg_402_f = float(reg_402_i)/10
     if reg_402_i > 32767:
     reg_402_i = 65535 - reg_402_i
     reg_402_f = float(reg_402_i)/10*-1
     print "Z = ",reg_402_f


     print ""
     print "Request reg 403"
     print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x93 x00 x01"
```

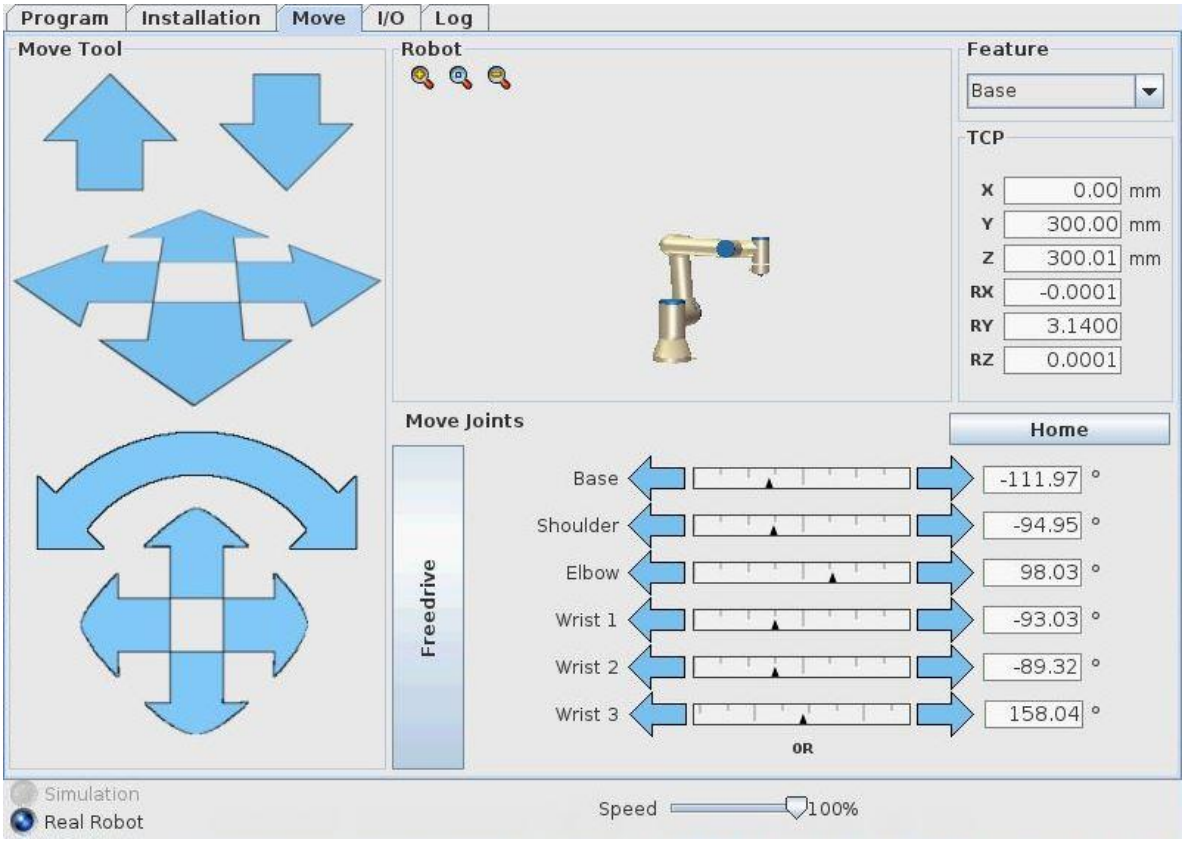
```
     reg_403 = ""

     s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x93\x00\x01") #request data fro
m register 128-133 (cartisian data)

     print "Modbus command send to read reg 403"

     print "Received"

     reg_403 = s.recv(1024)

     print repr(reg_403) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P

     print ""

     reg_403 = reg_403.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")

     reg_403 = reg_403.encode("hex") #convert the data from \x hex notation to pla
in hex

     if reg_403 == "":

       reg_403 = "0000"

     print reg_403

     print ""

     reg_403_i = int(reg_403,16)

     print reg_403_i

     if reg_403_i < 32768:

       reg_403_f = float(reg_403_i)/1000

     if reg_403_i > 32767:

     reg_403_i = 65535 - reg_403_i

     reg_403_f = float(reg_403_i)/1000*-1

     print "Rx = ",reg_403_f


     print ""

     print "Request reg 404"

     print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x94 x00 x01"

     reg_404 = ""

     s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x94\x00\x01") #request data fro
m register 128-133 (cartisian data)

     print "Modbus command send to read reg 404"

     print "Received"

     reg_404 = s.recv(1024)

     print repr(reg_404) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P

     print ""

     reg_404 = reg_404.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")
```

```
    reg_404 = reg_404.strip("\x00\x04\x00\x00\x00\x05\x00\x03\x02")

    reg_404 = reg_404.encode("hex") #convert the data from \x hex notation to pla
in hex

    if reg_404 == "":

      reg_404 = "0000"

    print reg_404

    print ""

    reg_404_i = int(reg_404,16)

    print reg_404_i

    if reg_404_i < 32768:

      reg_404_f = float(reg_404_i)/1000

    if reg_404_i > 32767:

      reg_404_i = 65535 - reg_404_i

      reg_404_f = float(reg_404_i)/1000*-1

    print "Ry = ",reg_404_f


    print ""

    print "Request reg 405"

    print "Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x95 x00 x01"

    reg_405 = ""

    s.send ("\x00\x04\x00\x00\x00\x06\x00\x03\x01\x95\x00\x01") #request data fro
m register 128-133 (cartisian data)

    print "Modbus command send to read reg 405"

    print "Received"

    reg_405 = s.recv(1024)

    print repr(reg_405) #Print the receive data in \x hex notation (notice that d
ata above 32 hex will berepresented at the ascii code e.g. 50 will show P

    print ""

    reg_405 = reg_405.replace ("\x00\x04\x00\x00\x00\x05\x00\x03\x02", "")

    reg_405 = reg_405.strip("\x00\x04\x00\x00\x00\x05\x00\x03\x02")

    reg_405 = reg_405.encode("hex") #convert the data from \x hex notation to pla
in hex

    if reg_405 == "":

      reg_405 = "0000"

    print reg_405

    print ""

    reg_405_i = int(reg_405,16)

    print reg_405_i
```

```
      if reg_405_i < 32768:
        reg_405_f = float(reg_405_i)/1000
      if reg_405_i > 32767:
        reg_405_i = 65535 - reg_405_i
        reg_405_f = float(reg_405_i)/1000*-1
      print "Rz = ",reg_405_f

      time.sleep(5.00)
      home_status = 1
      program_run = 0
      s.close()
    except socket.error as socketerror:
      print("Error: ", socketerror)
print "Program finish"
```

Program run:

The robot is in this position – Notice the X, Y, Z, Rx, Ry, Rz values in the Move screen.



Below is the printout on the screen with the X, Y, Z, Rx, Ry, Rz values when the Python program is executed. The values are collected from the MODBUS registers and correspond with the actual values of the robot position. The X, Y, Z are in mm and Rx, Ry, Rz are in radians.

```
>>>
Starting Program

Request reg 400
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x90 x00 x01
Modbus command send to read reg 400
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x00\x00’

0000

0
X = 0.0

Request reg 401
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x91 x00 x01
Modbus command send to read reg 401
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x0b\xb8’

0bb8

3000
Y = 300.0

Request reg 402
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x92 x00 x01
Modbus command send to read reg 402
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x0b\xb8’

0bb8

3000
Z = 300.0

Request reg 403
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x93 x00 x01
Modbus command send to read reg 403
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x00\x00’

0000

0
Rx = 0.0

Request reg 404
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x94 x00 x01
Modbus command send to read reg 404
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x0cD’

0c44

3140
Ry = 3.14

Request reg 405
Sending: x00 x04 x00 x00 x00 x06 x00 x03 x01 x95 x00 x01
Modbus command send to read reg 405
Received
‘\x00\x04\x00\x00\x00\x05\x00\x03\x02\x00\x00’
```

0000

0
Rz = 0.0