How to use HiTechnic sensor multiplexer in NXT++? Mindstorms NXT 2.0

f you are advanced user of NXT++, and want to add code to your existing project just go to point (II).

(I) Quick start in Visual Studio 2010 for begginers

1) install microsoft visual studio 2010 with visual c++,
2) install drivers of NXT_32 or NXT_64 depending on your system,
3) if you wish to use bluetooth connection with NXT brick, install bluetooth drivers (in case of problem with bluetooth drivers, uninstall drivers provided by computer dealer and use standard windows drivers,
4) download library http://wmii.uwm.edu.pl/~artem/nxtpp0-6-2.zip (the library include color sensor support and sensor multiplexer (SMUX) support, the original one without our changes is available in http://sourceforge.net/projects/nxtpp/),
5) extract file nxtpp0-6-2.zip, and move the folder nxtpp0-6-2 directly to drive C:\ (because the solution is configured on this path, if you have different path just change links in your project,
6) open file: C:\nxtpp0-6-2\FINAL\examplesMUX\vs10\re4.sln
if you have different path to re4.sln file, open solution re4.sln from your path, and in re4 properties, particularly in:
- configuration properties/C/C++/general Additional Include Directories/
and
- configuration properties/Linker/General/Additional Library Direcories/
change the links on yours,
7) plug in your sensors to proper sensor multiplexer subports,
8) turn on the power supply of sensor multiplexer, and then turn on NXT brick,
7) connect your NXT brick via cable or bluetooth and run the solution. In our example from file C:\nxtpp0-6-2\FINAL\examplesMUX\vs10\re4\re4\main.cpp we have fixed Solution Configurations as Release and Solution Platforms as Win32. And we made assumption that sensor multiplexer is connected to port IN_1, change if you use different port.

(II) Usage for advanced users:

Paste the following code into the file NXT++.h:

```
/*! Sets a sensor multiplexer (SMUX) in a specified port,
\param port can be the numbers 0-3 or IN_1, IN_2, IN_3, or IN_4.*/
void SetSMUX(Comm::NXTComm* comm, int port);

/*! Retrieves the value of a sensor connected to specified subport of sensor multiplexer,
\param port The port in which the sensor multiplexer is connected, can be the numbers 0-3 or
IN_1, IN_2, IN_3, or IN_4.
\param subport - is the number of sensor multiplexer subport, can by 1, 2, 3 or 4.
\param SensorType is the type of sensor, can be the number, 1 - for digital sensor, 2 - for
analog sensor,
This function return the sensor current values in raw mode,*/
int GetSMUXvalue(Comm::NXTComm* comm, int port,  int subport,  int SensorType);
```

Paste the following code into the file nxt++.cpp:

```
void NXT::Sensor::SetSMUX(Comm::NXTComm* comm, int port)
{
  ViUInt8 directCommandBuffer0[] = { 0x05, port, 0x0B, 0x80 };
  comm->SendDirectCommand( false, reinterpret_cast< ViByte* >( directCommandBuffer0 ),
sizeof( directCommandBuffer0 ), NULL, 0);
  Wait(50);

  ViUInt8 directCommandBuffer[] = {0x0F, port, 0x03, 0x00, 0x10, 0x20, 0x00};
```

```
  ViUInt8 directCommandBuffer2[] = {0x0F, port, 0x03, 0x00, 0x10, 0x20, 0x01};
  ViUInt8 directCommandBuffer3[] = {0x0F, port, 0x03, 0x00, 0x10, 0x20, 0x02};

  for(int i=0;i<2;i++)
  {
  comm->SendDirectCommand( false, reinterpret_cast< ViByte* >( directCommandBuffer ),
sizeof( directCommandBuffer ), NULL, 0);
  Wait(50);
  comm->SendDirectCommand( false, reinterpret_cast< ViByte* >( directCommandBuffer2 ),
sizeof( directCommandBuffer2 ), NULL, 0);
  Wait(500);
  comm->SendDirectCommand( false, reinterpret_cast< ViByte* >( directCommandBuffer3 ),
sizeof( directCommandBuffer3 ), NULL, 0);
  Wait(50);
  }
 }

int NXT::Sensor::GetSMUXvalue(Comm::NXTComm* comm, int port, int subport, int SensorType)
{
  int bytesRead = 0;
  do {
  ViUInt8 subportB;
  if(SensorType==1)
   {
    if(subport==1)
    {
     subportB=0x40;
    }
    if(subport==2)
    {
     subportB=0x50;
    }
    if(subport==3)
    {
     subportB=0x60;
    }
    if(subport==4)
    {
     subportB=0x70;
    }
   }
   if(SensorType==2)
   {
    if(subport==1)
    {
     subportB=0x36;
    }
    if(subport==2)
    {
     subportB=0x38;
    }
    if(subport==3)
    {
     subportB=0x3A;
    }
    if(subport==4)
    {
     subportB=0x3C;
    }
   }

  ViUInt8 directCommandBuffer4[] = {0x0F, port, 0x02, 0x01, 0x10, subportB};

  ViUInt8 responseBuffer4[] = {1, 1};
  comm->SendDirectCommand( true, reinterpret_cast< ViByte* >( directCommandBuffer4 ),
```

```
sizeof( directCommandBuffer4 ),
         reinterpret_cast< ViByte* >( responseBuffer4 ), sizeof( responseBuffer4));
   Wait(10);
        bytesRead = LSGetStatus(comm, port);
       } while (bytesRead < 1);

       ViUInt8 directCommandBuffer5[] = {0x10, port};
       ViUInt8 responseBuffer5[] = {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};


       comm->SendDirectCommand( true, reinterpret_cast< ViByte* >( directCommandBuffer5 ),
sizeof( directCommandBuffer5 ),
             reinterpret_cast< ViByte* >( responseBuffer5 ), sizeof( responseBuffer5 ));

       return (int)responseBuffer5[3];
}
```

If you use Visual Studio, open and rebuild solution NXT++.sln

And use nxt sensor in main.cpp of your project:

Exemplary usage:

```
/*the sensor multiplexer (SMUX) specified in port IN_1*/

// Tells the NXT brick to set sensor multiplexer in port IN_1,
this operation should be done only once, before reading from sensor multiplexer,
NXT::Sensor::SetSMUX(&comm, IN_1);

//how to read data from sensors connected to sensor multiplexer?
//function GetSMUXvalue return the value of chosen multiplexer subport,
int measurement=NXT::Sensor::GetSMUXvalue(&comm, IN1, subport, stype);

//parameter subport could be:
//'1' - setting to read from subport 1
//'2' - setting to read from subport 2
//'3' - setting to read from subport 3
//'4' - setting to read from subport 4

//parameter stype could be:
//'1' - setting to read from digital sensor
//'2' - setting to read from analog sensor

//exemplary digital sensors: ultrasonic sensor, compass sensor
//exemplary analog sensors: touch sensor, light sensor

//If we want to read data from digital sensor connected to subport 4 of sensor multiplexer
connected to port IN_1 of NXT brick, we coud use the following code,
measurement=NXT::Sensor::GetSMUXvalue(&comm, IN1, 4, 1);

//If we want to read data from analog sensor connected to subport 3 of sensor multiplexer
connected to port IN_2 of NXT brick, we coud use the following code,
measurement=NXT::Sensor::GetSMUXvalue(&comm, IN2, 3, 2);
```

To see all functionalities of sensor multiplexer, you can put the following code to your solution main.cpp file:

```
#include "NXT++.h"
#include <conio.h>
Comm::NXTComm comm;

int main()
{
 std::cout << "\naquiring signal ... ";
 int infinity_loop_pointer=1;

 if(NXT::OpenBT(&comm))/*initialize the NXT via cable or bluetooth and continue if it
succeeds*/
```

```
 //if(NXT::Open(&comm))
 {
  std::cout<< "signal found";
  std::cout<<"\nbattery Level = "<<NXT::BatteryLevel(&comm)/100<<" percent";
  std::cout<<"\navailable flash memory = "<<NXT::GetAvailableFlash(&comm);

  // In this example sensor multiplexer is connected to port IN_1
   NXT::Sensor::SetSMUX(&comm, IN_1);

  char decyzja;
  std::cout<<"\nPress S to get sensor values from sensor multiplexer ports";
  std::cout<<"\nPress K to stop the program";

  do
   {//beggining of main inifinity loop
    if(kbhit()==true)
     {
      decyzja=getch();

      if(decyzja=='K' || decyzja=='k')
       {
        break;
       }
      if(decyzja=='S' || decyzja=='s')
       {
       /*NXT::Sensor::GetSMUXvalue(&comm, port, subport, stype);
       \param port - the port of sensor multiplexer connection with NXT brick IN_1,
       IN_2, IN_3 or IN_4,
       \param subport - multiplexer subport number - 1,2,3 or 4,
       \param stype - sensor type: 1 - digital (for instance ultrasonic sensor, compass
        sensor), 2 - analog (for instance touch sensor, light sensor),*/

     // In this example sensor multiplexer is connected to port IN_1
     std::cout<<"\n\nReading for digital sensors:";
     std::cout<<"\nreading from subport 1 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 1, 1);
     std::cout<<"\nreading from subport 2 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 2, 1);
     std::cout<<"\nreading from subport 3 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 3, 1);
     std::cout<<"\nreading from subport 4 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 4, 1);

     std::cout<<"\n\nReading for analog sensors:";
     std::cout<<"\nreading from subport 1 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 1, 2);
     std::cout<<"\nreading from subport 2 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 2, 2);
     std::cout<<"\nreading from subport 3 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 3, 2);
     std::cout<<"\nreading from subport 4 = "<<NXT::Sensor::GetSMUXvalue(&comm, IN_1, 4, 2);
        }
        continue;
       }
   }// the end of main inifinity loop
   while(infinity_loop_pointer!=2);

    NXT::StopProgram(&comm);
  }
   NXT::Close(&comm); //close the NXT
  return 0;
}

//the sensor multiplexer (SMUX) support for NXT++ developed by artem
http://wmii.uwm.edu.pl/~artem
//the original library without our changes is available in:
//http://sourceforge.net/projects/nxtpp/
```