Как дополнить компиляцией метаданных проект Visual Studio с примерами из SDK nanoCAD без расширения Qt VS Tools

Рекомендуется использовать среду **Visual Studio 2019, toolset v142** и **Qt 5.15.1**.

1. Соберите проект HelloQt, который находится в $(NanoCadSDK)\ samples\NRX\HelloQt, где $(NanoCadSDK) – путь до папки SDK NanoCad.
2. Определите переменную среды QtRoot со значением, равным пути до папки с файлами Qt (например, C:\Qt\5.15.1\msvc2019_64).

Нажмите клавиши **Win+R** на клавиатуре, введите **sysdm.cpl** и нажмите Enter.

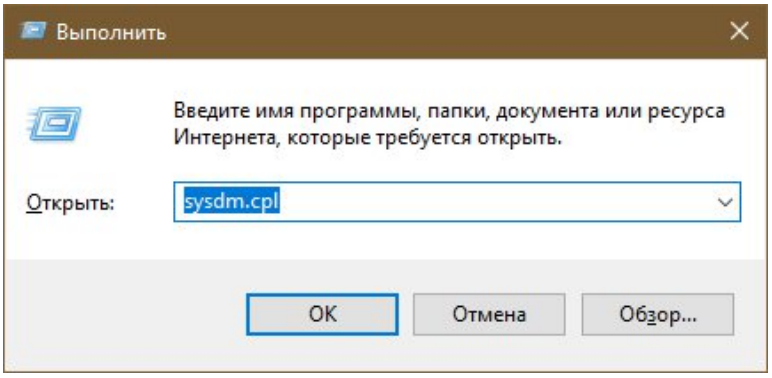

Рис. 1

На вкладке «Дополнительно» нажмите кнопку «Переменные среды…»

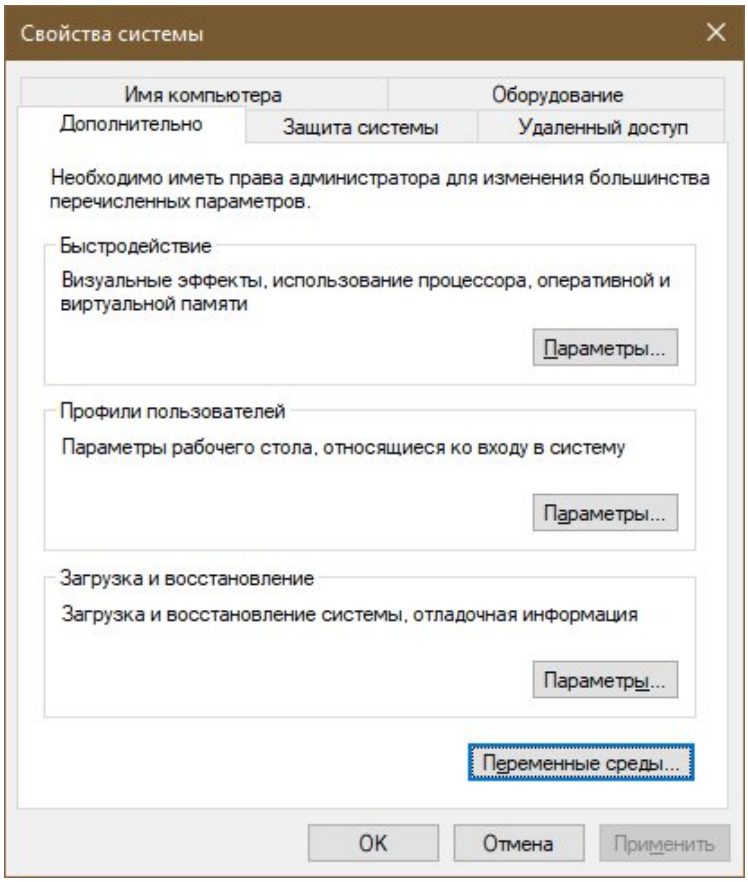

Рис. 2

В разделе «Переменные среды пользователя» (если требуется изменение только для текущего пользователя) или «Системные переменные» нажмите кнопку «Создать…»

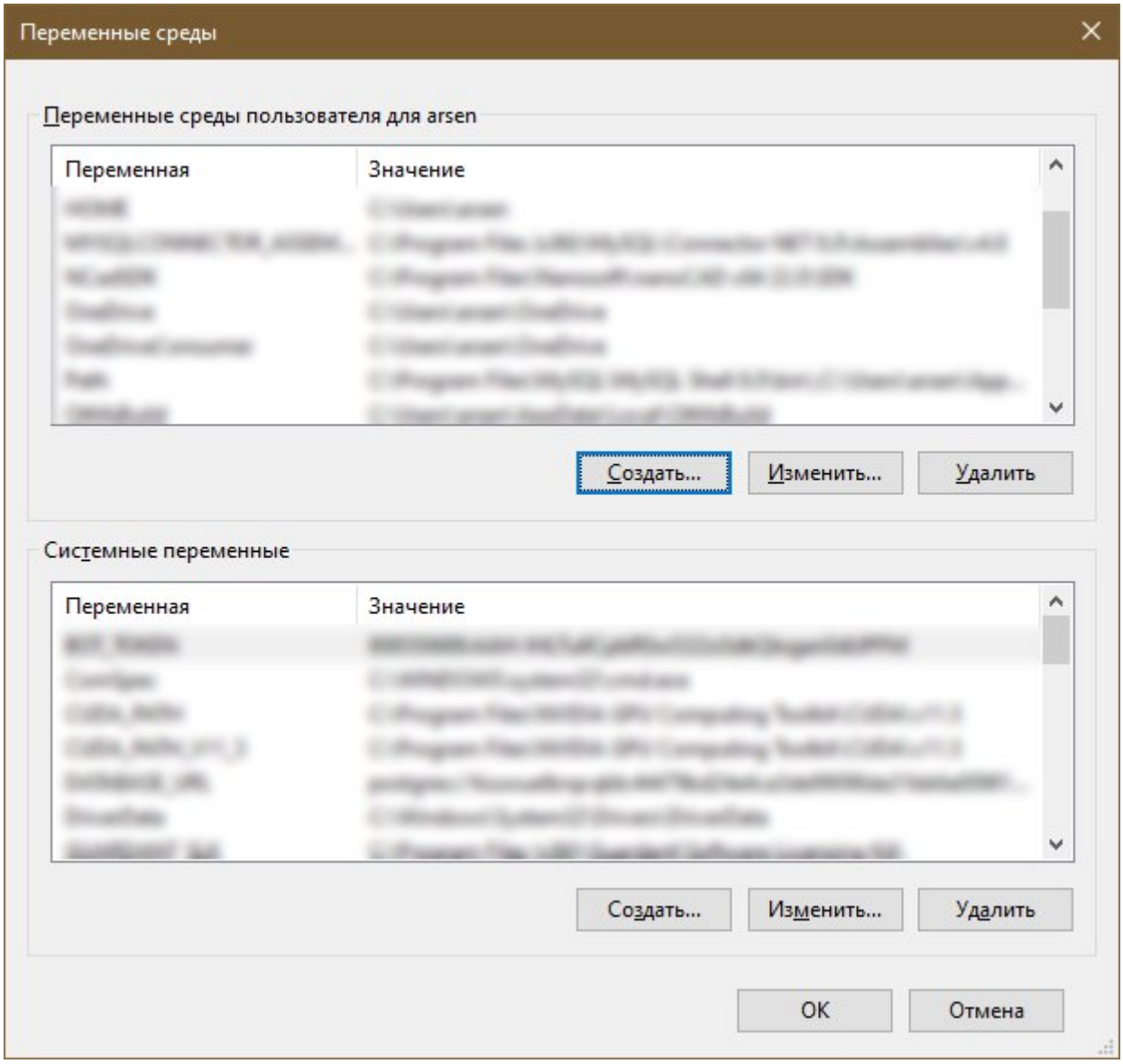

Рис. 3

Введите в поле «Имя переменной:» QtRoot, а в «Значение переменной:» путь до папки с файлами Qt (например, C:\Qt\5.15.1\msvc2019_64).

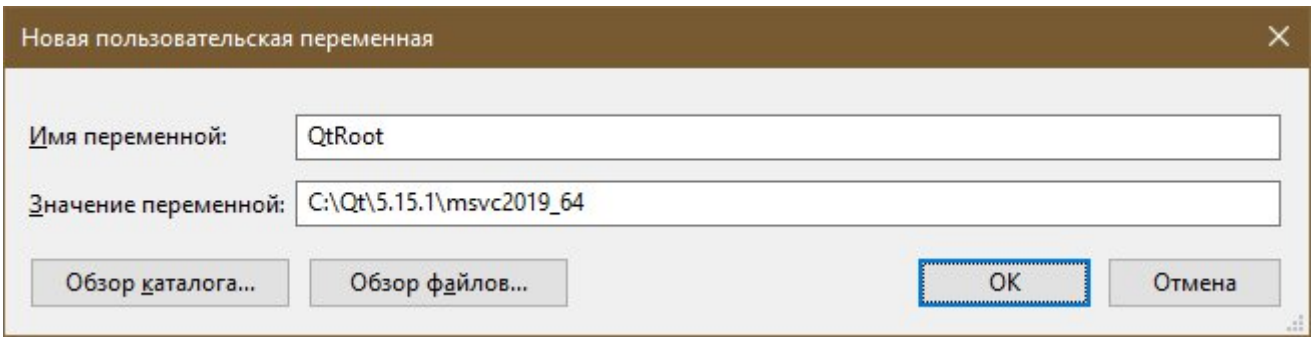

Рис. 4

3. Откройте файл проекта (*.vcxproj) в Visual Studio и добавьте необходимые файлы для создания Qt виджета (QtWidgetsClass.ui, QtWidgetsClass.h, QtWidgetsClass.cpp).
   *не забудьте добавить #include "stdafx.h" в начало файла QtWidgetsClass.cpp

4. Добавьте элемент ItemGroup в файл проект (*.vcxproj), в котором делается кастомная сборка и компиляция с вызовом moc.exe и uic.exe:

~~~~~~

```
  <ItemGroup Condition="'$(Configuration)|$(Platform)'=='Debug NCAD|x64' or
'$(Configuration)|$(Platform)'=='Release NCAD|x64'">
    <CustomBuild Include="QtWidgetsClass.ui">
      <AdditionalInputs>%(AdditionalInputs)</AdditionalInputs>
      <Command>$(QtRoot)\bin\uic.exe .\QtWidgetsClass.ui -o
"$(IntDir)\ui_QtWidgetsClass.h"</Command>
      <Message>UIC QtWidgetsClass.ui</Message>
      <Outputs>$(IntDir)\ui_QtWidgetsClass.h;%(Outputs)</Outputs>
    </CustomBuild>
    <CustomBuild Include="QtWidgetsClass.h">
      <AdditionalInputs>%(AdditionalInputs)</AdditionalInputs>
      <Command>$(QtRoot)\bin\moc.exe .\QtWidgetsClass.h -o
"$(IntDir)\moc_QtWidgetsClass.cpp"</Command>
      <Message>MOC QtWidgetsClass.h</Message>
      <Outputs>$(IntDir)\moc_QtWidgetsClass.cpp;%(Outputs)</Outputs>
    </CustomBuild>
    <ClCompile Include="QtWidgetsClass.cpp" />
  </ItemGroup>
```

~~~~~~

Элемент ItemGroup, приведенный выше должен быть расположен перед элементом ItemGroup, включающим в себя элементы ClInclude и ClCompile:

~~~~~~

```
<ItemGroup>
  <ClInclude Include="stdafx.h" />
</ItemGroup>
<ItemGroup>
  <ClCompile Include="HelloQt.cpp" />
  <ClCompile Include="stdafx.cpp">
    <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'=='Debug
NCAD|Win32'">Create</PrecompiledHeader>
    <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'=='Debug
NCAD|x64'">Create</PrecompiledHeader>
    <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'=='Release
NCAD|Win32'">Create</PrecompiledHeader>
    <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'=='Release
NCAD|x64'">Create</PrecompiledHeader>
```
~~~~~~

```
  </ClCompile>
</ItemGroup>
```

~~~~~~

После должен быть элемент ItemGroup с элемсентом ClCompile для файлов, сформированных компиляторами Qt:

~~~~~~

```
  <ItemGroup>
    <ClCompile Include="$(IntDir)\moc_QtWidgetsClass.cpp">
      <AdditionalOptions>/FIstdafx.h %(AdditionalOptions)</AdditionalOptions>
    </ClCompile>
  </ItemGroup>
```

~~~~~~

Файл проекта (*.vcxproj) можно отредактировать либо в текстовом редакторе, либо открыть проект в Visual Studio, нажать по нему правой кнопки мыши и выбрать пункт Unload Project.

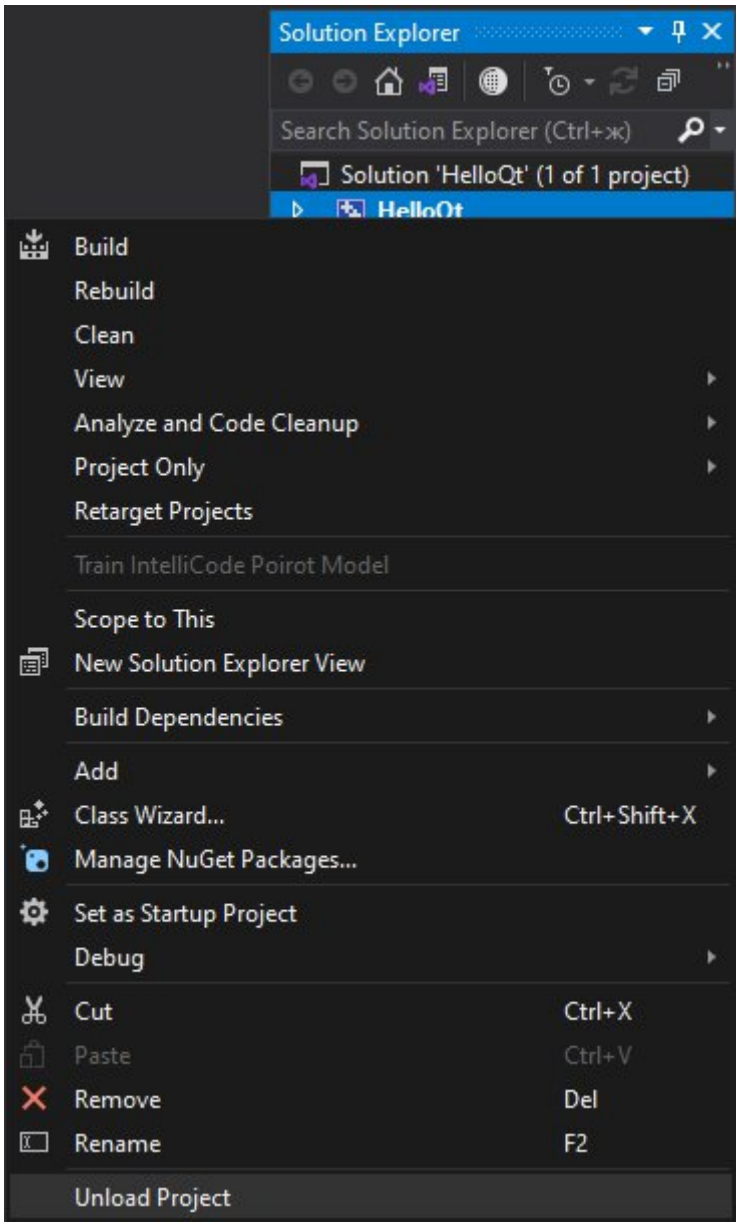

Рис. 5
~~~~~~

Затем отредактировать файл в среде Visual studio

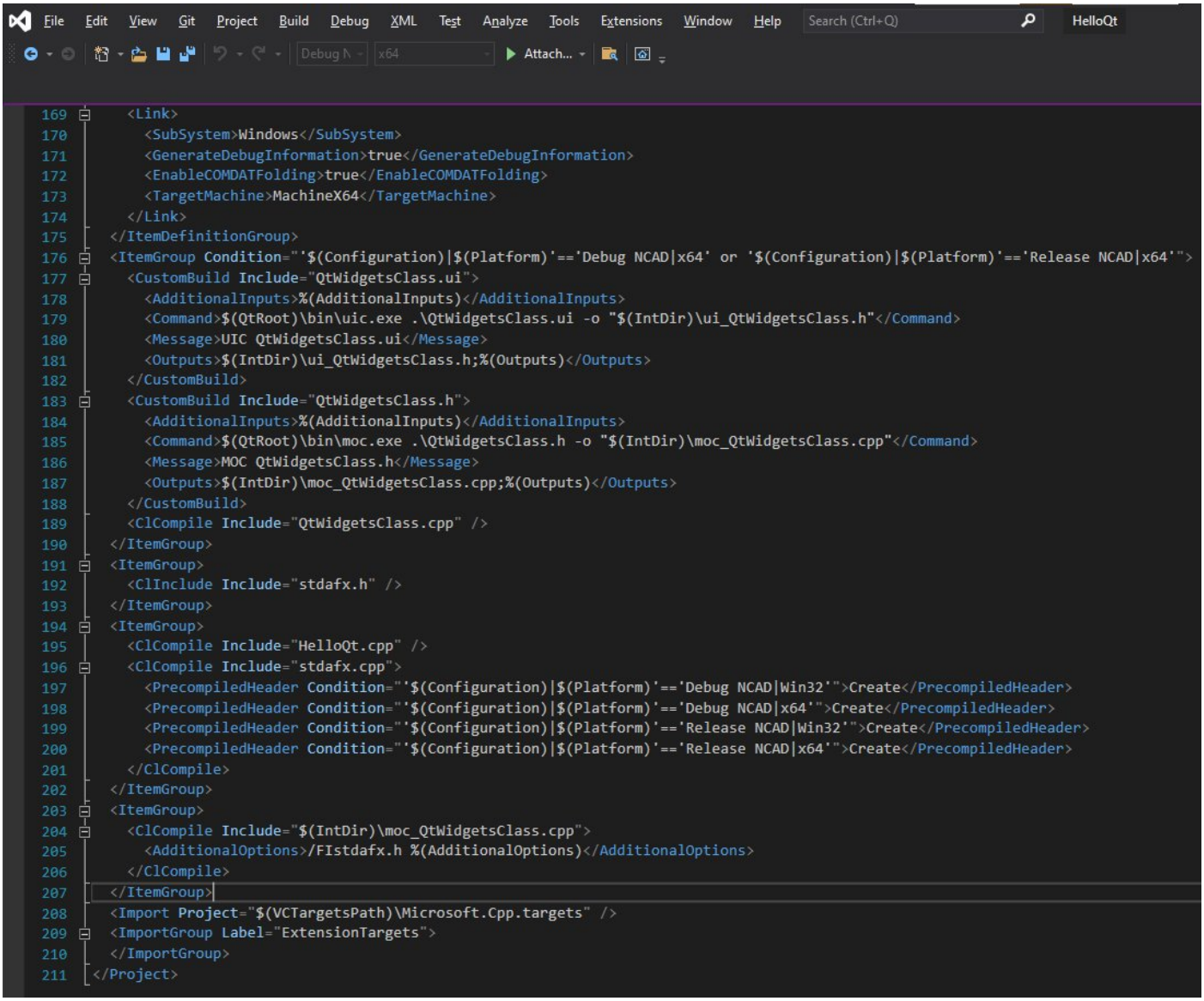

Рис. 6

и нажать Reload Project в контекстном меню проекта

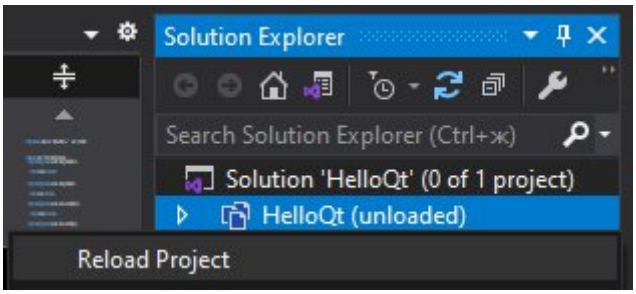

Рис. 7

5. Если вы редактировали файл проекта в текстовом редакторе, то сохраните файл и перезапишите проект в Visual Studio нажав кнопку «Save as…»:

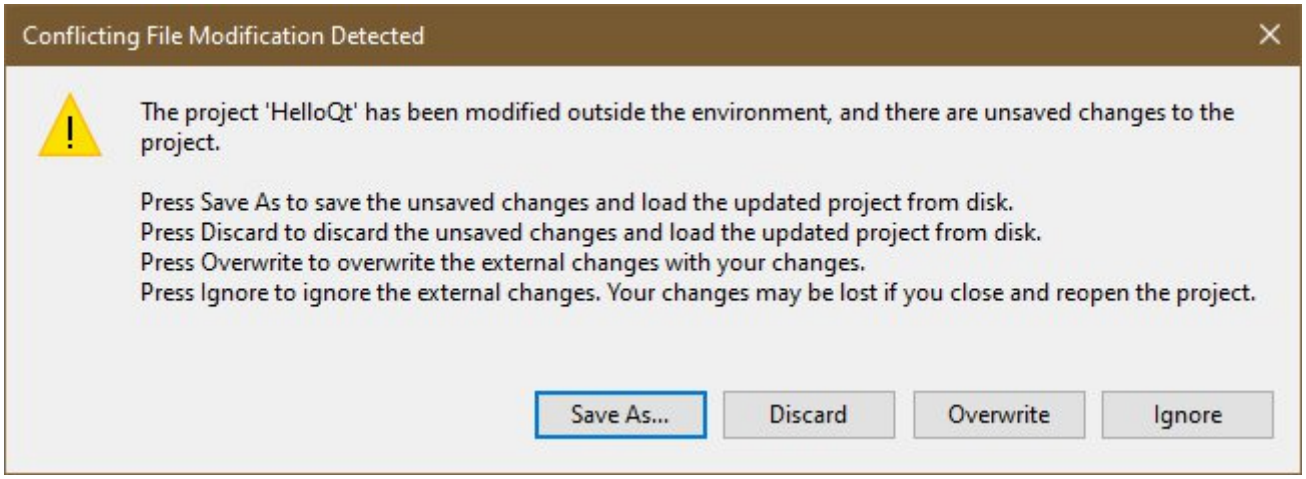

Рис. 8

6. Добавьте путь $(IntDir):
   Menu -> Project -> Properties -> Configuration Properties -> C/C++ -> General -> Additional Include Directories

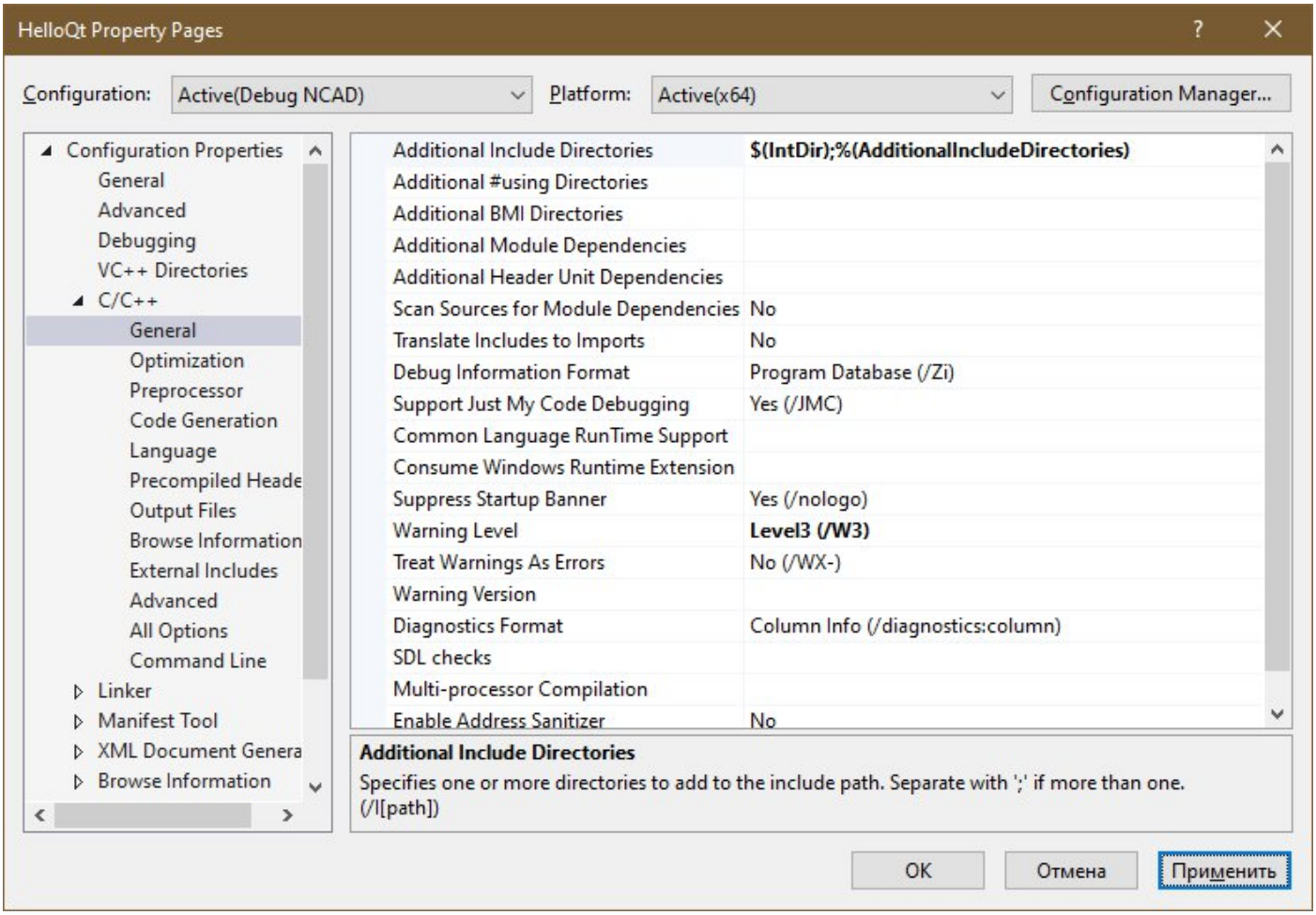

Рис. 9

7. Соберите проект.
8. Загрузите скомпилированный модуль в nanoCAD.

Команды приложения станут доступны в командной строке nanoCAD после загрузки.

Пример получившегося кода:

QtWidgetsClass.h

```
#pragma once

#include <QMainWindow>
#include "ui_QtWidgetsClass.h"

class QtWidgetsClass : public QMainWindow
{
	Q_OBJECT

public:
	QtWidgetsClass(QWidget *parent = nullptr);
	~QtWidgetsClass();

private:
	Ui::QtWidgetsClassClass ui;
};
```

QtWidgetsClass.cpp

```
#include "stdafx.h"
#include "QtWidgetsClass.h"

QtWidgetsClass::QtWidgetsClass(QWidget *parent)
	: QMainWindow(parent)
{
	ui.setupUi(this);
}

QtWidgetsClass::~QtWidgetsClass()
{}
```

stdafx.h

```
//
// Копирайт (C) 2019, ООО «Нанософт разработка». Все права защищены.
//
// Данное программное обеспечение, все исключительные права на него, его
// документация и сопроводительные материалы принадлежат ООО «Нанософт разработка».
// Данное программное обеспечение может использоваться при разработке и входить
// в состав разработанных программных продуктов при соблюдении условий
// использования, оговоренных в «Лицензионном договоре присоединения
// на использование программы для ЭВМ «Платформа nanoCAD»».
//
// Данное программное обеспечение защищено в соответствии с законодательством
// Российской Федерации об интеллектуальной собственности и международными
// правовыми актами.
//
// Используя данное программное обеспечение,  его документацию и
// сопроводительные материалы вы соглашаетесь с условиями использования,
// указанными выше.
//

#pragma once

//------------------------------------------------------------------------
//- 'DEBUG workaround' below prevents the MFC or ATL #include-s from pulling
//- in "Afx.h" that would force the debug CRT through #pragma-s.
#if defined(_DEBUG) && !defined(NC_FULL_DEBUG)
#define _DEBUG_WAS_DEFINED
#undef _DEBUG
#pragma message ("     Compiling MFC / STL / ATL header files in release mode.")
#endif
```

```
#pragma pack (push, 8)
#pragma warning(disable: 4786 4996)
//#pragma warning(disable: 4098)

//------------------------------------------------------------------------------
#define STRICT

#ifndef VC_EXTRALEAN
#define VC_EXTRALEAN            //- Exclude rarely-used stuff from Windows headers
#endif

//- Modify the following defines if you have to target a platform prior to the ones specified below.
//- Refer to MSDN for the latest info on corresponding values for different platforms.
#ifndef WINVER                  //- Allow use of features specific to Windows 95 and Windows NT 4 or
later.
#define WINVER 0x0501          //- Change this to the appropriate value to target Windows 98 and
Windows 2000 or later.
#endif

#ifndef _WIN32_WINNT            //- Allow use of features specific to Windows NT 4 or later.
#define _WIN32_WINNT 0x0501     //- Change this to the appropriate value to target Windows 2000 or
later.
#endif

#ifndef _WIN32_WINDOWS          //- Allow use of features specific to Windows 98 or later.
#define _WIN32_WINDOWS 0x0501   //- Change this to the appropriate value to target Windows Me or
later.
#endif

#ifndef _WIN32_IE               //- Allow use of features specific to IE 4.0 or later.
#define _WIN32_IE 0x0501        //- Change this to the appropriate value to target IE 5.0 or later.
#endif

//------------------------------------------------------------------------------
#define _ATL_APARTMENT_THREADED
#define _ATL_NO_AUTOMATIC_NAMESPACE
#define _ATL_CSTRING_EXPLICIT_CONSTRUCTORS    //- Some CString constructors will be explicit
//- Turns off ATL's hiding of some common and often safely ignored warning messages
#define _ATL_ALL_WARNINGS

//------------------------------------------------------------------------------
#include <afxwin.h>
#include <afxext.h>
#include <AtlBase.h>
#include <AtlCom.h>
using namespace ATL ;

#include "dbxHeaders.h"
#include "AcExtensionModule.h"

#include <QMessageBox>
#include <QHBoxLayout>
#include <QLabel>
#include <QLineEdit>
#include <QPushButton>
#include <QWindow>

#include "QtWidgetsClass.h"


#pragma pack (pop)

//------------------------------------------------------------------------------
#ifdef _DEBUG_WAS_DEFINED
#define _DEBUG
#undef _DEBUG_WAS_DEFINED
#endif
```

stdafx.cpp

```
//
// Копирайт (C) 2019, ООО «Нанософт разработка». Все права защищены.
//
// Данное программное обеспечение, все исключительные права на него, его
// документация и сопроводительные материалы принадлежат ООО «Нанософт разработка».
// Данное программное обеспечение может использоваться при разработке и входить
// в состав разработанных программных продуктов при соблюдении условий
// использования, оговоренных в «Лицензионном договоре присоединения
// на использование программы для ЭВМ «Платформа nanoCAD»».
//
// Данное программное обеспечение защищено в соответствии с законодательством
// Российской Федерации об интеллектуальной собственности и международными
// правовыми актами.
//
// Используя данное программное обеспечение,  его документацию и
// сопроводительные материалы вы соглашаетесь с условиями использования,
// указанными выше.
//

// stdafx.cpp : source file that includes just the standard includes
// HelloNRX.pch will be the pre-compiled header
// stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"

// TODO: reference any additional headers you need in STDAFX.H
// and not in this file
```

HelloQt.cpp

```
//
// Копирайт (C) 2019, ООО «Нанософт разработка». Все права защищены.
//
// Данное программное обеспечение, все исключительные права на него, его
// документация и сопроводительные материалы принадлежат ООО «Нанософт разработка».
// Данное программное обеспечение может использоваться при разработке и входить
// в состав разработанных программных продуктов при соблюдении условий
// использования, оговоренных в «Лицензионном договоре присоединения
// на использование программы для ЭВМ «Платформа nanoCAD»».
//
// Данное программное обеспечение защищено в соответствии с законодательством
// Российской Федерации об интеллектуальной собственности и международными
// правовыми актами.
//
// Используя данное программное обеспечение,  его документацию и
// сопроводительные материалы вы соглашаетесь с условиями использования,
// указанными выше.
//

#include "stdafx.h"

#include "hostUI.h"
#include "hostQt.h"

extern "C" __declspec(dllexport) bool showDialog(HWND parent)
{
  auto win = new QWinWidget(parent);
  win->showCentered();

  QMessageBox::about(win, "QtTests.dll", "Hello, hostQt.dll (based on
qtwinmigrate\\examples\\qtdll)!");

  delete win;

  return TRUE;
```

```
}

void helloQtModalDlgCmd()
{
  acutPrintf(L"\nHello, hostQt.dll!\n");

  showDialog(adsw_acadMainWnd());
}

hostUiPaletteSet* m_pPalSet = NULL;

HINSTANCE _hdllInstance =NULL ;
AC_IMPLEMENT_EXTENSION_MODULE(theArxDLL);

class helloQtPalette : public hostQtPalette
{
  DECLARE_DYNAMIC(helloQtPalette)

public:
  helloQtPalette() {};

  afx_msg void    OnSize(UINT nType, int cx, int cy)
  {
    if (m_pWinWidget)
    {
      HWND wnd = (HWND)m_pWinWidget->windowHandle()->winId();
      ::SetWindowPos(wnd, nullptr, 0, 0, cx, cy, SWP_NOZORDER);
    }
  }

  DECLARE_MESSAGE_MAP();
};

BEGIN_MESSAGE_MAP(helloQtPalette, hostQtPalette)
  //{{AFX_MSG_MAP(helloQtPalette)
  ON_WM_SIZE()
  //}}AFX_MSG_MAP
END_MESSAGE_MAP()

IMPLEMENT_DYNAMIC(helloQtPalette, hostQtPalette);

void helloQtPaletteCmd()
{
  if (!m_pPalSet) {
    CAcModuleResourceOverride ThisRes;
    m_pPalSet = new hostUiPaletteSet();
    m_pPalSet->Create(L"Test Qt Palette Set", WS_CHILD | WS_DLGFRAME | WS_VISIBLE, CRect(30, 50,
270, 300),
      CWnd::FromHandle(adsw_acadMainWnd()), PSS_SNAP | PSS_PROPERTIES_MENU | PSS_AUTO_ROLLUP |
PSS_CLOSE_BUTTON);
    m_pPalSet->EnableDocking(CBRS_ALIGN_ANY);
    m_pPalSet->RestoreControlBar();

    helloQtPalette* pPal = new helloQtPalette();
    pPal->Create(WS_CHILD | WS_VISIBLE, L"Test Qt Palette1", m_pPalSet, 0);
    m_pPalSet->AddPalette(pPal);

    QWidget* pPaletteWidget1 = pPal->paletteWidget();

    QVBoxLayout* vbox = new QVBoxLayout(pPaletteWidget1);
    vbox->setSpacing(5);
    vbox->setMargin(6);
    QPushButton* pb = new QPushButton("Qt command button", pPaletteWidget1);
    pb->setObjectName("pb");
    vbox->addWidget(pb);
    QLabel* label = new QLabel("Some label", pPaletteWidget1);
    label->setObjectName("label");
    vbox->addWidget(label);
    QLineEdit* le1 = new QLineEdit();
```

```
    le1->setObjectName("le1");
    vbox->addWidget(le1);
    QLineEdit* le2 = new QLineEdit();
    le2->setObjectName("le2");
    vbox->addWidget(le2);
    QLineEdit* le3 = new QLineEdit();
    le3->setObjectName("le3");
    vbox->addWidget(le3);
    vbox->addStretch();
    //WId winId = le3->winId(); // Make Qt windows real HWND windows

    pPaletteWidget1->setLayout(vbox);
    pPaletteWidget1->show();

    CRect cr;
    m_pPalSet->GetClientRect(&cr);
    pPal->OnSize(0, cr.Width(), cr.Height()); // Force to resize palette widget, needed when system
scale !=100%
  }
  else {
    m_pPalSet->Show(!m_pPalSet->IsWindowVisible());
  }
}

extern "C" __declspec(dllexport) AcRx::AppRetCode
acrxEntryPoint(AcRx::AppMsgCode msg, void* appId)
{
  switch (msg)
  {
  case AcRx::kInitAppMsg:
    acrxDynamicLinker->unlockApplication(appId);
    acrxDynamicLinker->registerAppMDIAware(appId);

    acedRegCmds->addCommand(L"HELLOQT_GROUP",
                            L"HELLOQTMODALDLG",
                            L"HELLOQTMODALDLG",
                            ACRX_CMD_TRANSPARENT,
                            helloQtModalDlgCmd);

    acedRegCmds->addCommand(L"HELLOQT_GROUP",
                            L"HELLOQTPALETTE",
                            L"HELLOQTPALETTE",
                            ACRX_CMD_TRANSPARENT,
                            helloQtPaletteCmd);
    break;

  case AcRx::kUnloadAppMsg:
    acedRegCmds->removeGroup(L"HELLOQT_GROUP");

    if (m_pPalSet){
      m_pPalSet->DestroyWindow();
      m_pPalSet = 0;
    }

    break;
  }

  return AcRx::kRetOK;
}
```