Системы автоматизированного проектирования

Лекция 1. Подготовка среды разработки программных модулей NX на базе библиотек Open API

Структура дисциплины

№ раздела	Наименование разделов	Количество часов				
		всего	аудиторная работа			внеауд. работа
			Л	ПЗ	ЛР	
1	Основы моделирования в NX	38	4	4	0	30
2	Создание чертежей и параметризация	26	2	2	0	22
3	Администрирование NX	26	2	2	0	22
4	Подготовка среды разработки программных модулей NX на базе библиотек Open API	26	4	0	2	20
5	Моделирование кривых	26	2	0	2	22
6	Моделирование объектов и действий над ними функциями	38	4	0	4	30
	Итого:	180	18	8	8	146
	Всего:	180	18	8	8	132

Литература

Основная литература

1 Бутко А.О. Основы моделирования в САПР NX [Электронный ресурс] / Бутко А.О., Прудников В.А., Цырков Г.А. - НИЦ ИНФРА-М, 2015. – Режим доступа: http://znanium.com/bookread2.php?book=503629.

2 Хорев П. Б. Объектно-ориентированное программирование с примерами на C#: Учебное пособие [Электронный ресурс] / Хорев П. Б. - НИЦ ИНФРА-М, 2016. – Режим доступа: http://znanium.com/bookread2.php?book=529350

Дополнительная литература

1 Каменев, С. В. Основы моделирования машиностроительных изделий в автоматизированной системе "Siemens NX 10" : учебное пособие [Электронный ресурс] / С. В. Каменев. - Оренбург : ОГУ, 2015. – 165 с. – Режим доступа: http://artlib.osu.ru/web/books/metod_all/9145_20151106.pdf.

2 Каменев, С. В. Инженерный анализ механизмов в системе моделирования движения "Siemens NX" [Электронный ресурс] : учебное пособие / С. В. Каменев; М-во образования и науки Рос. Федерации, Федер. гос. бюджет. образоват. учреждение высш. образования "Оренбург. гос. ун-т". - Электрон. дан. - Оренбург : ОГУ, 2018. - 1 электрон. опт. диск (CD-ROM). - Загл. с тит. экрана. -Систем. требования: IBM PC 686 (Pentium или выше) ; Microsoft Windows NT5.x (2000, XP, 7, 8, 10) ; процессор 233 МГц ; 512 Мб ; монитор, поддерживающий режим 1024x768 ; мышь или аналогич. устройство - ISBN 978-5-7410-1965-8.. - № гос. регистрации 0321900037.

Рекомендуемая литература

1 Тихомиров В.А. Разработка приложений для NX на языке C. - Издательство: ФГБОУ ВПО «КнАГТУ», 2011. - 469 с.

Вопросы

1. Установка и настройка среды компиляции проектов NX
2. Подготовка среды разработки
3. Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX
4. Отладка внутренних прикладных модулей NX

1 Установка и настройка среды компиляции проектов NX

Для доступа к трехмерной модели в среде NX и построения различных автоматизированных систем, управляющих, как самими моделями, так и операциями их изменения, предусмотрен механизм разработки прикладных программных модулей на языке Си с использованием библиотек встроенных функций **NX Open API**.

Прикладной модуль может быть внешнего типа и выполнен, как отдельное приложение в формате EXE файла, или внутреннего типа - DLL файл, который запускается из запущенной CAD-системы NX. Инструментарием для создания таких модулей рекомендуется использовать MS Visual Studio.

1 Установка и настройка среды компиляции проектов NX

Три пути установки и настройки среды Microsoft Visual Studio для компиляции проектов внутренних и внешних модулей для NX:

1. настройка, когда CAD система NX устанавливается после того, как установлена система MS Visual Studio;

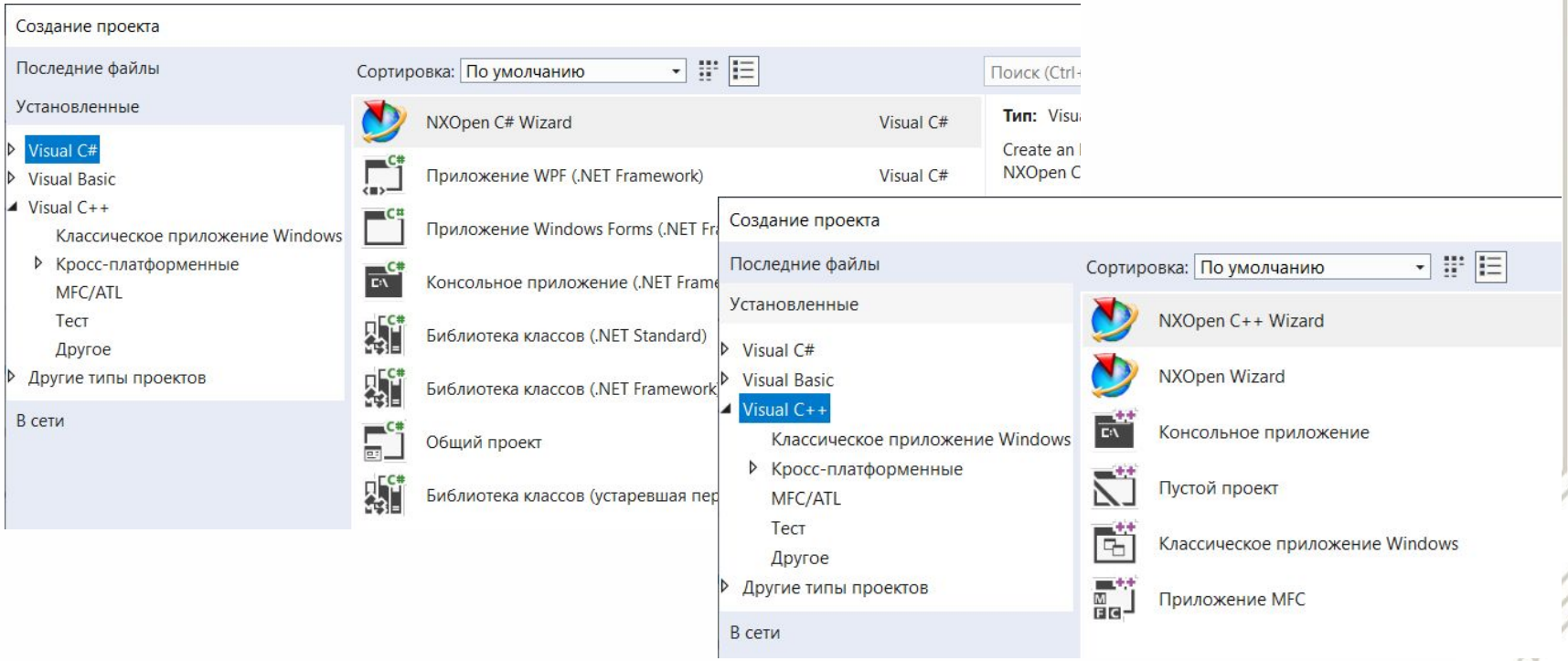
2. настройка, когда MS Visual Studio ставится после установки CAD системы NX;

3. ручная настройка проекта, в случае, если ни первый, ни второй путь не позволил получить требуемого результата.

Штатным (соответствующим рекомендациям разработчиков NX) следует считать первый вариант. При нем вся установка и настройка выполняется в автоматическом режиме.

1 Установка и настройка среды компиляции проектов NX

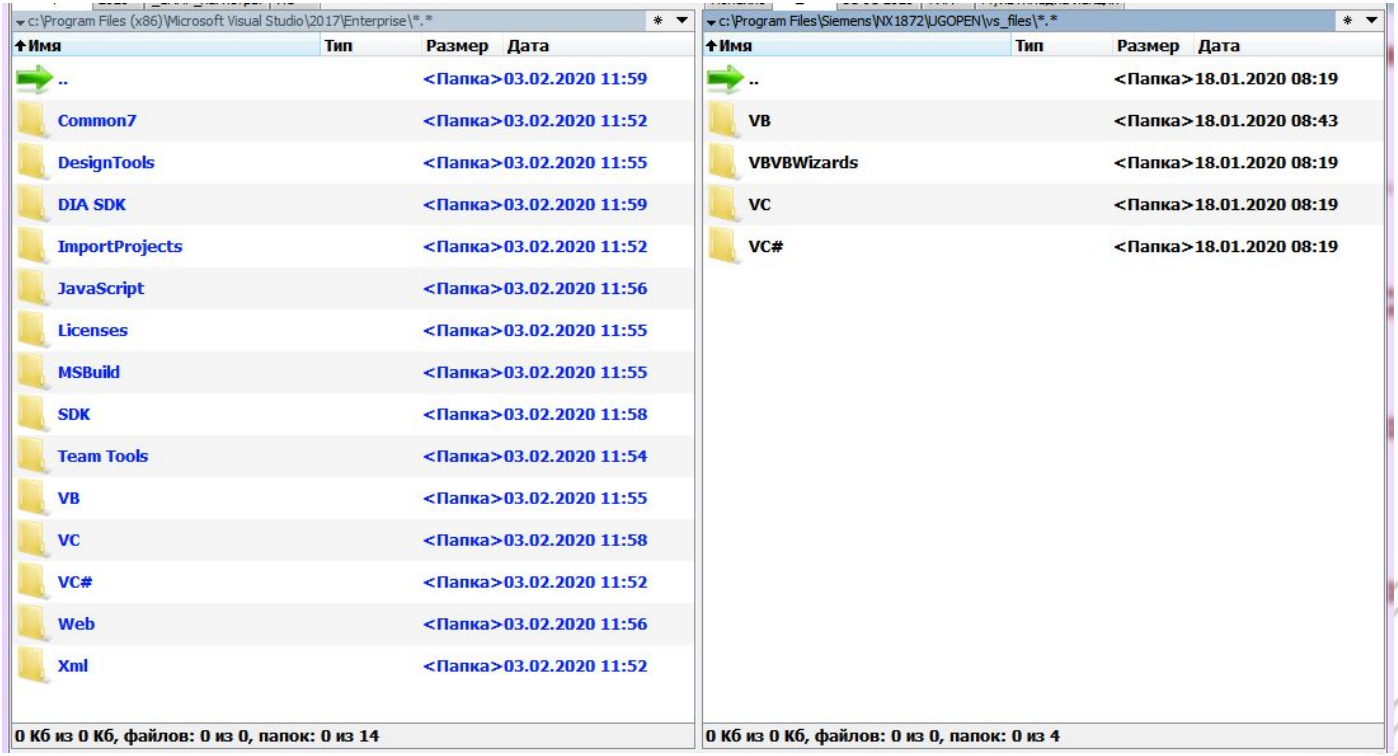
При **первом способе** во время установки происходит автоматическая интеграция пакета NX в среду разработки MS VS и если после завершения установки NX открыть Visual Studio (File ->New-> Project), можно увидеть в окне шаблонов проектов вновь добавленные специальные шаблоны для создания приложений NX. Новые шаблоны доступны вместе со стандартными - на вкладках для различных языков программирования: Visual C++ , Visual C#, Visual Basic.



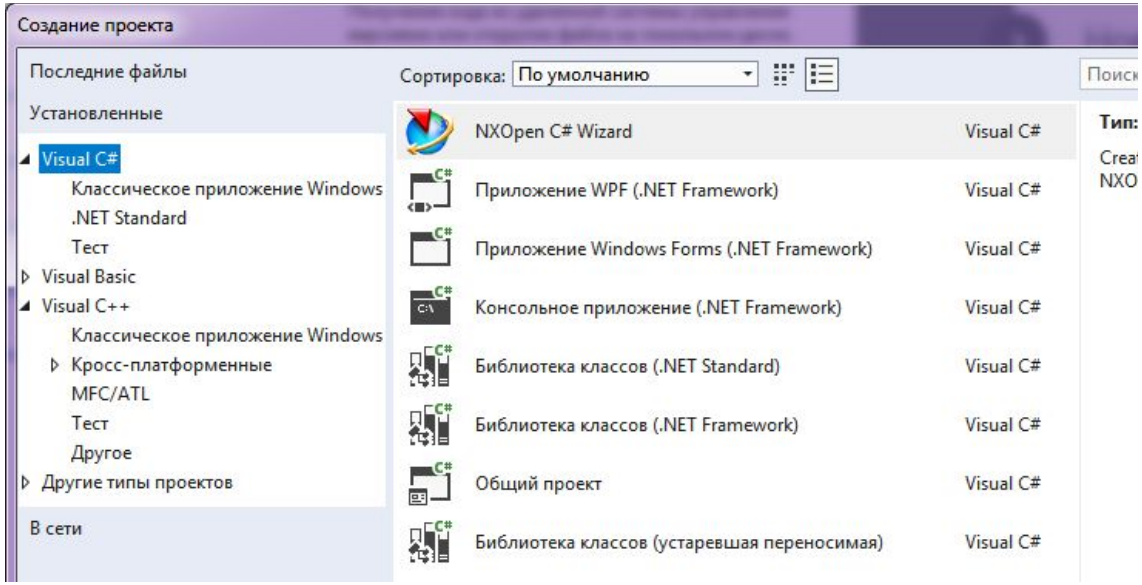
2 Подготовка среды разработки

Если по каким-то причинам правильный вариант (вариант-1) установки пакетов провести не удалось и пакеты установлены в обратном порядке, то автоматической интеграции NX в Visual Studio не происходит и шаблоны проектов NX в соответствующих окнах не появляются.

Чтобы **вручную установить шаблоны** проектов NX в Visual Studio необходимо скопировать содержимое папки X:\Program Files\Siemens\NX1872\UGOPEN\vs_files\в папку X:\Program Files (x86)\Microsoft Visual Studio\2017\Enterprise\, где X: - буква диска с установленным пакетом NX (с путями установки возможны варианты...).



2 Подготовка среды разработки

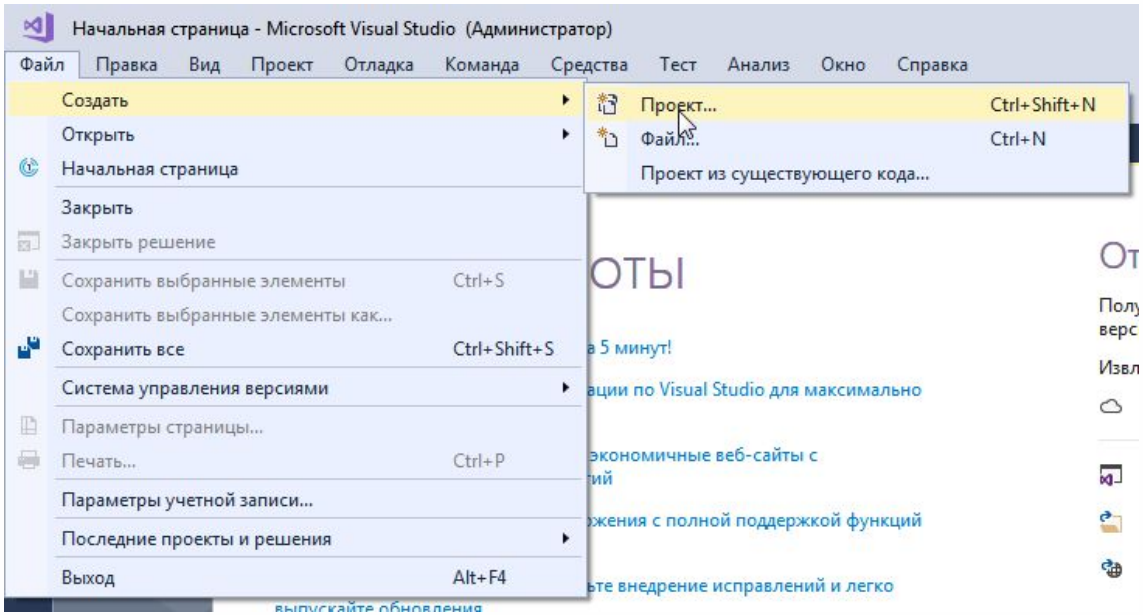


После копирования указанных папок в окне шаблонов проектов Visual Studio появятся шаблоны для создания приложений NX.

Ручное создание проекта NX

Возможна ситуация, когда необходимо создать проект NX в среде, не воспринимающей готовые шаблоны проектов, находящиеся в каталоге ...\UGOPEN\vs_files. Например, Visual Studio 2008 шаблоны версий NX 4.xx, 5.xx, 6.xx «не понимает».
В таком случае требуется ручная настройка проекта разрабатываемого модуля NX.

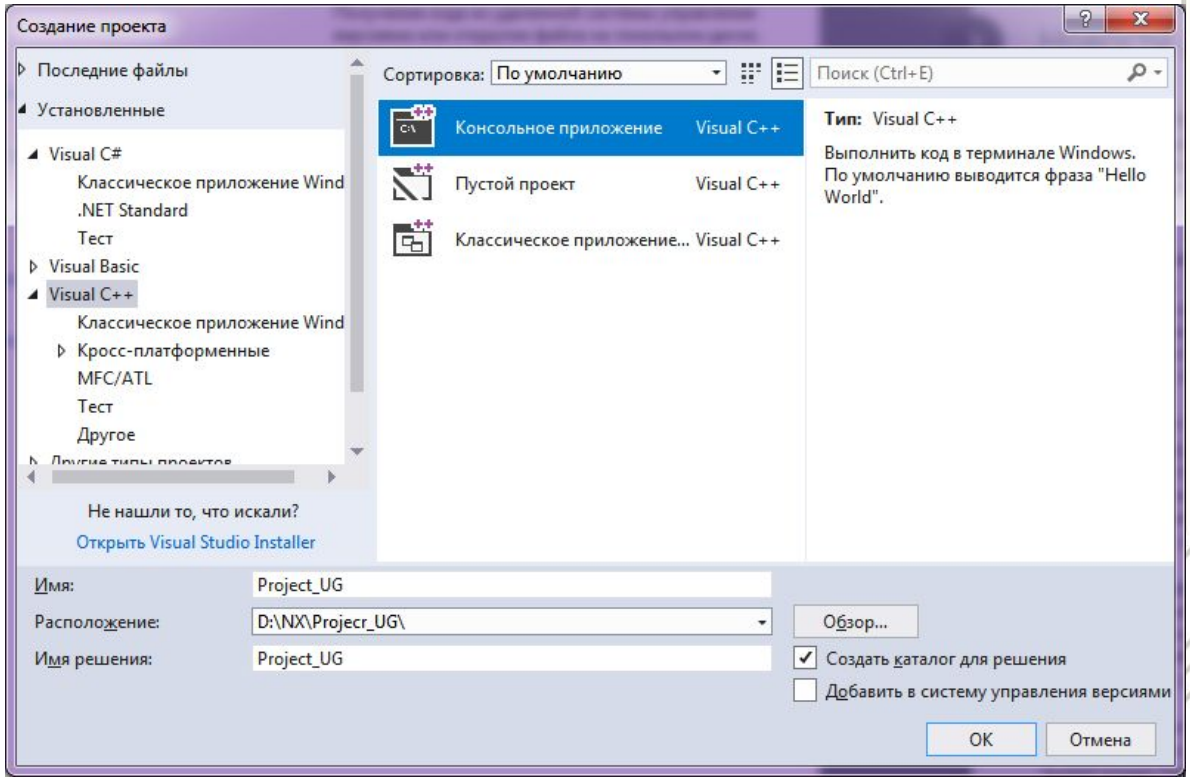
1 Откроем Visual Studio и выберем «Создать ->Проект».



2 Подготовка среды разработки

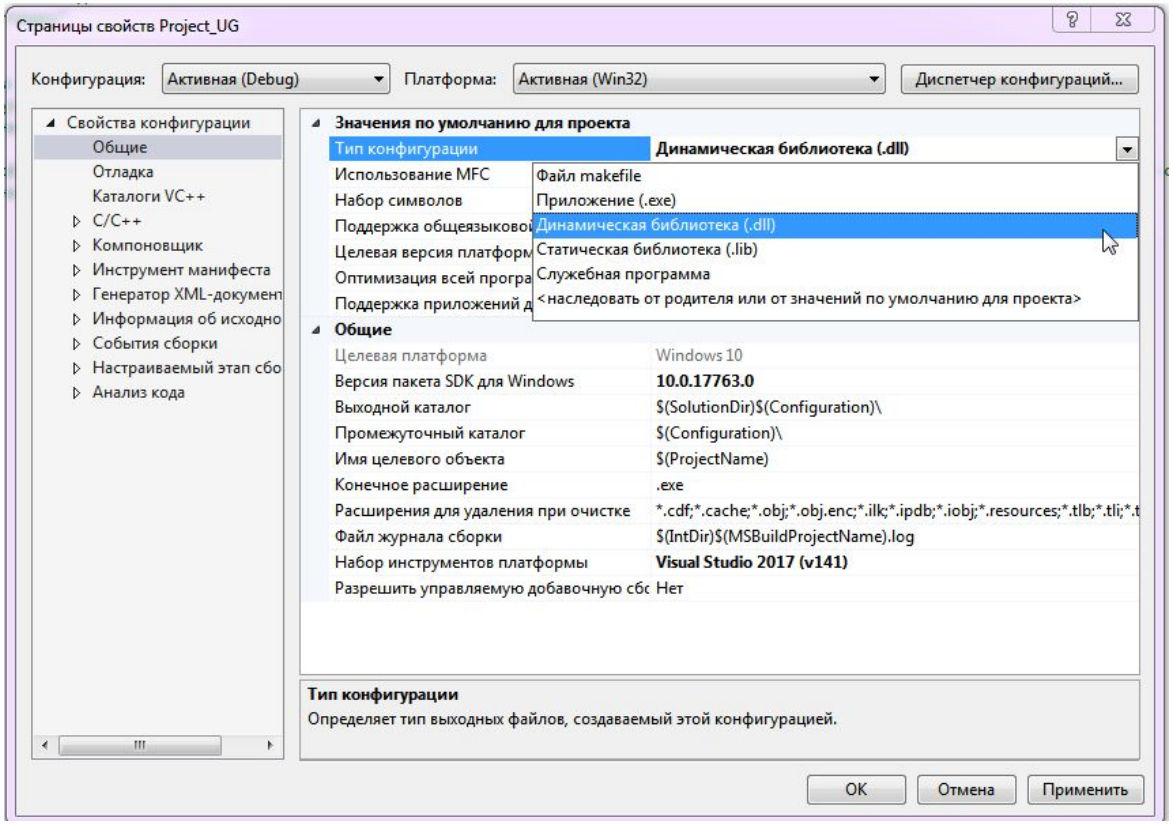
2 Следует выбрать язык программирования (в нашем случае C++), для чего в левом окне «Создание проекта» выберем Visual C++. Затем следует выбрать тип проекта - в нашем случае Консольное приложение (Win32 Console Application), и дать название проекту (в примере - Project_UG).

Замечание. Ни в путях, ни в названиях файлов не должны использоваться русские буквы!!!



2 Подготовка среды разработки

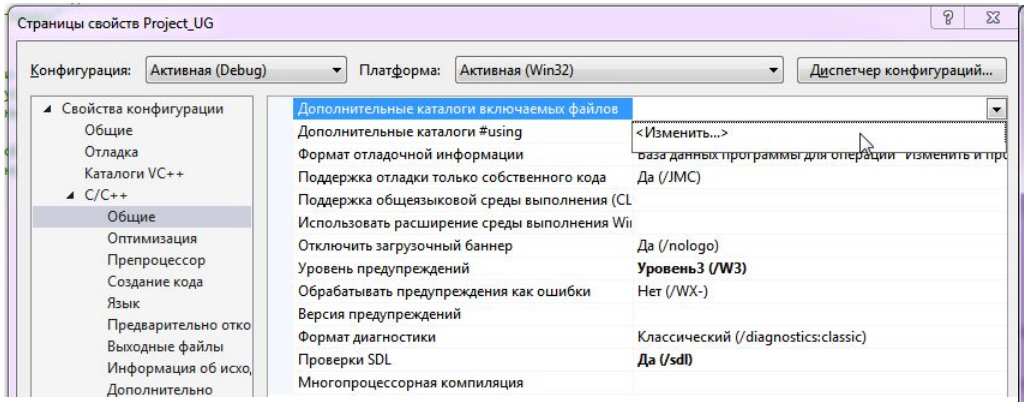
3 В опциях создания проекта «**Страницы свойств ProjectUG**» в разделе «**Свойства конфигурации**□**Общие**» в пункте «**Тип конфигурации**» следует выбрать тип «**Динамическая библиотека (.dll)**» (если планируется создание внутреннего модуля NX) или «Приложение (.exe)» (если планируется создание внешнего модуля NX).

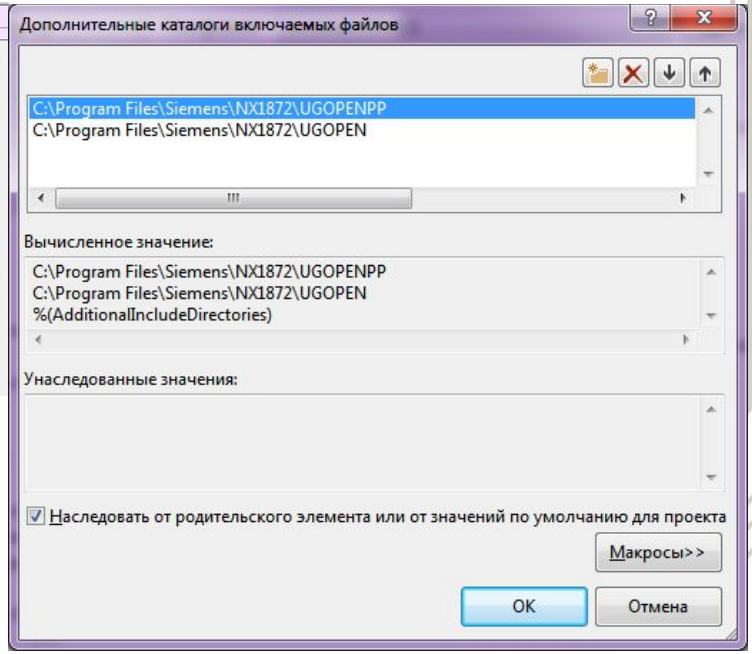


2 Подготовка среды разработки

4 Теперь следует настроить свойства проекта. Чтобы проект правильно скомпилировался в исполняемый файл NX, необходимо прописать место нахождения подключаемых библиотек и некоторые зависимости компоновщика.

Из вкладки «**Свойства конфигурации □ C/C++ □ Общие**» перейти к строке «**Дополнительные каталоги включаемых файлов**» (Additional Include Directories) добавить местонахождение папок **UGOPEN** и **UGOPENPP**, которые находятся в каталоге NX (в примере обе папки находятся в каталоге **c:\Program Files\Siemens\NX1872**).

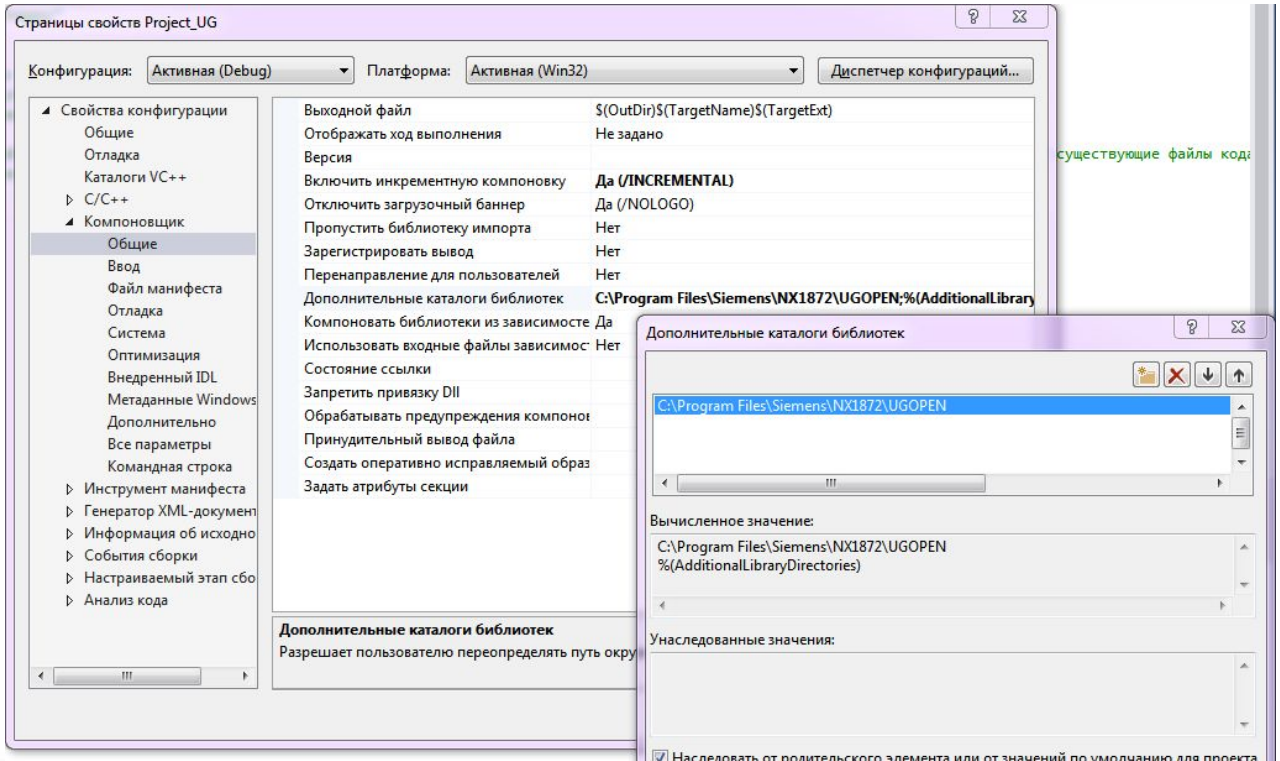




2 Подготовка среды разработки

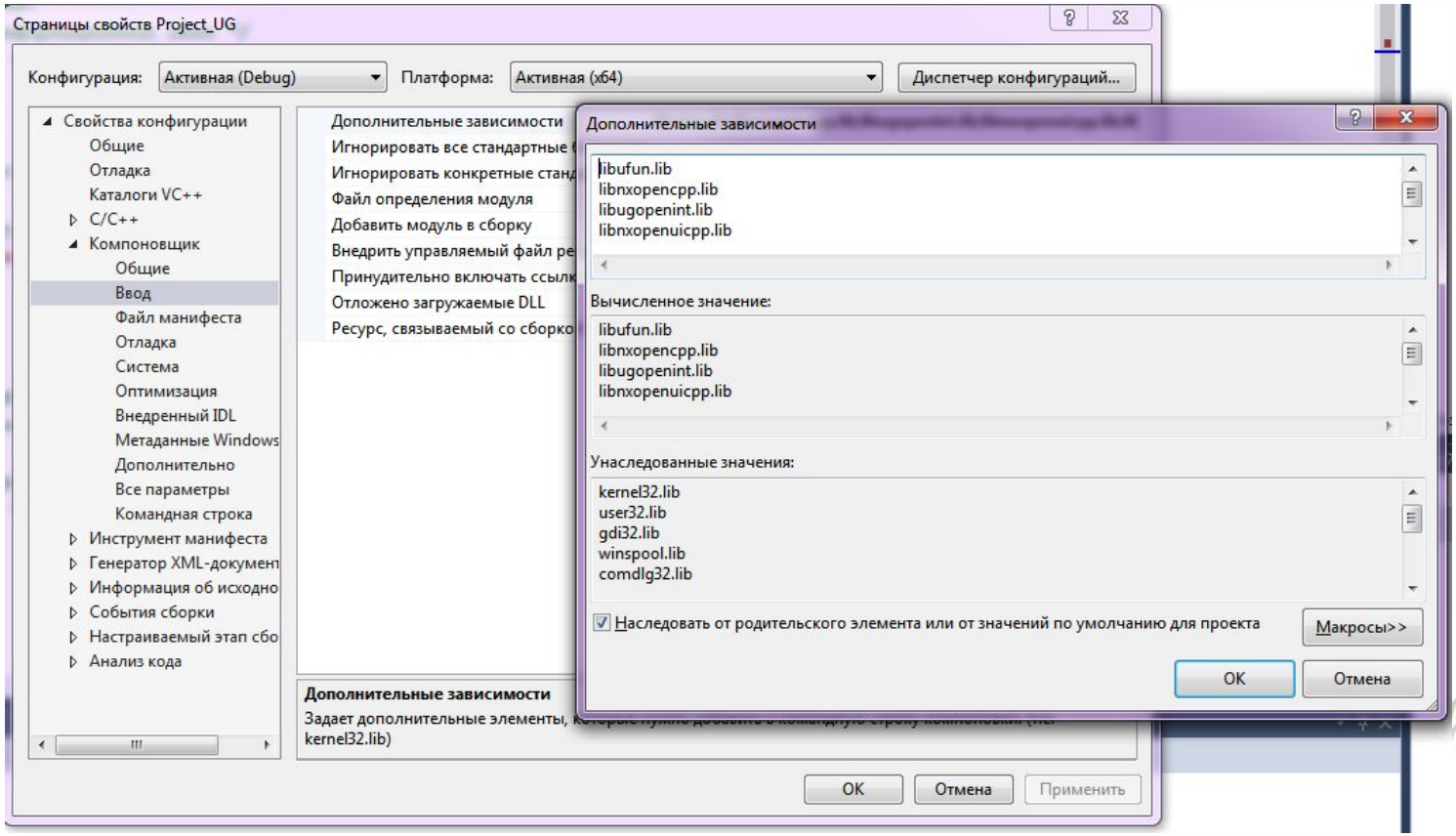
Данная опция подключает **UGOPEN** и **UGOPENPP** к проекту NX (для программирования только на языке С достаточно пути к одной папке **UGOPEN**)…

5 Из вкладки «**Свойства конфигурации**□**Компоновщик**□**Общие**» выбрать параметр «**Дополнительные каталоги библиотек**» (Additional Library Directories) и добавить путь к папке **UGOPEN**. Это необходимо для подключения к проекту основной папки с библиотеками **UGOPEN**.



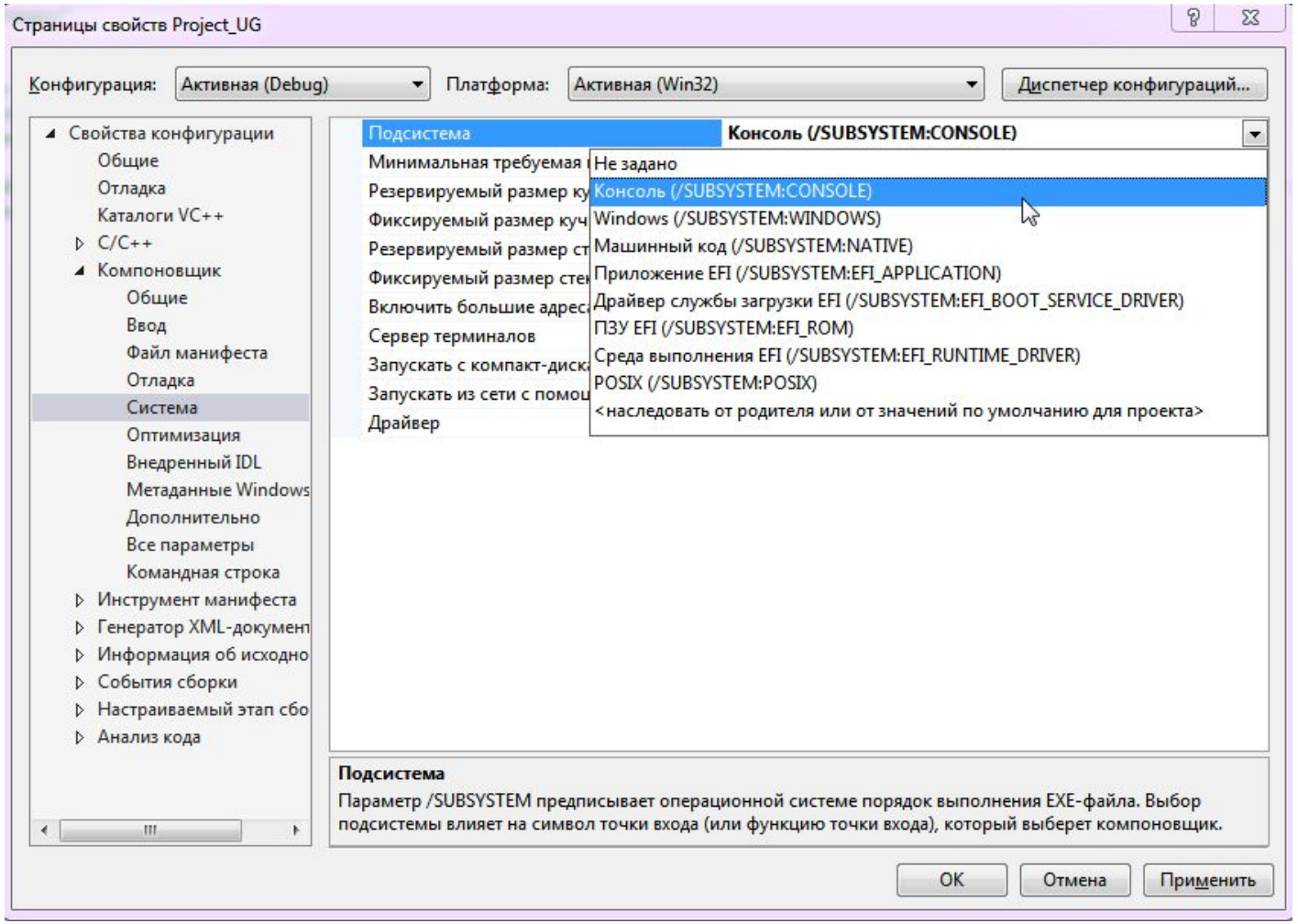
2 Подготовка среды разработки

6 Из вкладки «**Свойства конфигурации□Компоновщик□Ввод**» выбрать параметр «**Дополнительные зависимости**» (Additional Dependencies) и прописать, следующее библиотеки:
libufun.lib; libnxopencpp.lib; libugopenint.lib; libnxopenuicpp.lib; libopenpp.lib; libopenintpp.lib; libvmathpp.lib.
каждую в отдельной троке



2 Подготовка среды разработки

7 Из вкладки «**Свойства конфигурации□Компоновщик□Система**» для параметра «**Подсистема**» (Subsystem) **выбрать Консоль (/SUBSYSTEM:CONSOLE)**.



3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Внутренние (Internal) прикладные модули системы NX, разрабатываемые на языке C, создаются в формате DLL библиотек и запускаются из работающей NX системы. Таким образом, внутренние прикладные модули выполняются в адресном пространстве процесса системы NX, что облегчает доступ из модуля к объектам NX, ускоряет операции их создания и редактирования. Стандартной точкой входа в прикладную DLL NX является функция **ufusr ()**. Рабочий шаблон этой функции представлен в листинге 1.1

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Листинг 1.1

```
/* Минимальный шаблон внутреннего модуля NX на базе Open C
входные параметры:
param - если эта функция будет вызвана из menuscript, то сюда передастся
название меню, в других случаях этот параметр не используется;
parm_len - длина строки параметров
выходные параметры:
retcod - возвращаемый код прикладного модуля    */


#include   <uf.h> // Используемые файлы описаний
void  ufusr(char   *param, int   *retcod, int  parm_len)
{
    UF_initialize(); //Инициализация библ. функций Open API
    ... // здесь должно быть тело прикладного модуля


UF_terminate(); //Закрытие библиотек функций Open API

}
```

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

В шаблоне представлена одна, главная, точка входа в прикладной модуль - **`ufusr ().`** Всего же в NX предусмотрено 39 различных точек входа в процедуры исполняемого модуля, которые система NX вызывает в различные моменты своей работы. Информация о трех, наиболее часто используемых точках входа представлена в таблице 1

Таблица 1 - Основные точки входа во внутренних приложениях системы NX

Имя точки входа	Момент вызова	Назначение точки входа
`ufusr`	Запуск модуля	Запустить процесс
`ufusr_ask_unload`	После выгрузки модуля	Принять код завершения процесса
`ufusr_cleanup`	При завершении работы модуля	Выполнить операции по очистке и освобождению динамических переменных модуля

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Вторая точка входа имеет сигнатуру:

```
extern    int ufusr_ask_unload(void)
{
return    (<код  выхода  из  модуля>);
}
```

и обычно содержит только код возврата, определяющий правила завершения работы прикладного модуля. Возможные коды возврата представлены в таблице 2.

Таблица 2 - Коды возврата из внутренних приложений NX

Код возврата	Описание
UF_UNLOAD_IMMEDIATELY	Обеспечивает выгрузку DLL библиотеки модуля из памяти сразу после завершения работы модуля.
UF_UNLOAD_SEL_DIALOG	Обеспечивает возврат управления окну диалога, вызвавшему данный модуль.
UF_UNLOAD_UG_TERMINATE	Обеспечивает выгрузку DLL модуля вместе с завершением работы системы NX.

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Третья точка входа имеет сигнатуру:

```
extern void ufusr_cleanup(void) { return; }
```

В теле этой функции разработчик должен выполнить очистку и уничтожение всех динамических объектов, созданных им в ходе работы приложения (если таковые остались).
В результате - полный типовой шаблон стандартного внутреннего модуля системы NX должен состоять из трех вышеперечисленных функций.
Проведем проверочную компоновку и компиляцию такого шаблона,

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

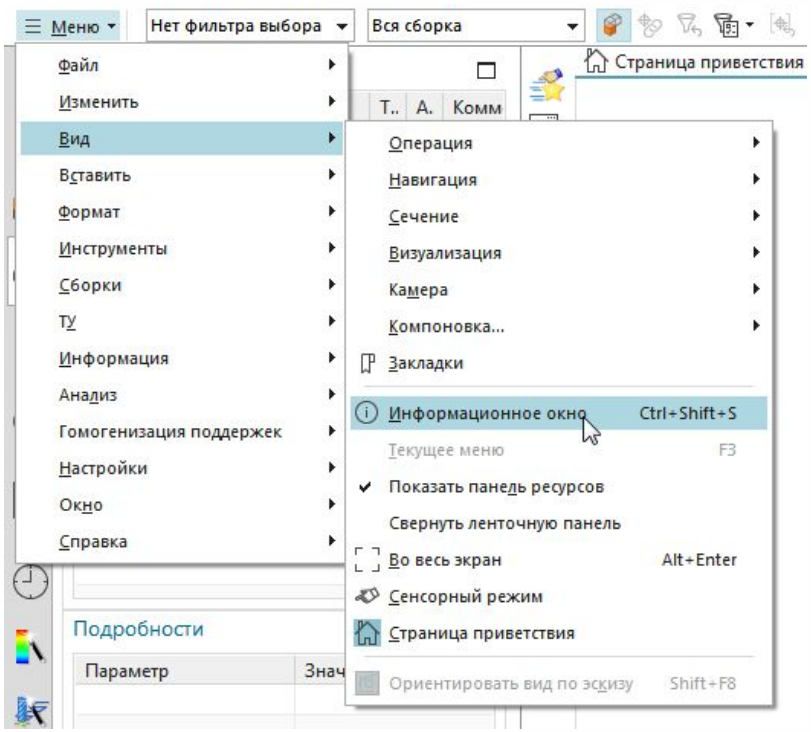
Листинг 1.2

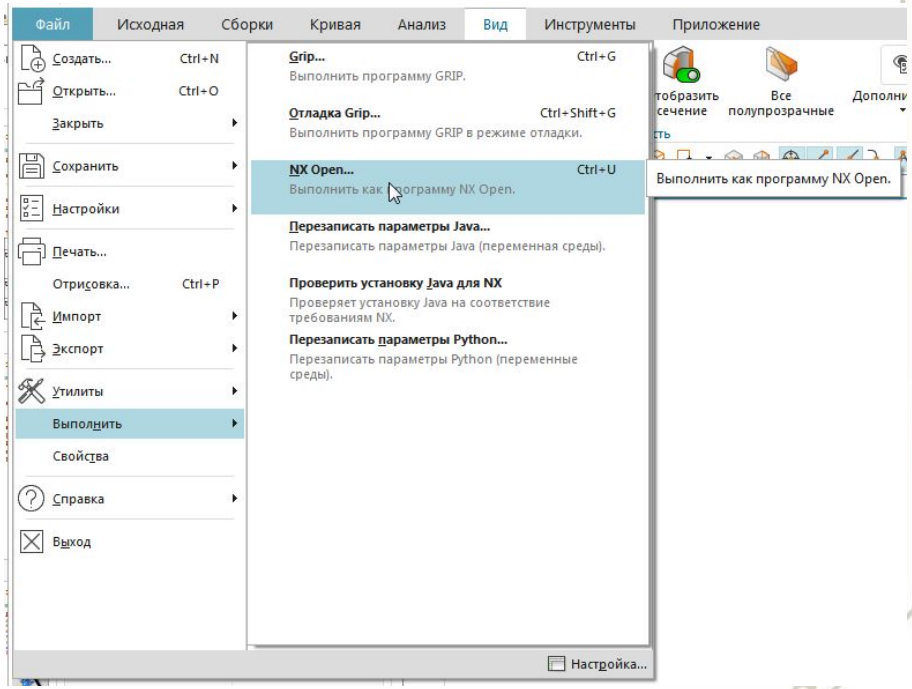
```
#include   <stdio.h>
#include   <uf.h>
#include   <uf_ui.h>
char  msg[120]; //строка для формирования текстового сообщения
// Процедура главной точки входа в прикладной модуль NX
extern   DllExport  void  ufusr(char   *param,  int *retcode, int   parm_len)
{
      if (UF_initialize())      return;
      /* Место для собственого кода разработчика:
      готовим текстовую строку и выводим ее в информационное окно */
      sprintf_s(msg,    "Вход  в  тело  DLL  выполнен \n");
      UF_UI_write_listing_window(msg);
      UF_terminate();
}
      //    Функция, вызываемая системой после выгрузки модуля
      extern   int   ufusr_ask_unload (   void   )
      {   //вывод информации о выходе из процедуры
      sprintf_s   (msg,    "Выход  из  DLL  выполнен \n\n");
      UF_UI_write_listing_window   (msg);
      // выгрузить модуль немедленно
      return    (UF_UNLOAD_IMMEDIATELY);
      }
      //    Функция, вызываемая системой при завершении приложения для
    //      выполнения операций очистки динамических переменных модуля
      extern     void  ufusr_cleanup(void)
      {     //вывод информации о входе в процедуру
            sprintf_s(msg, "Выполнение   функции  ОЧИСТКА \n\n");
            UF_UI_write_listing_window(msg);
            return;
      }
```

3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Перед запуском модуля сначала нужно запустить NX, создайть в нем новый документ, через меню **Меню□Вид** открыть информационное окно (Information Window) (или с помощью клавиш **Ctrl+Shift+S**).

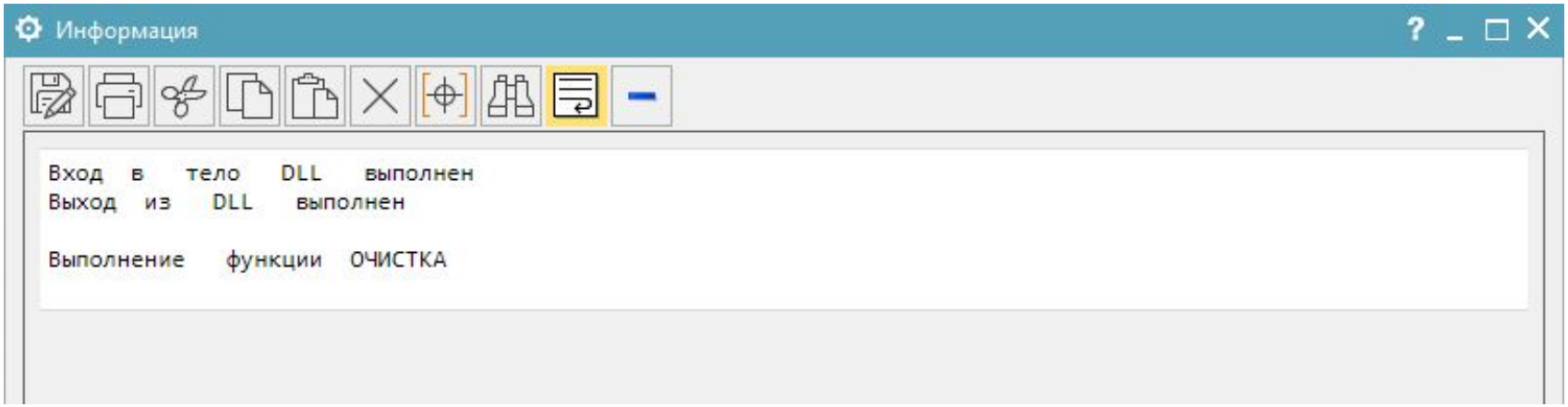
Открыть окно запуска модуля можно через меню **Файл □Выполнить□Nx Open** или сочетанием клавиш **Ctrl+U**. В окне выбирать созданную ранее DLL (в нашем примере **Project_UG**), произойдет загрузка и выполнение модуля.





3 Выполнение модулей DLL в среде NX. Шаблон внутренних прикладных модулей NX

Информационное окно заполняется строками из разработанной программы.

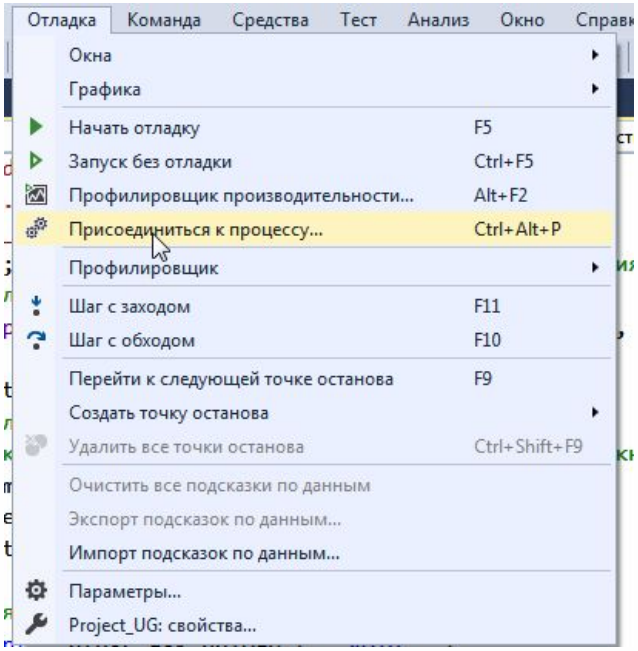


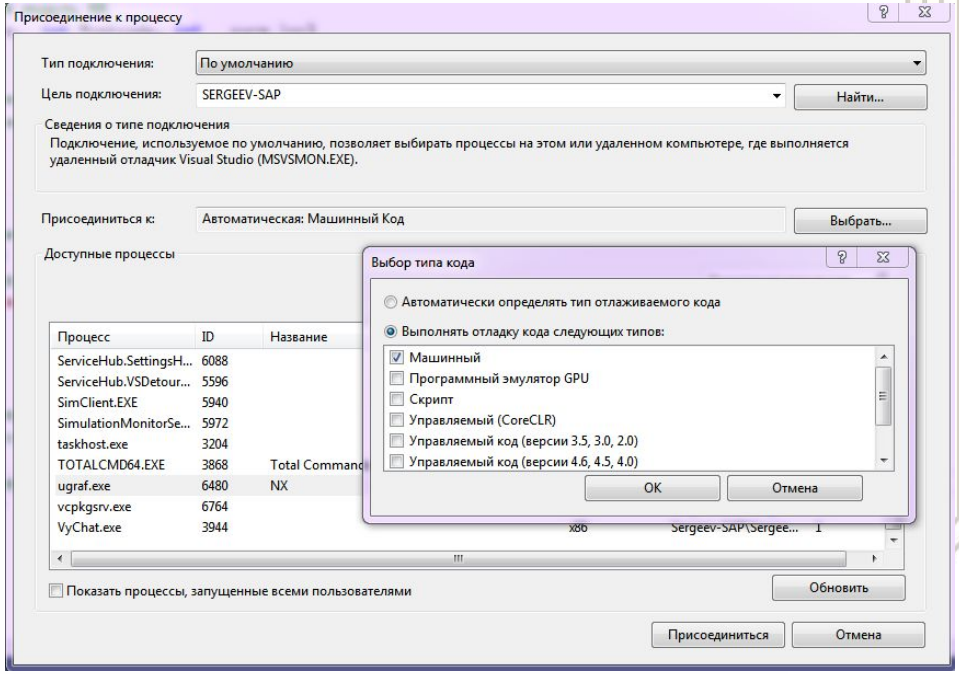
Как видно из рисунка, сначала был осуществлен вход в тело функции **ufsr (...),** что подтверждает вывод строки 1, затем произошел запуск функции **ufusr_ask_unload (...)** (строка 2) и последней (при выгрузке модуля) была запущена функция **ufusr_cleanup (…)** (строка 3).

4 Отладка внутренних прикладных модулей NX

Если возникла необходимость отладки (трассировки) программного кода внутреннего прикладного модуля NX, следует выполнить следующие простые действия. Сначала, в среде MS Visual Studio, применяемой для разработки приложения, следует присоединится к главному процессу системы NX. Для этого в меню «**Отладка**» надо выбрать режим «**Присоединиться к процессу...**» (**Ctrl+Alt+P**) для получения соответствующего окна со списком всех доступных процессов, имеющихся в операционной системе.

В этом окне следует нажать кнопку «**Выбрать**», чтобы открыть окно «**Выбор типа кода**» для выбора типа кода, подлежащего отладке. Надо включить радиокнопку «**Выполнять отладку кода следующих типов**» и выбрать тип кода «**Машинный**».





4 Отладка внутренних прикладных модулей NX

После чего окно «**Выбор типа кода**» закрыть, в списке «**Доступные процессы**» в колонке «**Процесс**» выбрать процесс с именем **ugraf.exe** (основной исполняемый модуль NX) и нажать кнопку «**Присоединиться**».

Среда разработки соединится с основным процессом системы NX. В исходном тексте модуля следует поставить точку останова (**F9**) на интересующее вас место программы, перейти в окно системы NX и запустить отлаживаемую DLL.
Как только исполнение кода дойдет до точки останова, будет вызвана среда разработки и программист сможет вести отладку разрабатываемого приложения в стандартном режиме.

Project_UG.cpp

Project_UG (Глобальная область)

```
1  #include   <stdio.h>
2  #include   <uf.h>
3  #include   <uf_ui.h>
4  char  msg[120]; //строка для формирования текстового сообщения
5  // Процедура главной точки входа в прикладной модуль NX
6  extern   DllExport  void  ufusr(char   *param,  int *retcode, int
7  {
8      if (UF_initialize())      return;
9      /* Место для собствееного кода разработчика:
10     готовим текстовую строку и выводим ее в информационное окно */
11     sprintf_s(msg,    "Вход  в   тело   DLL   выполнен \n");
12     UF_UI_write_listing_window(msg);
13     UF_terminate();
14     }
15     //  Функция, вызываемая системой после выгрузки модуля
16     extern   int   ufusr_ask_unload (   void   )
17     {   //вывод информации о входе в процедуру
18     sprintf_s   (msg,    "Выход  из   DLL   выполнен \n\n");
19     UF_UI_write_listing_window   (msg);
20     // выгрузить модуль немедленно
21     return    (UF_UNLOAD_IMMEDIATELY);
22     }
23     //  Функция, вызываемая системой при завершении приложения для
24     //  выполнения операций очистки динамических переменных модуля
25     extern     void  ufusr_cleanup(void)
26     {     //вывод информации о входе в процедуру
27         sprintf_s(msg, "Выполнение   функции  ОЧИСТКА \n\n");
28         UF_UI_write_listing_window(msg);
29         return;
30     }
```

Вопросы для самоконтроля

1. Способы разработки собственных инструментов и приложений в NX.
2. Последовательность среды разработки приложений Visual Studio и Siemens NX.
3. Настройка приложений при различных вариантах установки.
4. Ручная настройка проекта разрабатываемого модуля NX.
5. В каких каталогах находятся библиотеки функций для C и C++?
6. Каким образом в NX запускаются библиотеки разработанные *.dll?
7. Варианты точек входа во внутренних приложениях системы NX.
8. Создание типового шаблона стандартного внутреннего модуля системы NX.
9. Как обеспечить выгрузку разработанного модуля из памяти сразу по завершению его работы.
10. Подключение отладчика к приложению NX.
11. Точки останова.


Спасибо за внимание!