**ПЛАН ЗАНЯТИЯ**

**Дисциплина:** МДК.01.04 Системное программирование

**Преподаватель:** Галузин А.Б.

**Курс: 4**

**Группа:** П-40

**Специальность:** 09.02.07 Информационные системы и программирование

**Дата:** 25.09.25

**Время проведения:** 1 пара

**Тема:** Особенности разработки программ в Masm и Visual Studio.

**Цель занятия:**

**дидактическая:** изучить особенности разработки программ в Masm и Visual Studio

**развивающая**: развивать абстрактное мышление, логику

**Вид занятия** лекция

**Литература**

Юров В.И. Assembler. Учебник для вузов. - 2-ое изд. – СПб.: Питер, 2003, стр. 128.

**Интернет-ресурсы:**

https://metanit.com/assembler/tutorial/1.4.php

Создание проекта в Visual Studio с ассемблерным модулем ttps://rutube.ru/video/baf45c7b660bf14335f65a7675b84aae/?r=plwd

**ЗАДАНИЕ**: законспектировать лекцию с учетом контрольных вопросов.

**КОНСПЕКТ ЛЕКЦИИ**

План

1. Некоторые особенности компилятора Masm
2. Установка MASM
3. Структура программы на ассемблере MASM
4. Компиляция программы
5. Создание проекта на языке ассемблера в Microsoft Visual Studio 2022
6. Изменить тип приложения с консольного на оконное

**1. Некоторые особенности компилятора Masm**

Компиляторы языков высокого уровня (Си, Паскаль) в определенной степени совместимы между собой и, хотя, исходный текст, предназначенный для одного компилятора, не всегда без переделок транслируется на другом, синтаксис и прочие языковые концепции остаются неизменными.

С трансляторами ассемблера все обстоит иначе. Казалось бы, x86 ассемблер - он же ассемблер, так ведь нет! Помимо поддержки мнемоник машинных команд, каждый транслятор обладает своим собственным набором директив и макросредств, зачастую ни с чем не совместимых. Ассемблерный листинг, "заточенный", например, под MASM, бесполезно переносить на FASM, поскольку, возможности, предоставляемые макросредствами, у них сильно неодинаковые!

Выбирая транслятор ассемблера, вы делаете выбор, изменить который впоследствии очень сложною. Придется переучиваться, фактически осваивая новый язык. Но это еще полбеды! Всякий уважающий себя программист со временем обрастает ворохом маленьких библиотек, делающих кучу грязной работы. Их-то куда при переходе на новый ассемблер девать?! Перенести ничуть не легче, чем переписать с нуля!

К счастью, совместное использование нескольких трансляторов еще никто не запрещал. На практике, обычно, так и поступают. Ставится конкретная задача и для ее решения выбирается наиболее адекватный инструмент. Естественно, чтобы сделать правильный выбор необходимо знать, какие ассемблеры вообще есть и чем они отличаются.

На прошлых занятиях мы с вами использовали компилятор TASM, самый популярный транслятор ассемблера времен MS-DOS, созданный фирмой Borland (известной в народе как Багдад), и полностью совместимый с MASM'ом вплоть до версий 6.x и поддерживающий свой собственный режим IDEAL с большим количеством улучшений и расширений.

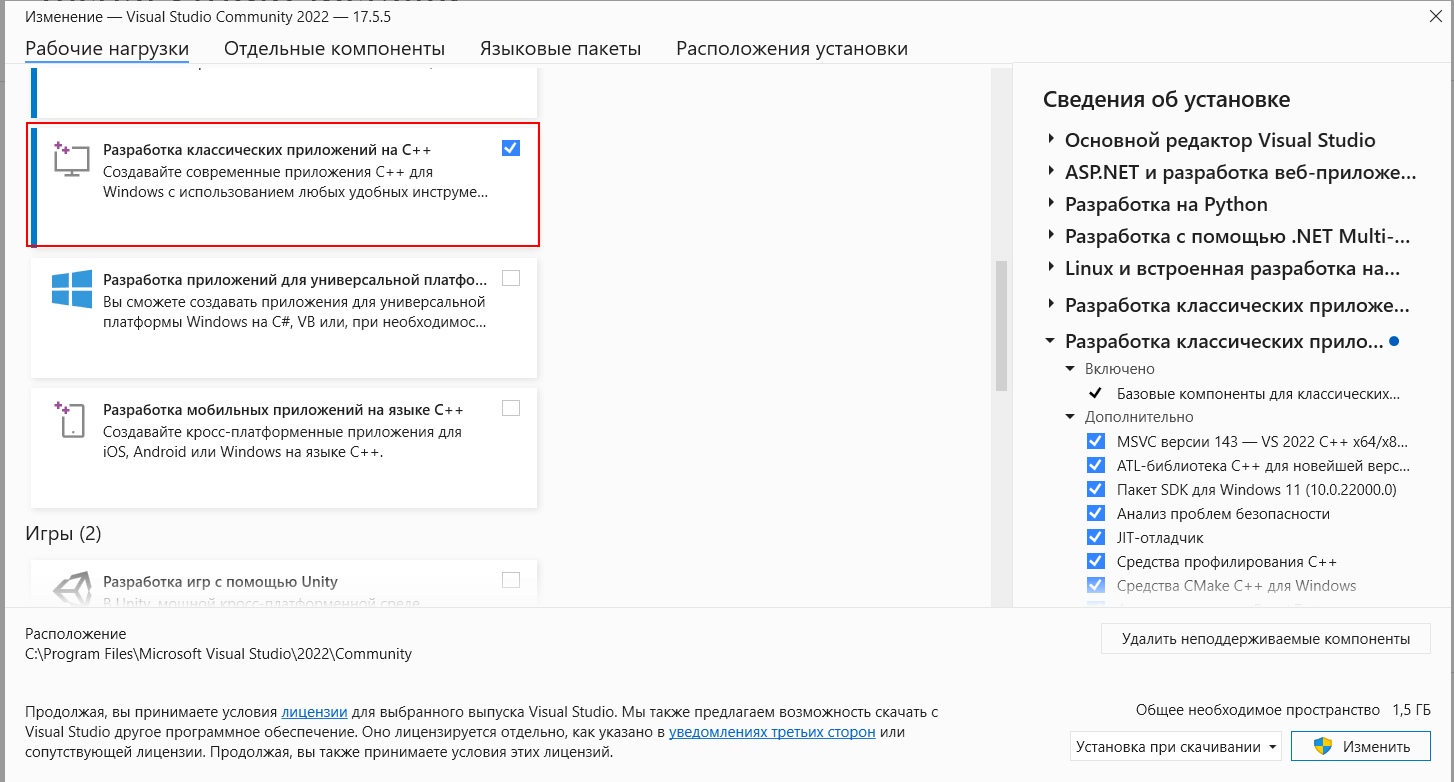
Удобство программирования, скромные системные требования и высокая скорость трансляции обеспечивали TASM'у лидерство на протяжении всего существования MS-DOS (буква "T" означает Turbo, вот так - без ложной скромности). Но с появлением Windows возрос спрос на другой компилятор - MASM.

Аббревиатура MASM расшифровывается отнюдь не как Microsoft Assembler, хотя он и является продуктом жизнедеятельности ранней компании Microsoft, а как Macro Assembler, то есть Ассемблер с поддержкой Макросов, покрывающих своими возможностями широкий круг задач, по сравнению с которым препроцессор языка Си выглядит жалкой подделкой. Кроме того, MASM активно используется для программирования в ОС Windows.

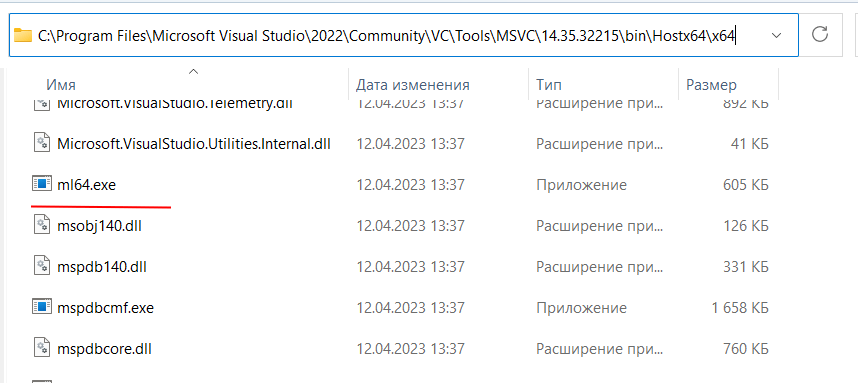
Сегодня мы с вами попробуем понять, как создать и запустить проект. на языке ассемблера в Microsoft Visual Studio

**2. Установка MASM**

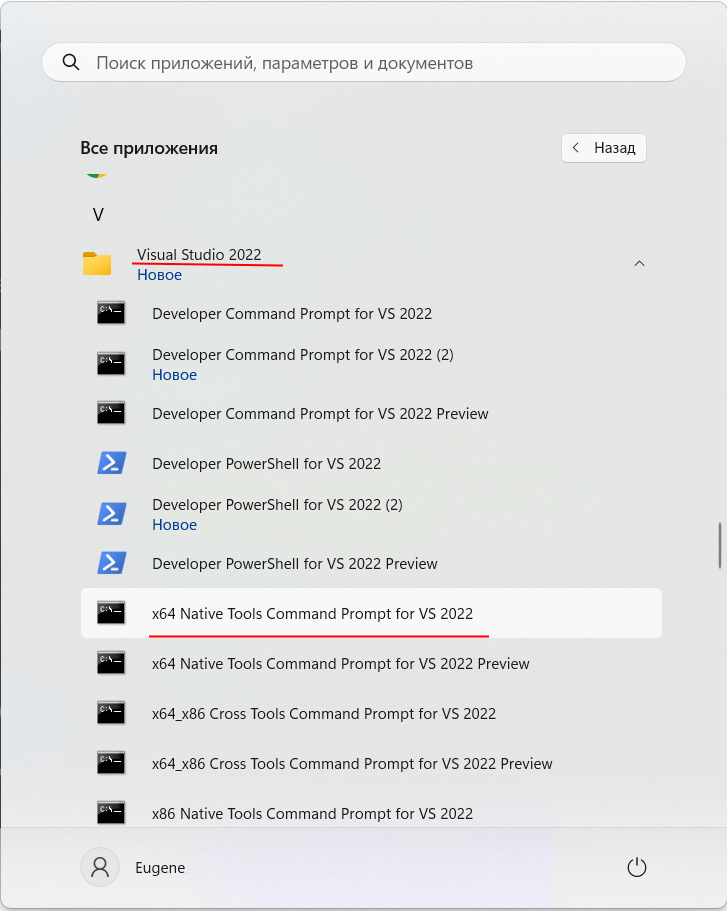
Для работы с MASM надо установить для Visual Studio инструменты разработки для C/C++. Установщик для среды Visual Studio можно загрузить по следующему адресу: [Microsoft Visual Studio 2022](https://visualstudio.microsoft.com/ru/downloads/). После загрузки программы установщика Visual Studio запустим ее и в окне устанавливаемых опций выберем пункт Разработка классических приложений на C++:



Visual Studio включает как 32-разрядные, так и 64-разрядные версии MASM. 32-раздяная версия представляет файл ml.exe, а 64-разрядная - файл ml64.exe. Точное расположение файлов может варьироваться от версии Visual Studio. Например, в моем случае это папка C:\Program Files\Microsoft Visual Studio\2022\Community\VC\Tools\MSVC\14.35.32215\bin\Hostx64  
\x64



Для использования MASM64 перейдем к меню Пуск и в списке программ найдем пункт Visual Studio и подпункт x64 Native Tools Command Prompt for VS 2022



Нам должна открыться консоль. Введем в нее ml64, и нам отобразится версия ассемблера и некоторая дополнительная информация:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Visual Studio 2022 Developer Command Prompt v17.5.5

\*\* Copyright (c) 2022 Microsoft Corporation

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[vcvarsall.bat] Environment initialized for: 'x64'

C:\Program Files\Microsoft Visual Studio\2022\Community>ml64

Microsoft (R) Macro Assembler (x64) Version 14.35.32217.1

Copyright (C) Microsoft Corporation. All rights reserved.

usage: ML64 [ options ] filelist [ /link linkoptions]

Run "ML64 /help" or "ML64 /?" for more info

C:\Program Files\Microsoft Visual Studio\2022\Community>

Стоит отметить, что запуск этой этой утилиты фактически представляет запуск файла C:\Program Files\Microsoft Visual Studio\2022\Community\VC\Auxiliary\Build\vcvars64.bat - он по сути вызывает другой файл - vcvarsall.bat, который собственно и настраивает окружение для выполнения ассемблера.

**3. Структура программы на ассемблере MASM**

Типичная программа на MASM содержит одну или несколько секций, которые определяют, как содержимое программы будет располагаться в памяти. Эти секции начинаются с таких директив MASM, как .code или .data. Данные, используемые в программе, обычно определяются в секции .data. Инструкции ассемблера определяются в секции .code.

В общем случае программа на ассемблере MASM имеет следующий вид:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | .code    main proc      ret  main endp    end |

Директива .code указывает MASM сгруппировать операторы, следующие за ней, в специальный раздел памяти, зарезервированный для машинных инструкций.

Ассемблер преобразует каждую машинную инструкцию в последовательность из одного или нескольких байт. CPU интерпретирует эти значения байт как машинные инструкции во время выполнения программы.

Далее с помощью операторов main proc определяется процедура main. Операторы main endp указывают на конец функции main. Между main proc и main endp располагаются выполняемые инструкции ассемблера. Причем в самом конце функции идет инструкция ret, с помощью которой выполнение возвращается в окружение, в котором была вызвана данная процедура.

В конце файла кода идет инструкция end

Программа может содержать комментарии, которые располагаются после точки с запятой:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | .code   ; начало секции с кодом программы    main proc   ; Функция main     ret ; возвращаемся в вызывающий код    main endp ; окончание функции main    end ; конец файла кода |

Комментарии на работу программы никак не влияют и при компиляции не учитываются.

**4. Компиляция программы**

Компиляция программы на MASM обычно происходит в командной строке. Например, воспользуемся кодом выше и напишем простейшую программу на ассемблере, которая ничего не делает. Для этого определим на жестком диске папку для файлов с исходным кодом. Допустим, она будет называться C:\asm. И в этой папке создадим новый файл, который назовем hello.asm и в котором определим следующий код:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | .code   ; начало секции с кодом программы    main PROC   ; Функция main     ret ; возвращаемся в вызывающий код    main ENDP    END ; конец файла кода |

Откроем программу x64 Native Tools Command Prompt for VS 2022 и перейдем в ней к папке, где располагается файл hello.asm. Затем выполним следующую команду

ml64 hello.asm /link /entry:main

В данном случае вызываем приложение ml64.exe и передаем ему для компиляции файл hello.asm. А флаг /link указывает MASM скомпоновать скомпилированный файл в файл приложения exe, а все дальнейшие параметры (в частности, параметр /entry:main) передаются компоновщику. Параметр /entry:main передает компоновщику имя основной процедуры/функции, с которой начинается выполнение программы. Компоновщик сохраняет этот адрес этой процедуры/функции в специальном месте исполняемого файла, чтобы Windows могла определить начальный адрес основной программы после загрузки исполняемого файла в память.

В результате ассемблер скомпилирует ряд файлов

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Visual Studio 2022 Developer Command Prompt v17.5.5

\*\* Copyright (c) 2022 Microsoft Corporation

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[vcvarsall.bat] Environment initialized for: 'x64'

C:\Program Files\Microsoft Visual Studio\2022\Community>cd c:\asm

c:\asm>ml64 hello.asm /link /entry:main

Microsoft (R) Macro Assembler (x64) Version 14.35.32217.1

Copyright (C) Microsoft Corporation. All rights reserved.

Assembling: hello.asm

Microsoft (R) Incremental Linker Version 14.35.32217.1

Copyright (C) Microsoft Corporation. All rights reserved.

/OUT:hello.exe

hello.obj

/entry:main

c:\asm>

В итоге в каталоге программы будут сгенерированы объектный файл hello.obj и собственно файл программы - hello.exe.

**Контрольные вопросы**

1. Как расшифровывается аббревиатура MASM?
2. В чем сходство и отличие трансляторов TASM и MASM?
3. Опишите структуру программы на ассемблере MASM.
4. Какой директивой задаётся секция инструкций ассемблера?
5. Какой консольной командой компилируется ассемблеровский файл в MASM?