# Из C или C++ в язык ассемблера

## 1. Что это?

Эта инструкция описывает, как получать код на языке ассемблера из исходного кода, написанного на языках C или C++, а также как его собирать в рабочую программу.

## 2. Перед работой

Чтобы превратить код рабочей программы на C или C++ понадобится только совместимый компилятор: `clang` или `gcc`. Для Windows удобнее устанавливать `gcc` — он входит в комплект поставки всего одной программы.

Для установки `gcc` на Windows достаточно скачать и установить пакет MSYS2 (`https://www.msys2.org/`) и собственно сам `gcc`. На других операционных системах, вроде Linux, BSD и macOS, любой из компиляторов легко устанавливается с помощью пакетного менеджера (`apt`, `brew`, `pacman` и так далее), а `gcc` есть ещё с момента установки ОС.

## 3. Как пользоваться?

Процесс состоит из нескольких шагов: настройка среды, разработка программы, получение кода на языке ассемблера и сборка ассемблерного кода.

### 3.1. Настройка среды

Настройка среды выполняется ТОЛЬКО ПЕРВЫЙ РАЗ и нужна исключительно для полной установки MSYS2 и компилятора.

1. Установить и обновить MSYS2, следуя инструкциям на сайте. Устанавливайте версию, которая соответствует разрядности вашей операционной системы: 32-битную версию для 32-битной ОС, 64-битную для 64-битной. Обратите внимание на путь, куда устанавливается программа. По умолчанию для 32-битной версии это «`C:\msys32\`», для 64-битной — «`C:\msys64\`». Запомните этот путь! Он понадобится дальше.

2. Открыть оболочку MSYS2 MinGW, запустив программу «MSYS2 MinGW 32-bit» на 32-битной системе или «MSYS2 MinGW 64-bit» на 64-битной.

3. Установить gcc с помощью такой команды:

   ```
   pacman -S $MINGW_PACKAGE_PREFIX-gcc
   ```

4. Закрыть оболочку MSYS2 с помощью команды:

   ```
   exit
   ```

5. Добавить папку исполняемых файлов в переменную PATH. Для этого зайти в «Панель управления» → «Система» → «Дополнительные параметры системы» (в боковом меню) → «Переменные среды…» (кнопка снизу). В части окна, подписанной «Переменные среды пользователя для…» найти переменную path (регистр не важен: Path, PATH — одинаковые имена). В поле «Значение переменной» дописать путь, который вы запомнили в шаге 1, добавив в него строку «\mingw(32или64)\bin». Если в поле уже есть какое-то значение, НЕ УДАЛЯЙТЕ ЕГО!

   Например, если у вас 32-битная операционная система и MSYS2 установлен в папку «C:\msys32\», то в поле нужно дописать «C:\msys32\mingw32\bin». Если же система 64-битная и MSYS2 установлен в папку «C:\msys64\», то в поле нужно дописать «C:\msys64\mingw64\bin». Если поле не пустое, то есть уже содержит какие-то другие пути, перед строкой, которую вы вставляете, введите символ «;». Пример на рис. 1.

6. Убедиться, что все команды выполнены правильно. Для этого открыть командную строку Windows (cmd.exe) и выполнить команду:

   ```
   gcc --version
   ```

   Среда настроена и готова к работе, если вывод вышеописанной команды выглядит примерно так:

   ```
   gcc (Rev1, Built by MSYS2 project) 7.3.0
   Copyright (C) 2017 Free Software Foundation, Inc.
   This is free software; see the source for copying conditions.
   There is NO warranty; not even for MERCHANTABILITY or FITNESS
   FOR A PARTICULAR PURPOSE.
   ```

## 3.2. Разработка программы

Раздел для тех, кто всю жизнь пользовался Visual Studio и другими IDE и не собирал исходники руками.

1. Написать нужную программу на C или C++, сохранив её текст в файл с соответствующим расширением (.c или .cpp). Например, example.c.

2. Открыть командную строку в папке с файлом исходного кода. Если открыто окно Проводника в нужной папке, то удобнее всего это делается с помощью комбинации клавиш «Shift + Правая Клавиша Мыши» — «Открыть окно команд».

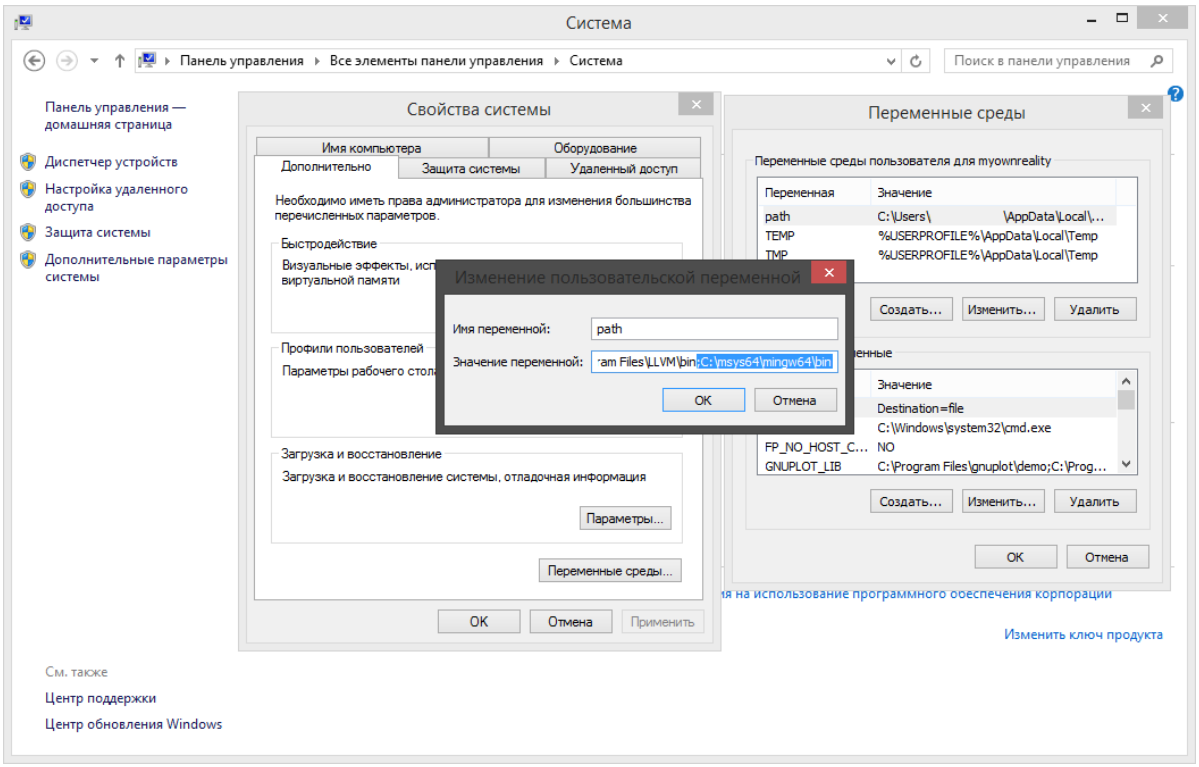


Рис. 1: Добавление папки с исполняемыми файлами в переменную PATH

3. Скомпилировать нужную программу. Например, для программы с исходным кодом в файле `example.c`[1]:

   ```
   gcc example.c
   ```

   В результате будет создан исполняемый файл `a.exe`.

4. Убедиться, что программа работает как нужно, запустив программу из командной строки:

   ```
   a.exe
   ```

Важно убедиться, что программа работает правильно, чтобы собирать ассемблерный код только один раз.

## 3.3. Получение кода на языке ассемблера

Если программа работает правильно, можно приступать к получению кода на языке ассемблера. Для этого нужно:

1. Запустить сборку программы со специальными параметрами. На примере `example.c`:

[1] Если программа написана на C++, то тут и далее нужно заменить `gcc` на `g++`, а расширение файла с `.c` на `.cpp`.

```
gcc -S -masm=intel example.c
```

В результате будет создан файл `example.s`, который и содержит код на языке Ассемблера. Название зависит от названия файла с вашим исходным кодом на C или C++, но файл с кодом на языке ассемблера всегда будет иметь расширение `.s`.

Параметр `-S` говорит компилятору производить код на языке ассемблера. Параметр `-masm` со значением `intel` говорит, что результирующий код должен быть написан на синтаксисе Intel. Именно этот синтаксис используется в тексте лаб, да и вообще используется чаще. Также можно использовать значение `att`, если нужен код с использованием синтаксиса AT&T. Этот параметр никак не зависит от производителя процессора.

### 3.4. Сборка ассемблерного кода

Превращение полученного кода на языке ассемблера в работающую программу выполняется компилятором. «Чистые» ассемблеры (`nasm`, `tasm`, `yasm`) обычно не могут этого сделать. Скорее всего, потому что компилятор оставляет специальные директивы для компоновщика (линкера) и использует некоторые собственные директивы, например, `.seh_stackalloc`. Таким образом, чтобы собрать полученный код, нужно:

1. Запустить сборку файла исходного кода на языке ассемблера:

```
gcc example.s -o example.exe
```

В результате будет создан исполняемый файл `example.exe`. Если вы выполнили все предыдущие этапы (у вас настроена среда и есть исходники рабочей программы), то вам нужен только сгенерированный код (файл `example.s`). При сдаче лабы стоит выполнять только этот этап. Остальные файлы из папки можно удалить или переместить.

## 4. Visual Studio

Для тех, у кого установлен Visual Studio, можно попытаться сделать то же самое с помощью установленных `cl.exe` и `ml.exe`. Если исходный код вашей программы записан в файле `example.c`, то получить код на языке ассемблера можно такой командой:

```
cl.exe /FA example.c
```

В результате должен быть создан файл `example.asm`. Чтобы собрать полученный файл в программу нужно выполнить такую команду:

```
ml.exe /nologo example.asm
```

Внимание! Все данные по Visual Studio вслепую взяты из MSDN. Я не могу проверить работоспособность этих команд, так что их правильную работу я не гарантирую.