Ziron Programming Language

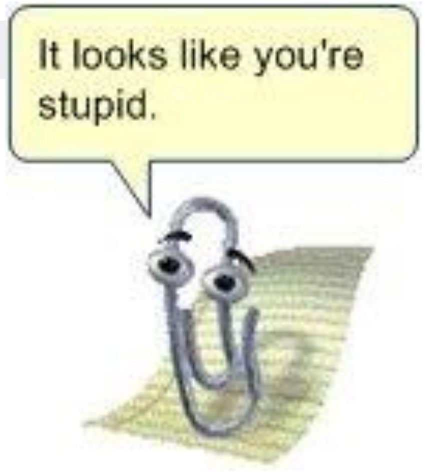
Burgas Free University

2012.2012

Introduction

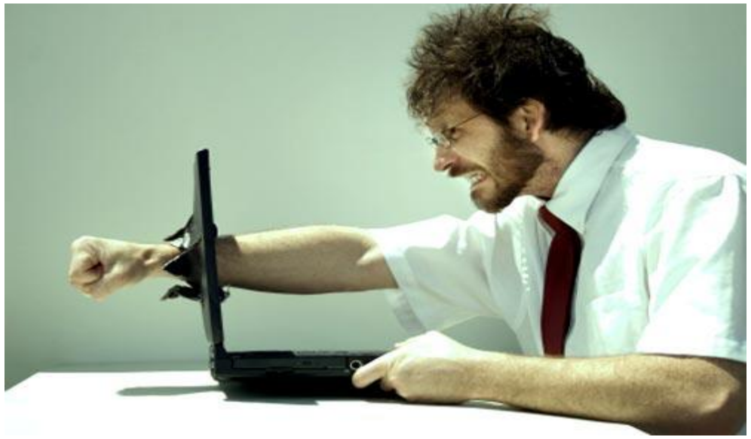
- Personal project.
- Under development since early 2011.
- Input is assembled into machine code.
- The name “Ziron” is a mix of OverHertz and Iron (a tough metal).
- Similar syntax to Assembly, C and Pascal.
- High performance assembler.

Story



- The beginning.
- During a project with debugging problems.
- Compiler assembling incorrect code.
- Hard to find fault in high level compilers.
- Solution: writing inline assembly.
- Under development

  Approx. 8 months total



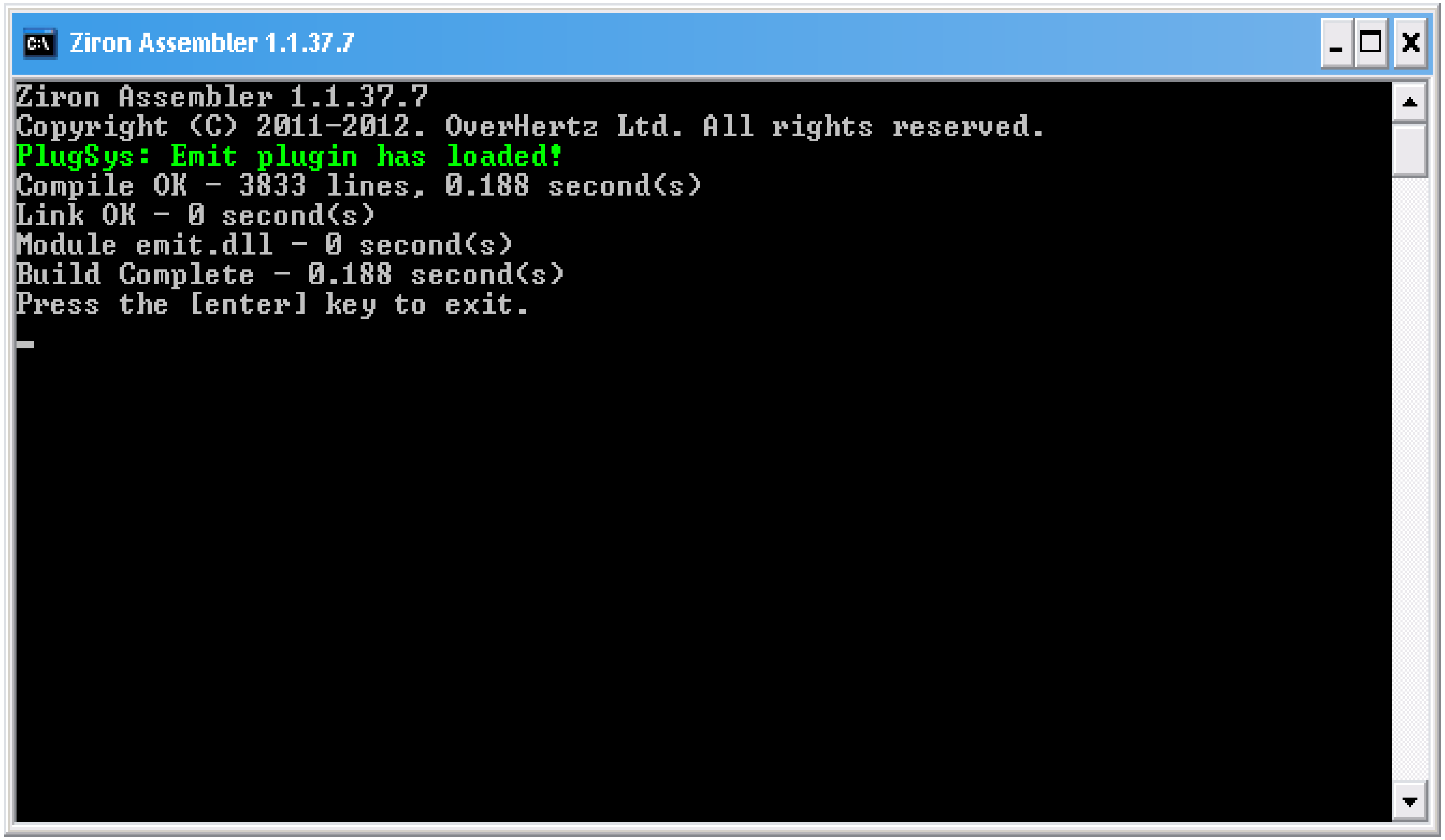
The problems of many modern day Programmers

- Lazy
- Rely solely on pre-written libraries.
- Missing out on their true potential.
- Expect too much from hardware manufacturers.
- Slowing down progression of Technologies.

Philosophy

- Designed for multiple programming styles.
- Bring high and low level code together.
- Making assembly easier to understand.
- For high performance optimised code.
- Small foot-print. (Using minimal resources)
- Closer to hardware level.
- Non-strict data and return types.

Ziron Assembler



```
Ziron Assembler 1.1.37.7
Copyright (C) 2011-2012. OverHertz Ltd. All rights reserved.
PlugSys: Emit plugin has loaded!
Compile OK - 3833 lines, 0.188 second(s)
Link OK - 0 second(s)
Module emit.dll - 0 second(s)
Build Complete - 0.188 second(s)
Press the [enter] key to exit.
_
```

Features

- OOP (Object Orientated Programming)
- Inline macro system.
- Easy to learn syntax similar to C and Pascal.
- Plug-in system and API.
- Common calling conventions such as stdcall, fastcall etc.
- High level block structures such as if, while, repeat and case statements.
- Custom data types using blocks (structs), typedef and enumeration.
- Syntax error reporting.
- Built-in linker. (output executable file format)

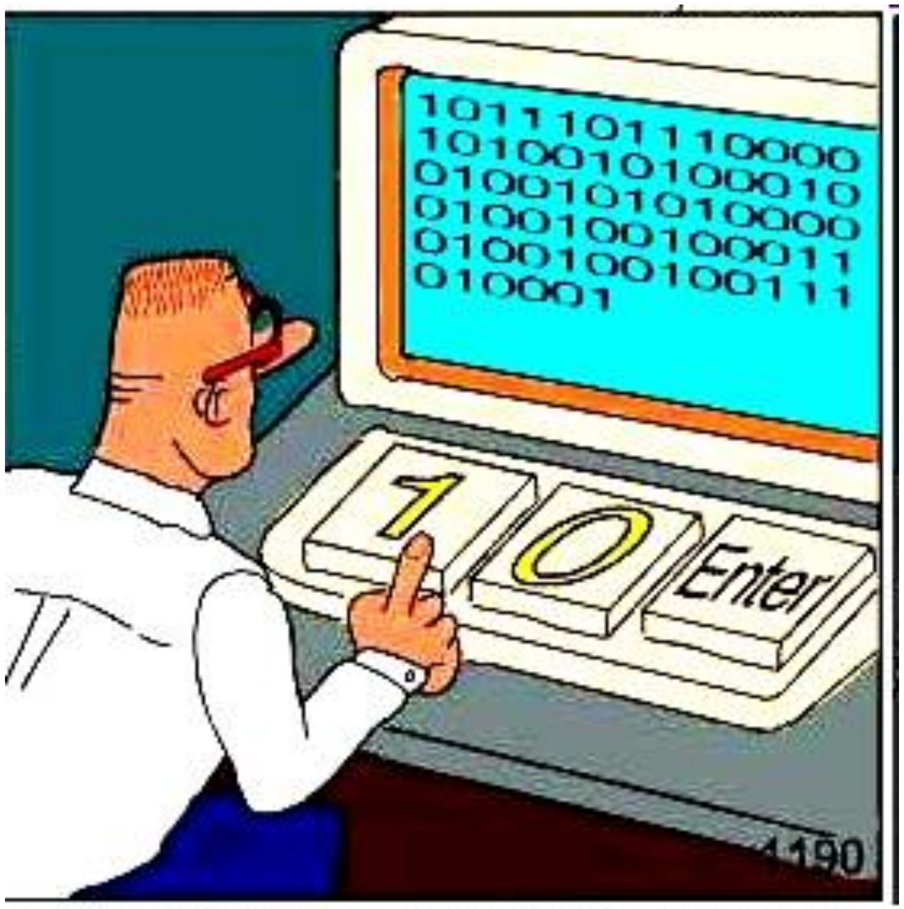
Syntax

- High and low level syntax.
- High level syntax similar to C and Pascal.
- Low level syntax can be written directly inline. (examples)
- Semicolon is often used to denote the end of a statement.
- Curly braces are used to group statements.
- Supports default parameters.

Hello, World! (console application)

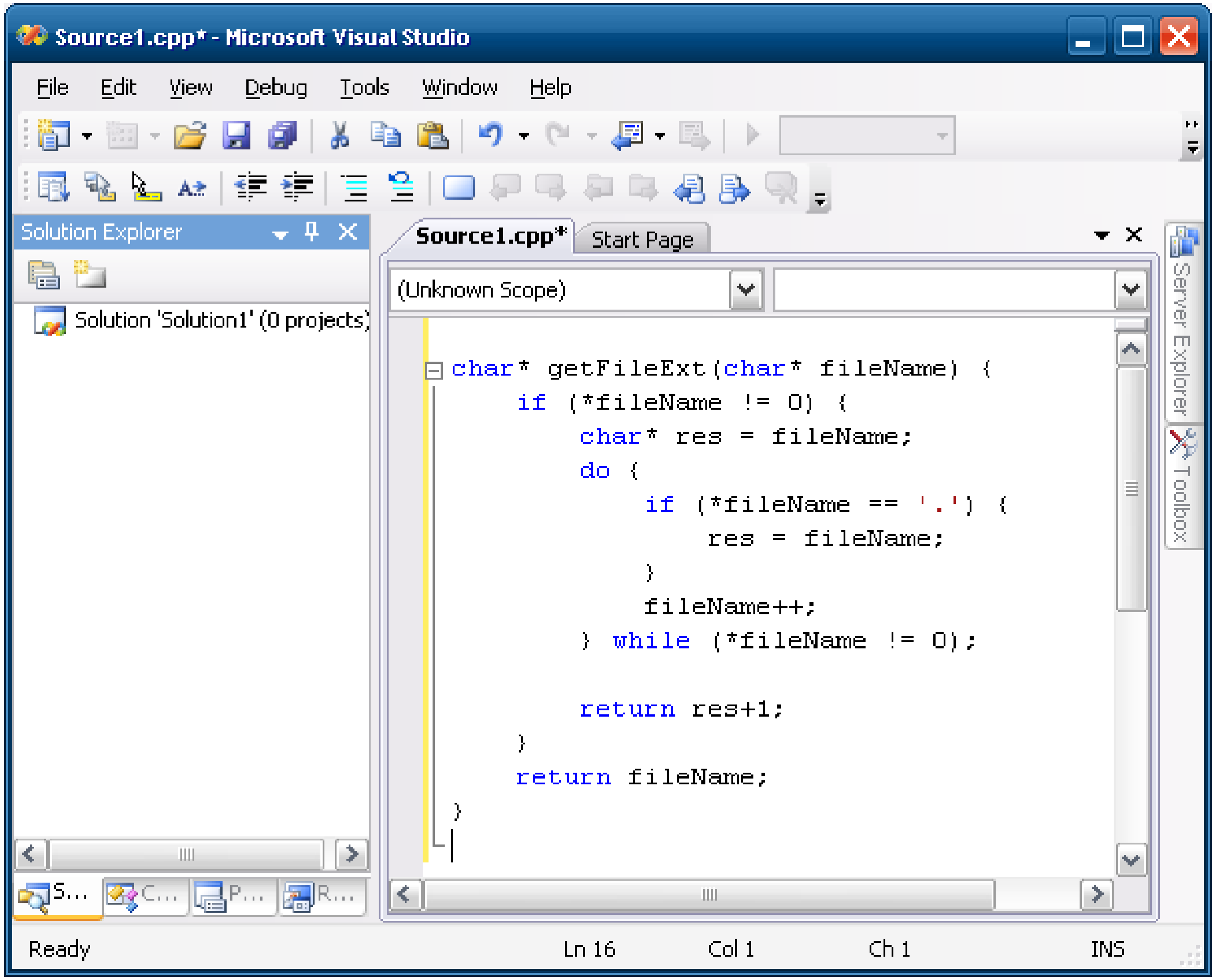
```
program WIN32CUI 'Hello World';

//included files
#include 'console.zir';

//show the hello world message and wait for a keypress
wait_key('Hello, World!');

//exit the program gracefully.
ExitProcess(0);
```

Comparison and Optimisation



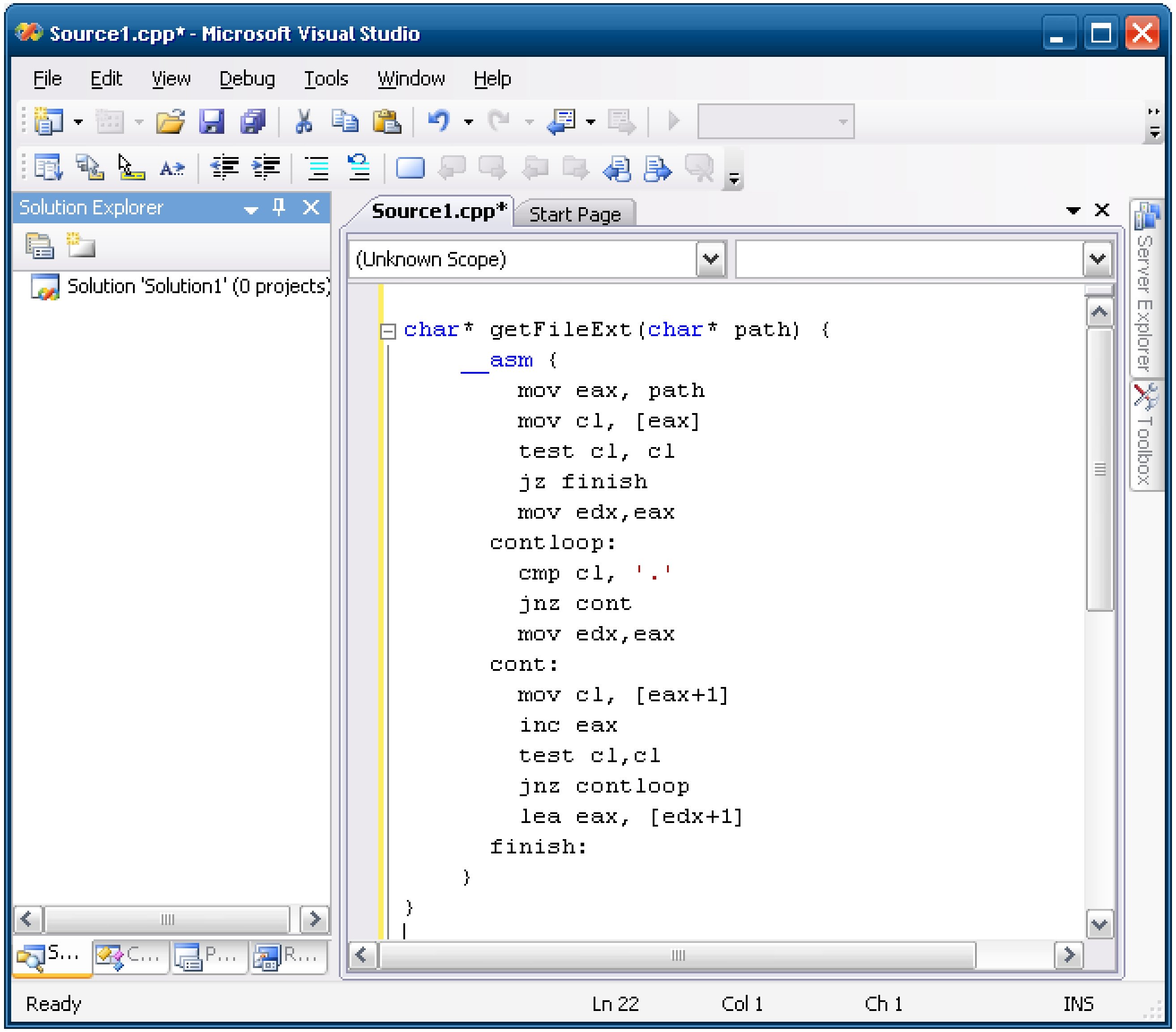
REAL Programmers code in BINARY.

C function


Source1.cpp* - Microsoft Visual Studio
File Edit View Debug Tools Window Help
Solution Explorer
Solution 'Solution1' (0 projects)
Source1.cpp*
Start Page
(Unknown Scope)
Server Explorer
Toolbox
Ready
Ln 16
Col 1
Ch 1
INS

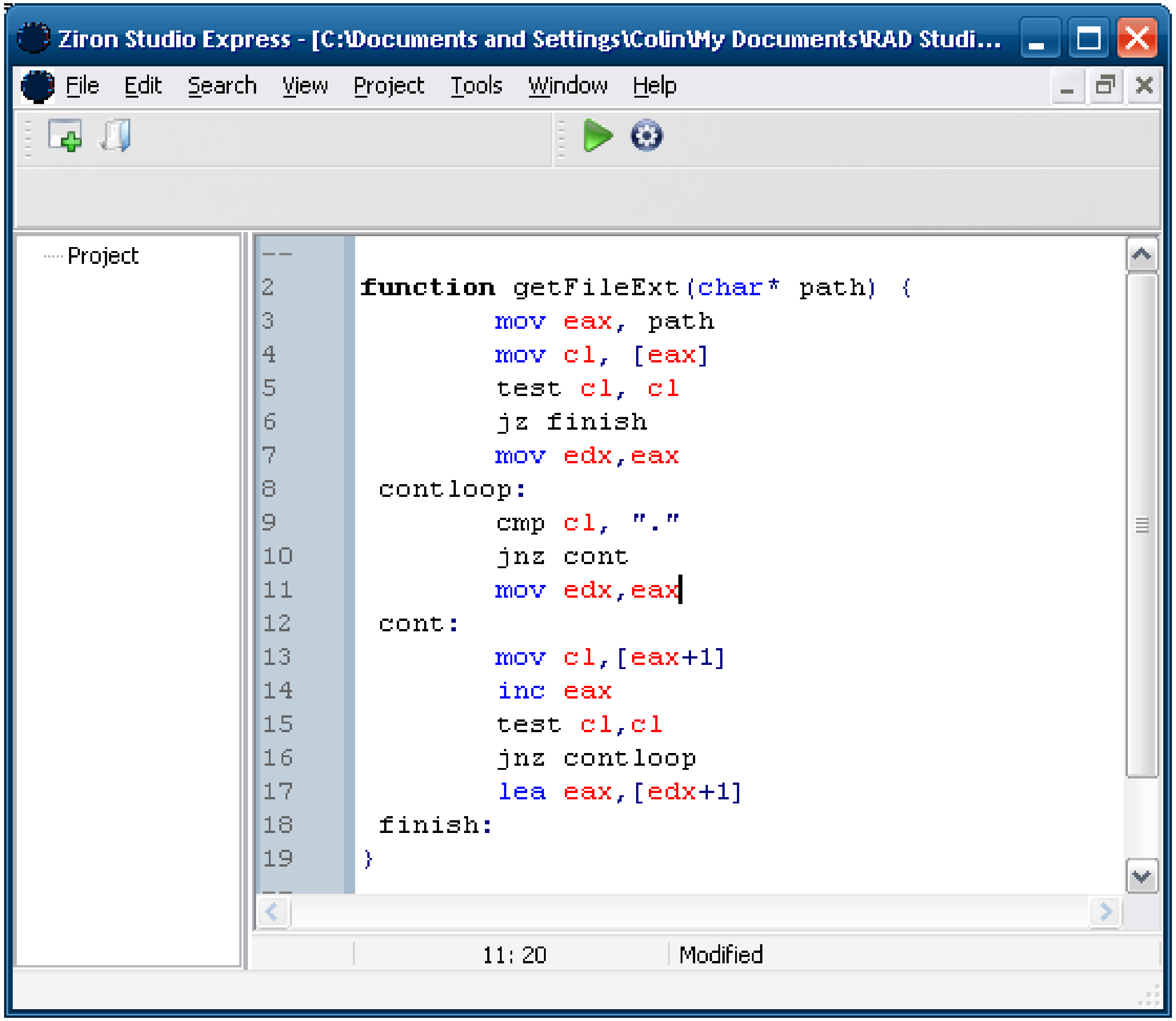
```c
char* getFileExt(char* fileName) {
    if (*fileName != 0) {
        char* res = fileName;
        do {
            if (*fileName == '.') {
                res = fileName;
            }
            fileName++;
        } while (*fileName != 0);

        return res+1;
    }
    return fileName;
}
```

Inner code disassembled


Source1.cpp* - Microsoft Visual Studio
File Edit View Debug Tools Window Help
Solution Explorer
Solution 'Solution1' (0 projects)
Source1.cpp*
Start Page
(Unknown Scope)
Server Explorer
Toolbox
Ready
Ln 22
Col 1
Ch 1
INS

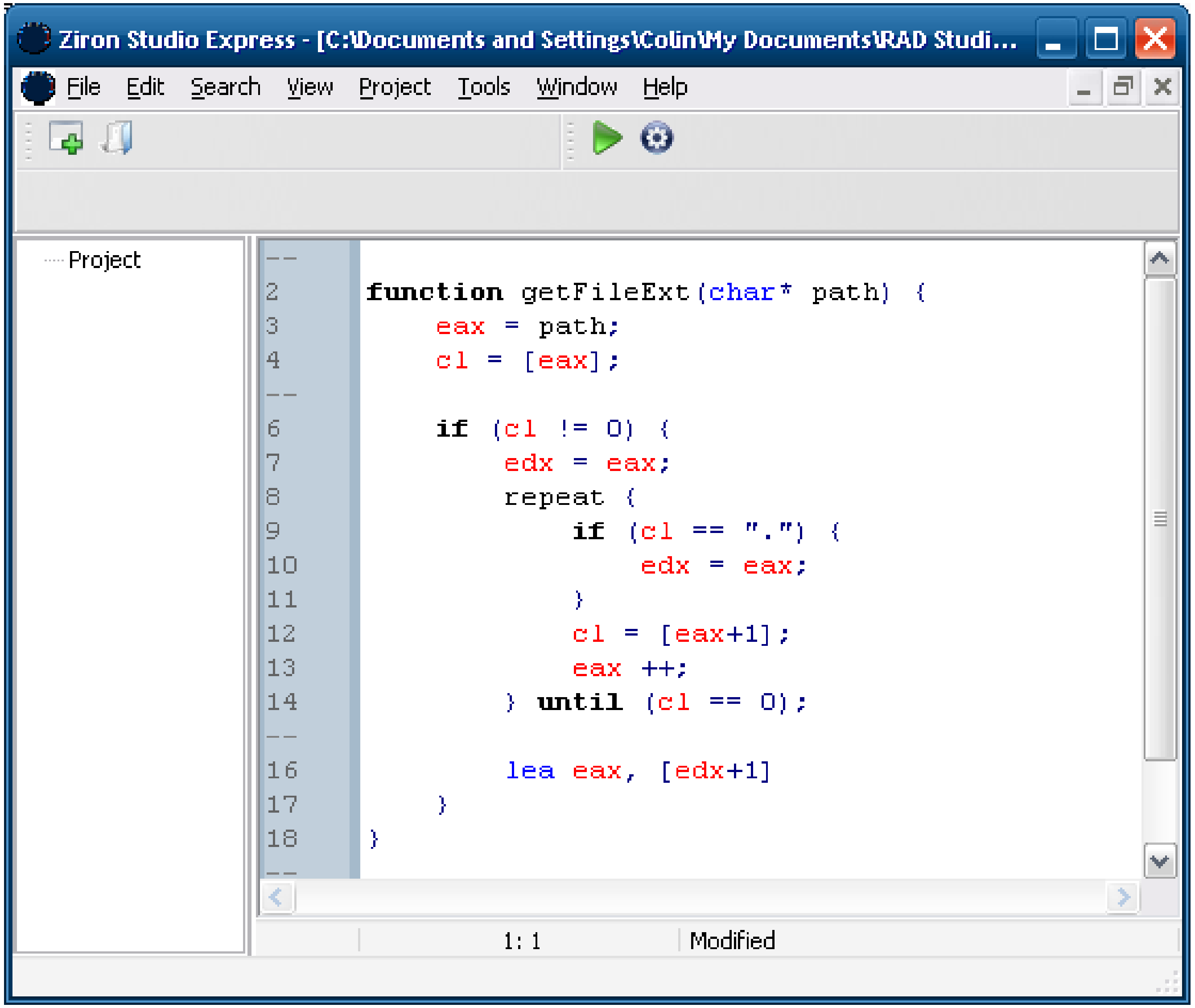
```
char* getFileExt(char* path) {
    __asm {
        mov eax, path
        mov cl, [eax]
        test cl, cl
        jz finish
        mov edx,eax
      contloop:
        cmp cl, '.'
        jnz cont
        mov edx,eax
      cont:
        mov cl, [eax+1]
        inc eax
        test cl,cl
        jnz contloop
        lea eax, [edx+1]
      finish:
    }
}
```

Ziron equivalent


Ziron Studio Express - [C:\Documents and Settings\Colin\My Documents\RAD Studi...
File Edit Search View Project Tools Window Help
Project
11: 20
Modified

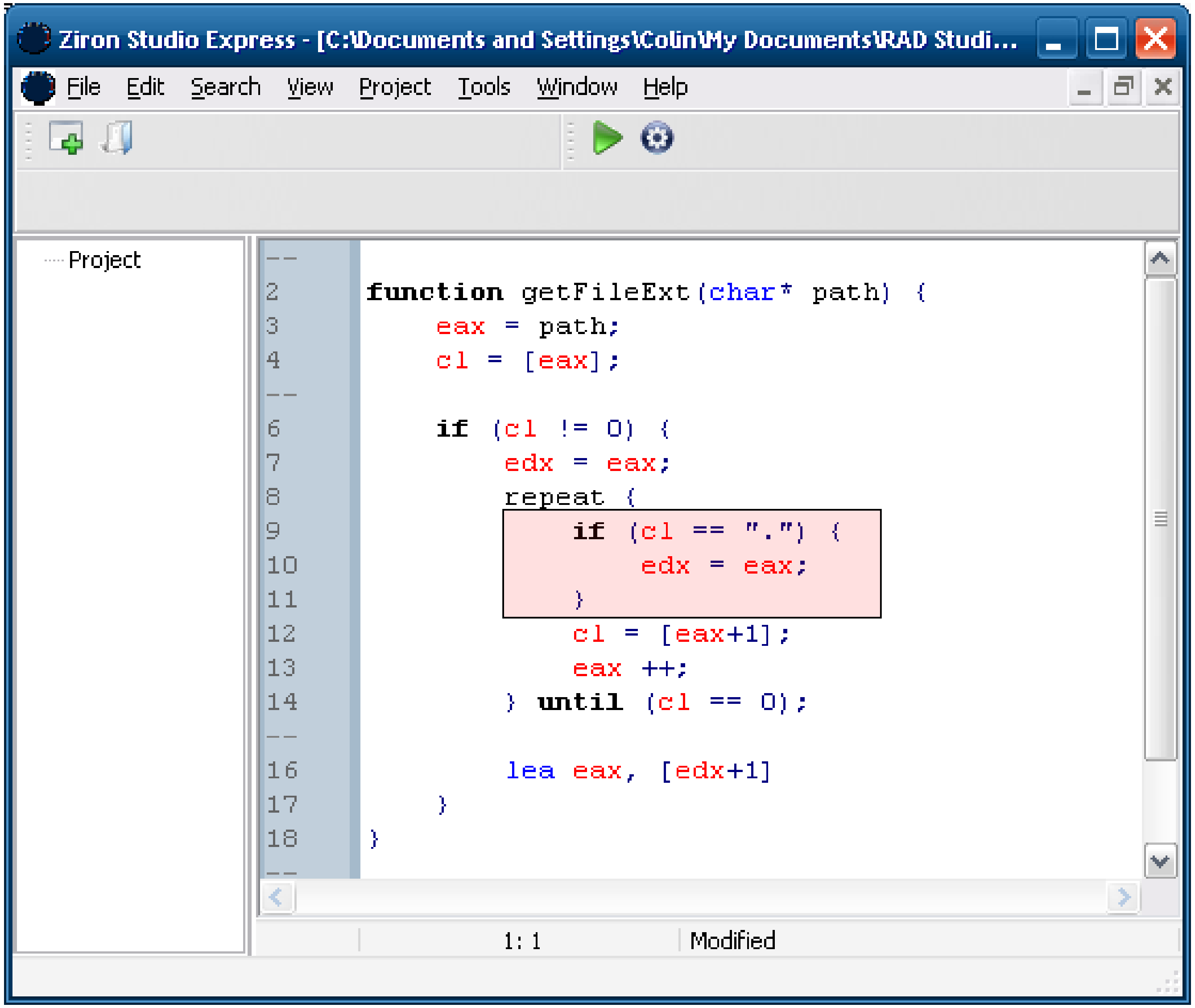
```
function getFileExt(char* path) {
        mov eax, path
        mov cl, [eax]
        test cl, cl
        jz finish
        mov edx,eax
 contloop:
        cmp cl, "."
        jnz cont
        mov edx,eax
 cont:
        mov cl,[eax+1]
        inc eax
        test cl,cl
        jnz contloop
        lea eax,[edx+1]
 finish:
}
```

Conversion to high level


Ziron Studio Express - [C:\Documents and Settings\Colin\My Documents\RAD Studi...
File Edit Search View Project Tools Window Help
Project
1: 1
Modified

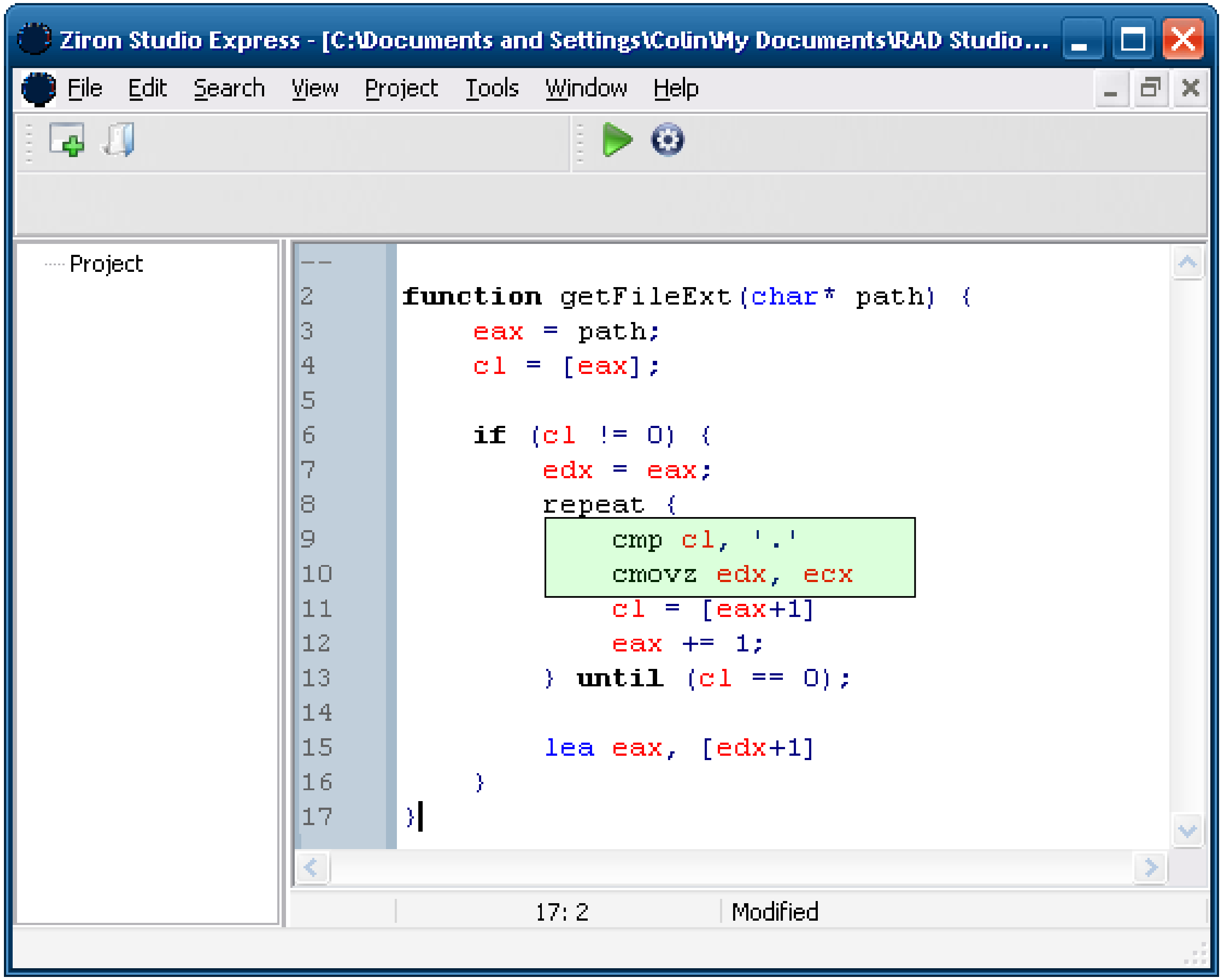
```
function getFileExt(char* path) {
     eax = path;
     cl = [eax];

     if (cl != 0) {
         edx = eax;
         repeat {
             if (cl == ".") {
                 edx = eax;
             }
             cl = [eax+1];
             eax ++;
         } until (cl == 0);

         lea eax, [edx+1]
     }
}
```

Optimisation (1/2)


Ziron Studio Express - [C:\Documents and Settings\Colin\My Documents\RAD Studi...
File Edit Search View Project Tools Window Help
Project
1: 1
Modified

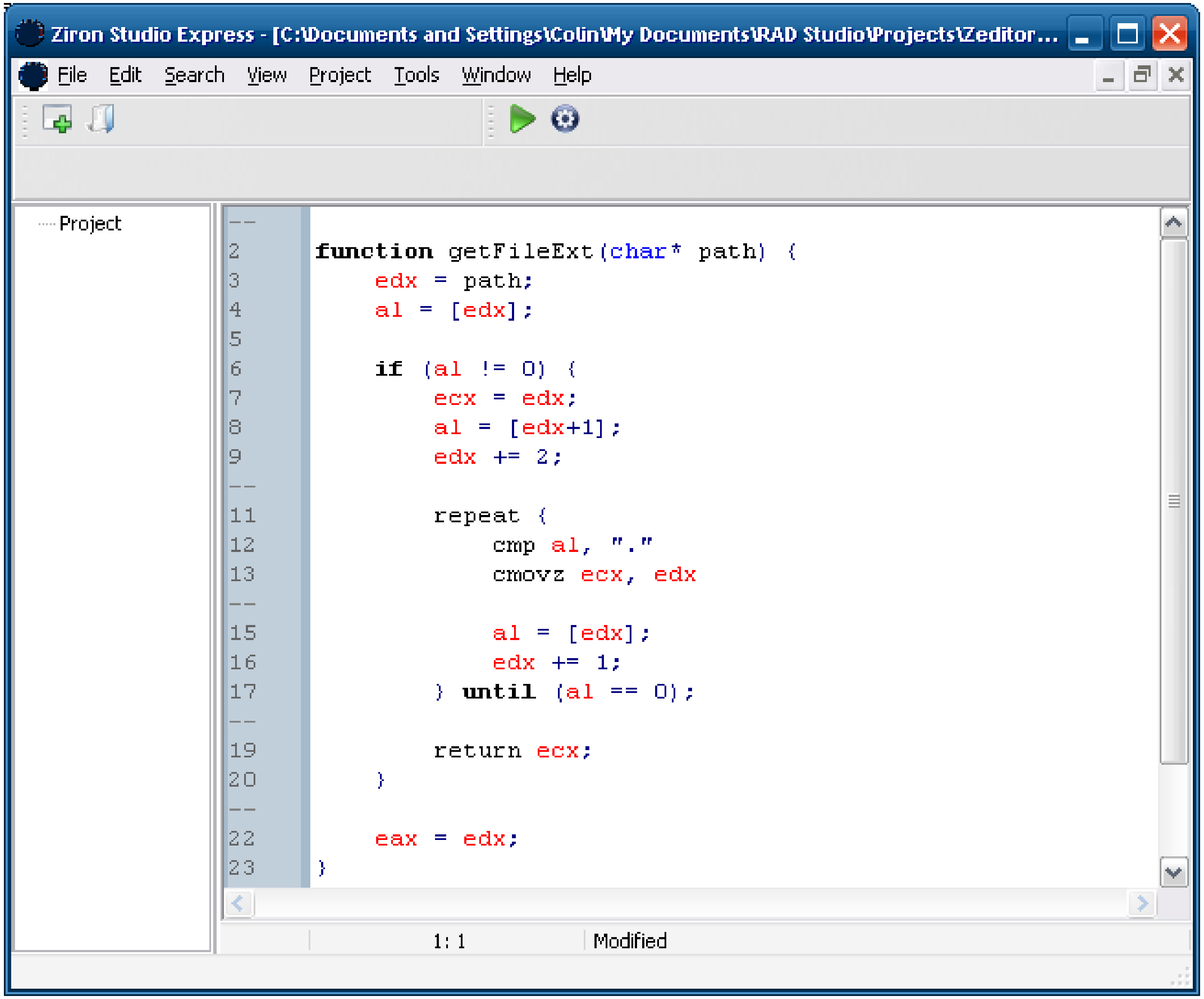
```
function getFileExt(char* path) {
    eax = path;
    cl = [eax];

    if (cl != 0) {
        edx = eax;
        repeat {
            if (cl == ".") {
                edx = eax;
            }
            cl = [eax+1];
            eax ++;
        } until (cl == 0);

        lea eax, [edx+1]
    }
}
```

Optimisation (2/2)


Ziron Studio Express - [C:\Documents and Settings\Colin\My Documents\RAD Studio...
File Edit Search View Project Tools Window Help
Project
17: 2
Modified

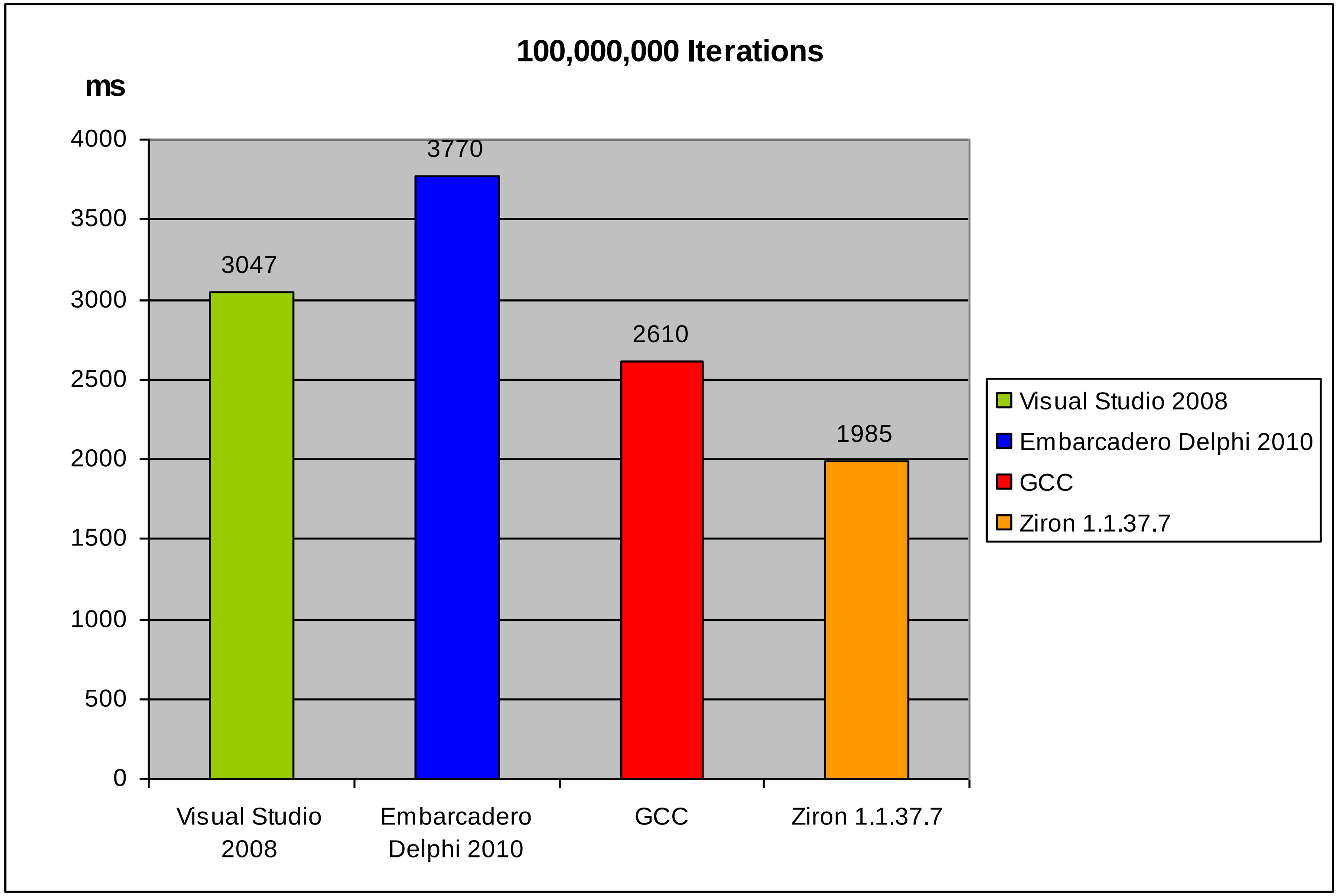
```
function getFileExt(char* path) {
    eax = path;
    cl = [eax];

    if (cl != 0) {
        edx = eax;
        repeat {
            cmp cl, '.'
            cmovz edx, ecx
            cl = [eax+1]
            eax += 1;
        } until (cl == 0);

        lea eax, [edx+1]
    }
}
```

Slightly better!


Ziron Studio Express - [C:\Documents and Settings\Colin\My Documents\RAD Studio\Projects\Zeditor...
File Edit Search View Project Tools Window Help
Project
1: 1
Modified

```

function getFileExt(char* path) {
    edx = path;
    al = [edx];

    if (al != 0) {
        ecx = edx;
        al = [edx+1];
        edx += 2;

        repeat {
            cmp al, "."
            cmovz ecx, edx

            al = [edx];
            edx += 1;
        } until (al == 0);

        return ecx;
    }

    eax = edx;
}
```

Statistics of our function in different Compilers / Languages



Current development

- Latest version: 1.1.38.1
- Supported architecture: x86
- Support website and Forum: http://www.codeziron.com
- Large library of pre-written functions.
- Documentation available on website.

The future of Ziron

- Ziron 2
- Open for suggestions.
- Modular.
- Support for other platforms via plug-ins, Linux, Mac-OS, Android.
- Better documented features.

Conclusion

- Ziron provides programmers a tool to optimise their software in a simple manner.
- Writing assembly can help you understand the workings of the CPU
- Ziron can help you to learn assembly language with ease.
- Has a simple syntax that is easy to learn by many programmers.

How to get involved!

- Register at the Code::Ziron Forum.
- Learn the syntax and language features.
- Write plug-ins to extend the features of the language and assembler.
- Submit bug reports and feature requests.
- If you would like to know or learn more, please ask at the forums!

Thank you!

visit the website at:

http://www.codeziron.com