1 Sunmary for clients

1.1 Overview

Our team achieved a simple but complete set of system binary tools for a high level programming language based on C++. We implemented components including editor, compiler, assembler, loader and simulator. Eventually, following the user manual, the user should be able to see our work and accomplishments.

Our final work presented itself as a Qt Mega project, which consists of several children projects that have seperate but connected functions. Upon running our Mega project, the user should have the following experiences. Firstly, an editor window will pop up and the user is indicated to enter his or her own program there. Secondly, when the user feels like finishing the editing, he or she can use the save button to store his program in a selected folder. Thirdly, our compiler is activated and pick up the stored program and begins compilation process, it will produce an equivalent program but in MIPS assembly language. Fourthly, assembler will read the file output by the compiler and convert it into a binary file. Thirdly, the program, now in binary format, will be loaded into the virtual memory and be executed by the simulator. This is the end of our flow.

1.2 User manual

Attention! Do not change the file folder name!

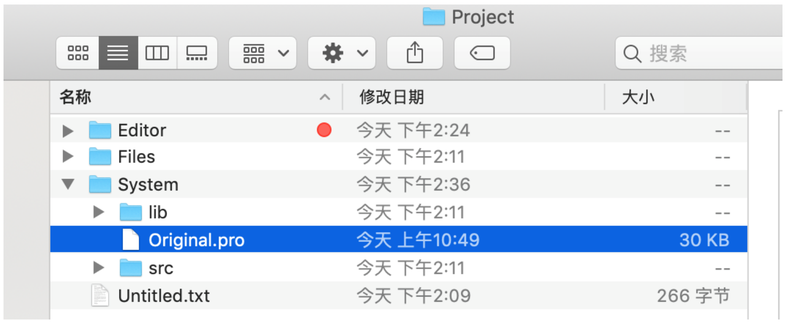
The following guides the user through the process of deploying our project, we aslo made a demo video available at

`https://cuhko365-my.sharepoint.com/:v:/g/personal/118010073_link_cuhk_edu_cn/EdASXJhSaB9HpB4hc1ux0nYBE6n1qX8jKQZitle58mAMjQ?e=ciH8pl`

- **Installation:** Download our Qt Mega project to your two personal computer. We have two versions available depending on your operating system, one for Windows 10, the other for MacOS. Folder for MacOS is named "Project", folder for Windows 10 is named "Project_Win". A Qt environment is required, version is expected to be the latest.

- **Prepare our project: (Mac as example)**

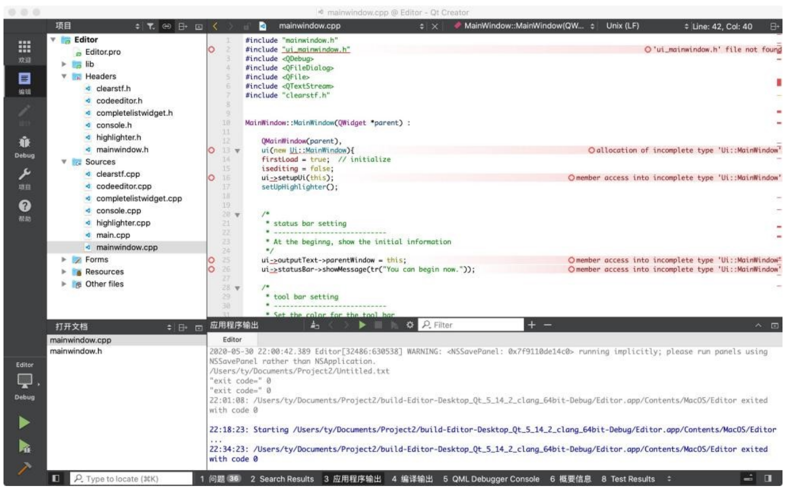
  1. Find and click Original.pro, and build this Qt project.



名称	修改日期	大小
Editor	今天 下午2:24	--
Files	今天 下午2:11	--
System	今天 下午2:36	--
lib	今天 下午2:11	--
Original.pro	今天 上午10:49	30 KB
src	今天 下午2:11	--
Untitled.txt	今天 下午2:09	266 字节

Figure 1: Original.pro

Notice, due to a lack of Debug folder to optimize memory, the following scenario might occur, if so, do not worry, close it and build it again.

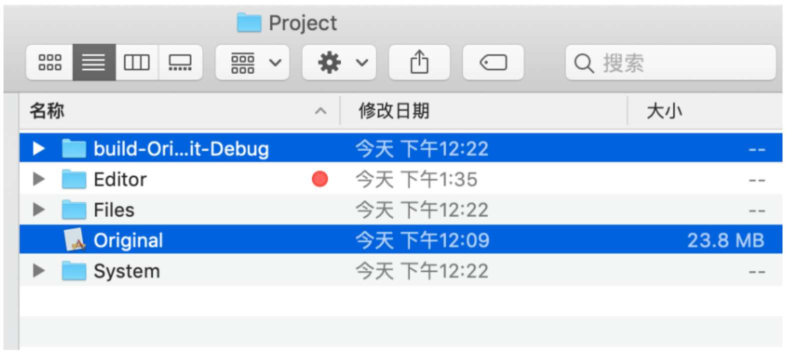


```
#include "mainwindow.h"
#include "ui_mainwindow.h"                          'ui_mainwindow.h' file not found
#include <QDebug>
#include <QFileDialog>
#include <QFile>
#include <QTextStream>
#include "clearstf.h"


MainWindow::MainWindow(QWidget *parent) :

    QMainWindow(parent),
    ui(new Ui::MainWindow){                         allocation of incomplete type 'Ui::MainWindow'
    firstLoad = true;  // initialize
    isediting = false;
    ui->setupUi(this);                              member access into incomplete type 'Ui::MainWindow'
    setUpHighlighter();


    /*
     * status bar setting
     * ---------------------------
     * At the beginng, show the initial information
     */
    ui->outputText->parentWindow = this;            member access into incomplete type 'Ui::MainWindow'
    ui->statusBar->showMessage(tr("You can begin now."));   member access into incomplete type 'Ui::MainWindow'

    /*
     * tool bar setting
     * ---------------------------
     * Set the color for the tool bar
```

```
2020-05-30 22:00:42.389 Editor[32486:630538] WARNING: <NSSavePanel: 0x7f9110de14c0> running implicitly; please run panels using
NSSavePanel rather than NSApplication.
/Users/ty/Documents/Project2/Untitled.txt
"exit code=" 0
"exit code=" 0
22:01:08: /Users/ty/Documents/Project2/build-Editor-Desktop_Qt_5_14_2_clang_64bit-Debug/Editor.app/Contents/MacOS/Editor exited
with code 0

22:18:23: Starting /Users/ty/Documents/Project2/build-Editor-Desktop_Qt_5_14_2_clang_64bit-Debug/Editor.app/Contents/MacOS/Editor
...
22:34:23: /Users/ty/Documents/Project2/build-Editor-Desktop_Qt_5_14_2_clang_64bit-Debug/Editor.app/Contents/MacOS/Editor exited
with code 0
```

Figure 2: Scenario 1

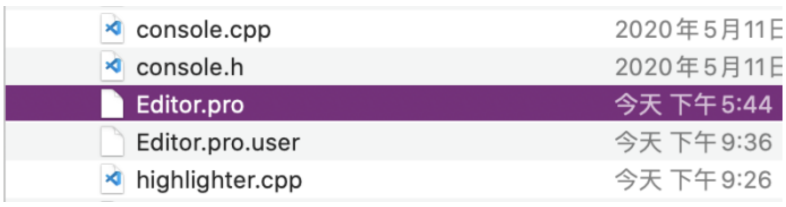
2. Under the Qt Debug folder, Find and move the Original.app (Original.exe for windows) into the Project(Project_Win for windows) folder.



名称	修改日期	大小
build-Ori...it-Debug	今天 下午12:22	--
Editor	今天 下午1:35	--
Files	今天 下午12:22	--
Original	今天 下午12:09	23.8 MB
System	今天 下午12:22	--

Figure 3: Move Original.app

3. Find and click Editor.pro, build and run it with Qt creator.



console.cpp	2020年5月11
console.h	2020年5月11
Editor.pro	今天 下午5:44
Editor.pro.user	今天 下午9:36
highlighter.cpp	今天 下午9:26

Figure 4: Editor.pro

- An editor window should pop up already. The user can then enter any program he or she wish to test it, note the following points.

  1. If the user simply copy and paste another program inside, the IDE will consider not editing is in procress, so make sure in this case, type something extra like a blank space to notify the IDE.
  2. The programming language we supoort is almost the same as C++. Minor differences are in ways of input and output, as well as our main program does not have a return value, libary inclusion and namespace. You can only edit under the main function. PS: You may find some prepared sample test files available under the project folder.

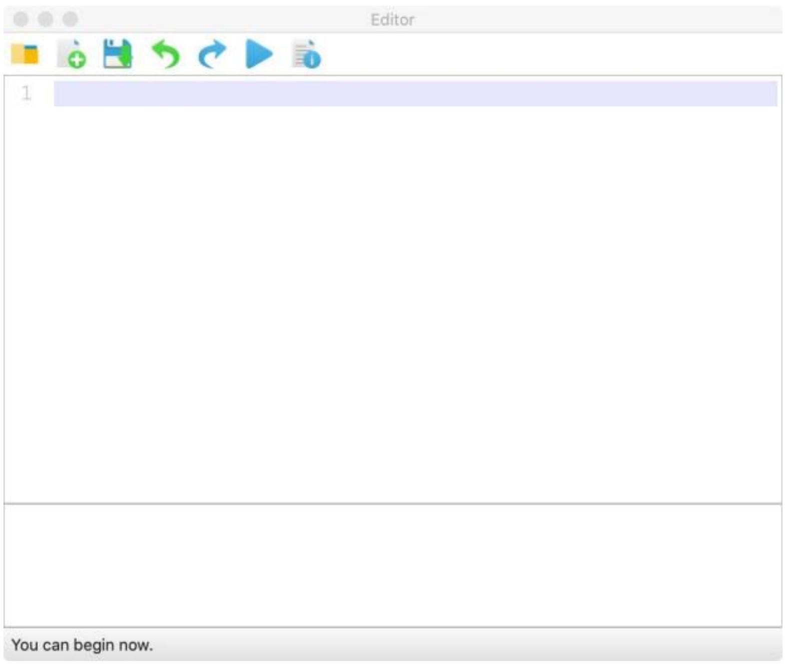


Figure 5: Editor.pro

3. The supported functions include all the essentials of empirical programming. For example, arithmetic at any level of complexties, logic operator(and/or), if, for and while loop, input and output. Four data types are supported: int, char, int array, char array (Like string).

   Notice, the user is recommended to look into appendix C for details about for functions of C++ we support and what differences we have with standard C++.
4. During editing, we implemented common features of C++ program editor that will bring convinience for the program. These features including, different color for diffrent syntax, auto-suggestion list of keywords, auto-completion of brackets, state bar for operation instructions etc. The user can look into our editor section to get more details.
5. When finishing the editing of the program. To execute the program, the user should press the run button directly and then he or she will be prompted to save the file in any directory and the execution follows.
6. Note, input must be entered in English Mode instead of Chinese or others to ensure the success of the execution.

- By now, we are capable of executing the program and the user should get the interaction or results he or she is expecting. However, note it is really hard to demonstrate our work of compiler, assembler and simulator, since they are mostly performing low-level software or even (virtual) hardware tasks, so they do not have any user friendly output. To shed more lights on these components, the user will see the next additional features other than the normal output of his or her program.

- First, the user can check the same folder he or she stored the original program, there will be two new files that are the corresponding MIPS assembly and binary equivalents, translated by compiler and assembler respectively.
- We believe the simulator is the most crucial component since it can leverage all the work done by the previous ones, from a high level language program, we now have a binary program that follows strict rules and syntax. Therefore, when executing the program, whenever an instruction pointed by the program counter is currently being executed, we will print the its location in memory. Whenever the content of any register is changed, we will print out this information. Whenever a branching occured, the destination will be printed. All data loading from the data segment of main memory will also be wise to the user.
- For user-friendly concerns, all helping information mentioned above will be printed out in a diffrent color then the actual output of the program.

2 Editor

2.1 Overview

The editor provides a visual graphical interface for the entire project, where all the interactions with the user in this project will take place. Through the editor, the user will be able to get an intuitive representation of the functions we have completed and achieve all the operations he needs. Overall, the design goal of our editor is user-friendly.

In the final project, our editor has implemented all the functions proposed in the proposal, such as file create, file open, save, save as (file operations) and cut, copy, paste, undo, redo (text editing functions) and so on. In addition, we have also implemented a "Run" button, added the result return area, and designed the application icon.

2.2 Related work

There have already been a number of well-established and fully functional text editors. When desgining functions of our editor, we mainly refer to the logic and methods of Notepad++ and Neatpad. We also took inspiration from the MAC's textedit and use it as a prototype.

As for the UI design, such as background color and text box display, as well as the realization of the connection between different buttons and their actual functions, we were inspired by the HJ editor[1] and refer to the Qt's open source library to see what kind of libraries are needed to build the editor.

The following is a list of the extended libaraires of Qt that we adopted in our implementation of editor. [2]