ICS Stage2 实验报告

许逸培 18307130103

目录：

分工情况：

单人模式

代码说明：

BasicSample.cs

IStandaloneFileBrowser.cs

StandaloneFileBrowser.cs

StandaloneFileBrowserEditor.cs

StandaloneFileBrowserLinux.cs

StandaloneFileBrowserMac.cs

StandaloneFileBrowserWindows.cs

通过windows窗口来读入一个文件的路径并显示到文件路径输入窗口中

File_Input.cs

从文件路径窗口中读入文件路径

Read_code.cs

Read_yo.cs

Read_ys.cs

从给定的文件路径中得到y86代码并写入内存，同时输出到code.out中

其中Read_ys.cs为ys文件读入，Read_yo.cs为yo文件读入

Control.cs

Updata.cs

Fetch.cs

Decode.cs

Excute.cs

Memory.cs

Write_back.cs

Program.cs

流水线的实现程序，分别对应与y86流水线中逻辑控制、时钟上升沿的更新、取指、译码、执行、访存、写回阶段，program.cs控制流水线整体的运行

Display.cs

View_Code.cs

View_Mem.cs

NumberSystem.cs

处理器显示功能的实现程序

display.cs为显示应用界面上的数值，

View_Code.cs打开存放y86代码的code.out并显示

View_Mem.cs打印当前内存中的数值到memory.out并显示

NumberSystem.cs控制把数据按照十六进制、有符号十进制、无符号十进制显示

Exit.cs

Run.cs

Step.cs

Next.cs

Show_Speed.cs

Y86处理器基本功能的实现，分别对应推出应用、持续执行、单步执行、执行到下一条命令进入取指阶段

其中Show_Speed.cs显示并控制持续执行的速度

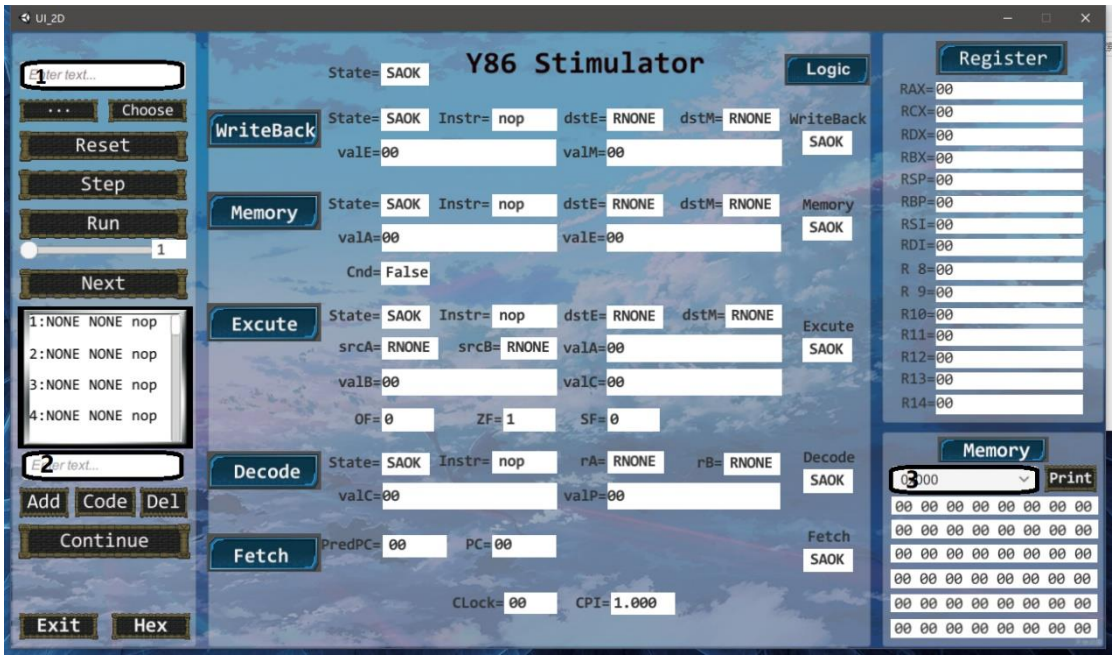
BreakControl.cs

控制断点跳转的实现程序

Keyboar.cs

快捷键的实现程序

模拟器使用方法：



运行环境：

Windows10 版本1803

.NET Framework 4.0及以上

Screen resolution选择1280*720。

基本使用方法说明：

1. 打开UI_2D.exe可执行程序文件，选择1280*720

2.在左上方文本框1中输入文件路径，或者点击文本框1左下方的· · ·按钮，通过windows窗口选择要输入的文件，之后该文件路径会显示在文本框1中。确认后按下Choose键可读入该文件

3. 单击左侧Run键可以让Y86持续运行，Run键下方滑动条可调节运行速度，从左往右调客加快速度，该应用的速度分为1~150，速度越快数值越大。

4. 单击左侧Step键可以让Y86执行一个时钟周期。

5. 单机左侧Reset键清空所有状态

6. 按下左下方Exit键退出。

附加功能使用方法介绍：

Next: 持续执行直到当前处于取指阶段的命令在内存中的下一条命令也进入取指阶段为止

Add/Del/Continue: 通过文本框2输入普通断点或条件断点，点击Add来添加断点，点击Del来删除断点，点击Continue来跳到下一个断点，若是条件断点则会跳到第一次满足条件时的断点位置。具体操作见下面的详细介绍。

Code: 点击左侧Code按钮可以跳出显示y86的二进制代码及其对应的汇编代码的写字板

Hex/Dec/uDec: 点击左下角的按钮可以切换显示的进制，按钮上现实的当前使用的进制，可以通过单击按钮使进制按照Hex/Dec/uDec即十六进制、有符号十进制和无符号十进制的顺序循环切换。

Print/3号框: 点击右侧的Print按钮可以跳出显示所有内存的，点击框3中的箭头符号可以切换在右下角显示的内存信息的开头位置，该窗口可以显示48位的内存（每一行显示8个）

快捷键：F9相当于Run、F10相当于Next、F11相当于Step、F6相当于Continue、F5相当于Reset

实现功能详细介绍：

可输入文件并判断是否合法：

通过上述方法输入文件路径后，然后按下Choose键可读入该文件。同时若不存在目标文件，则在文本框1中显示”No Such File”；若目标文件不是合法的文件类型（.ys/.yo），则在文本框1中显示“Invalid File”；若目标文件存在且

合法，则显示“Find the Target”。同时在点击Choose后应用上的所有状态全部清空，变成应用打开时的初始状态。

可输入ys文件并转化为yo文件：

在文本框1中可以输入ys文件路径，对于该ys文件，处理器会先翻译为yo文件并输出到exe文件同一个目录下的code.out中。对于ys文件，处理器还支持直接输入长度为1字节、2字节、4字节的静态类型的值，分别需要用.byte、.value、.long开头来表示显示的是长度多少的类型。

可控制速度运行并停止：

按完Choose键后，单击左侧Run键可以让Y86持续运行，Run键上方滑动条可调节运行速度，越往右运行越快，同时在滑动条右侧可以得到速度的数值大小（越大越快）。在点击Run后按钮会变成Stop，这时在点击Stop按钮可以令程序停止，并且按钮重新变成Run。

可执行基本操作：

单击左侧Step键可以让Y86执行一个时钟周期。

按下右上方Reset键清空应用所有状态，若再次运行必须读入文件路径。

按下左下方Exit键退出。

可跳转到下一条指令：

点击Next可以持续执行直到当前处于取指阶段的命令在内存中的下一条命令也进入取指阶段为止

可展示Y86流水线处理器运行过程中的寄存器状态；

15个寄存器显示在左上方面板Registers处。

可展示Y86流水线处理器运行过程中的各部件状态；

5个流水线显示在中间Write_Back/Memory/Excute/Decode/Fetch处，同时三个条件码显示在中间Excute阶段内。

在下方显示当前的时钟周期和流水线每个命令需要几个时钟周期（CPI)。

中间面板的右侧竖着显示流水线逻辑控制程序所控制的流水线各阶段的状态，SAOK为正常，SBUB表示该阶段为气泡，STAL表示该阶段为暂停。

中间面板左上方为程序总的运行状态。对于总的及个流水线寄存器的状态来说，SAOK为正常，SMEM为出现访存错误，SINS为出现错误指令

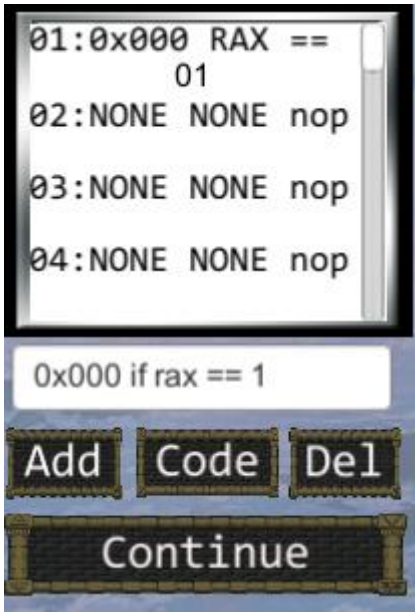
可展示内存：

右下方的面板Memory内可显示48位内存 (8为一行)。若要调节显示哪些位置的内存，可点击框3中的箭头符号，切换显示的内存信息的开头位置。若要看所有位置上的内存，可点击print，程序会在exe文件同一目录下打印一个memry.out来显示所有的内存。若当前的显示进制是十六进制，那么就十六位一行，否则就十位一行

可实时切换十进制十六进制：

Hex/Dec/uDec：单击该键可在十六进制、有符号十进制和无符号使进制之间切换各阶段与寄存器展示值的进制格式。

可跳转到断点并且设置条件断点：



通过文本框2输入两种类型断点，点击Add来添加断点，点击Del来删除断点，点击Continue来跳到下一个断点。

对于Del，在输入框中输入要删除的断点的序号（第几个断点），以0x或0X开头，则视为读入十六进制的序号，否则视为读入十进制表示的序号。之后点击Del。若输入不合法，则断点窗口没反应。

对于Add，按以下格式输入：

对于普通断点，只要输入一个数值就可以跳转。若该数值以0x或0X开头，则视为读入十六进制的断点，否则视为读入十进制表示的断点。该数值中间不能有空格；

对于条件断点，格式为”断点位置 if 寄存器名称 判断条件 数值”断点位置的格式与普通断点输入格式相同，接下去需要输入” if 寄存器名称 判断条件 数值“形式的字符串，if和寄存器名字之间需要加若干个空格，寄存器名称为全大写或全小写的rax/RAX、rcx/RCX等15个寄存器名称，判断条件为==、<、>、>=、<=之一，数值的输入格式也为以0x开头的十六进制或不加的十进制，负数的话需要在数值最前面加负号，数值、寄存器名称、if各自中间都不能空格，例如“R AX”就是不合法的。但是它们之间可以有空格，例如”if RAX== 1 ” 就是合法的。

条件断点会跳转到第一次满足条件（寄存器的值变成设定值）时的断点位置。

之后点击Add，若格式不合法，则断点窗口无反应。

程序可以跳转到第一个到达的断点，之后该断点会被用红色字体显示出来。

条件断点只能支持判断在该语句取指阶段就已经能得到的寄存器的值，即该寄存器的值需要在跳转到断点的时候或之前在译码、执行、访存、写回阶段就已经被算出，否则就只能根据在执行到断点当前已得到的寄存器的数值进行判断。因此，若需要在满足某个条件时跳转到某一条指令，可以在该指令后面加上若干个(4个就一定可行)nop指令并跳转到最后一个nop上，使执行到该nop操作时寄存器的值能够被更新出来。

同时，该断点个数最多只有100个，对于存储空间只有0xFFF的程序来说还是绰绰有余的。若当前跳到某个断点时，断点会显示在断点窗口的最上方并用红色字体标出。

按下Continue即可跳转。

断点可以通过拖动断点显示窗口右侧的滑轮来显示其余断点信息。

断点的标号、数值显示格式也受左下角十进制/十六进制显示格式的控制。

可打印处理器运行日志：

在处理器运行过程中后会在exe同一目录下打印log.out，显示运行时各流水线各部件的情况和15个寄存器的数值。考虑到输出文件的大小问题，值显示了前10000个时钟周期的情况。

可实现快捷键功能：

F9相当于Run、F10相当于Next、F11相当于Step、F6相当于Continue、F5相当于Reset

实现细节：

使用C#/Unity的后端加前端的结构进行图形界面的实现。

首先把c++版本的流水线改编成c#版本，同时尝试了WPF制作前端失败后开始用C#实现前端。

Ys翻译器：一行一行读入字符串。若当前字符串被注释就跳过。否则判断当前命令是否是.pos（定义之后二进制命令在内存中的位置），.align（更改让当前内存位置），.quad/.long/.value/.byte（读入保存在内中某一长度的数值），halt/nop等命令，后面加: 的函数名。对于这些情况一一判断并进行相应的字符串处理。

断点处理程序：通过字符串处理判断是哪种类型的断点和是否合法，之后用数据结构存下来，在每个时钟周期结束后判断当前PC是否是断点、寄存器的值（需考虑使用冒险获得在译码、执行、访存、写回的数据）是否与满足条件

讨论/改进措施：

未来主要会向三个方面进行改进：

1. 考虑是否支持版本回退（支持会给予时间、空间极大的负担）；
2. 美化用户界面设计；
3. 考虑如何支持完整版的条件跳转

心得体会：

这是我第一次写前端的应用，使用unity让我体会到了前端和后端交互的过程，同时也体会到了编写应用的过程中不断改进功能、不断创新、不断竞争的过程。

References：

https://github.com/gkngkc/UnityStandaloneFileBrowser