==<font color="#800000">'''关键词'''</font>==

<font color="#808080">Mathematica、BooleanFuction、Hack Machine Language、instruction、A,M,D、jump、von Neumann architecture.</font>

==<font color="#800000">'''逻辑模型'''</font>==

[[File:第五周逻辑模型wsj.png|1100px]]

==<font color="#800000">'''学习报告正文'''</font>==

===<font color="#800000">'''课程目标'''</font>===

<font color="#808080">

\*完成project4

\*完成第五周学习报告，总结第五周学习内容

\*尝试运用 Mathematica完成前几周的作业

</font>

===<font color="#800000">'''课程内容'''</font>===

<font color="#808080">

'''1.project4程序的编写'''

\*Mult

\*\*Mult的作用

\*\*\*With this terminology in mind, this program computes the value R0\*R1 and stores the result in R2.

\*\*\*Note that in the context of this program, we assume that R0>=0, R1>=0, and R0\*R1<32768 (you are welcome to ponder where this limiting value comes from). Your program need not test these conditions, but rather assume that they hold.[http://nand2tetris.org/04.php]

\*\*\*就是说它一个能够运算R0\*R1的程序，并能把运算结果寄存在R2里，R0、R1都是正整数，且R0\*R1不会溢出。

\*\*Mult的构建

\*\*\*首先我们需要用一些比较清晰的“标签”来对应每一种机器语言的指令。通过第四章的学习，我知道了机器语言的几种基本的运算符和几种基本构件模式，比如选择语句和循环语句的构建，由于我们编写的ALU中只有加减两种运算方式，所以想要构建整数间的乘法，比较容易能想到的是R0\*R1=R0+R0+……+R0（R1个R0）,而相同的相加运算可以做成一个LOOP，使用负责内容责任圈的同学们教授的方法来检验是否加够了R1次从而决定是否跳出LOOP，是一个比较简单的程序。

\*Fill

\*\*Fill的作用

\*\*\*The program runs an infinite loop that listens to the keyboard input. When a key is pressed (any key), the program blackens the screen, i.e. writes "black" in every pixel; the screen should remain fully black as long as the key is pressed.[http://nand2tetris.org/04.php]

\*\*\*1.这个程序是一个无限循环，只要程序开着就会一直运行下去

\*\*\*2.这个程序能在键盘有输入时使屏幕全黑，在键盘没有输入时使屏幕全白。

\*\*Fill的构建

\*\*\*构建时有几个小技巧

\*\*\*1.screen是地址为16384-24575的一段内存，当每段内存里的16位的二进制数的某一位为“1”时，它所对应的屏幕上的一个像素块就是黑的，如果为“0”，则为白的，所以要使一个内存内的所有位的数都为1，就应把该内存对应的值设为-1，即<pre>M=-1</pre>

\*\*\*2.A寄存器既可以作为外源常数的输入，又对应着M的address，所以当A变化了（不管是被赋值还是通过计算变化），M对应的Register也立即发生变化，这样以来只要做一个LOOP，在这个LOOP中创建一个每次数值加1的中间值i，并在每次回圈时把它的值赋给A，再改变M的值为-1，就能使地址16384-24575中的所有Register的存储全变为-1。这样以后再在回圈中做一个检测键盘是否有输入的选择语句，如果没有输入，就跳到一个有类似语法结构但是将M设为0的回圈中，这个程序就构建完成了。

\*\*Fill中部分语句

<pre>(DONE)

D=A

@i

M=D

(BLACK)

@KBD

D=M

@WHITE

D;JEQ

@i

D=M

A=D+1

M=-1

@i

M=M+1

D=M

@24575

D=A-D

@BLACK

D;JGE

@DONE

0;JMP</pre>

\*'''感悟'''

\*\*1.当A变化了（不管是被赋值还是通过计算变化），M对应的Register也立即发生变化

\*\*2.能够做成回圈的语句尽量做成回圈

\*\*3.学会将C语言等高级语言的编写模式搬运到低级语言中来。

'''2.顾老师的Mathematica课程'''

\*在Mathematica中可以自己定义函数和语句，并在这个文档的命名空间里赋予它们各自的名字

\*去除提示的程序的构建

\*\*1.NotebookDirectory[]，用来提取相同目录下目标文件的目录。

\*\*2.Import[地址,具体格式]，用来导入某个文件中某种形式的内容。导入hdl文件中的text文本

\*\*如果将其转换为原代码可以看到一些其他的东西比如“/n”“/t”

\*\*[[File:M1wsj.png|1000px]]

\*\*3.StringSplit[xxx，patt]用来将两个指定分隔符之间的内容抽取出来并列举出来。但是按照顾老师的操作后还有一些空格不知道是一些什么符号

\*\*:=表示定义，Fuction[x\_]:=表示这个函数的自变量是x，例如IsCommentedLineQ[line\_] :=，它的变量就是Line

\*\*4.Module[{x,y,……},expr],指定在 expr 中出现的符号 x、y、…… 应被当作局部值。但我对局部变量的理解还有待加深。

'''3.对计算思维的学习感悟'''

\*在完成作业时感觉Nand2Tetris这门课里的内容十分符合计算思维与系统设计的理念，计算机的每个部分我们都要从最底层的0101……开始编写最底层的Functon、Contract，再对这些fuction进行校验之后再拿这些函数来编写更高级的函数，而在高一阶的函数编写中，我们无需去考虑比它低一层的函数的内容，比如说我们不用去考虑Nand的更底层是什么，那不在我们考虑的范围内，或者说，不在圈子内，如之前的课程介绍里所说，对于复杂的系统模型，套用一个迭代的概念: “It's turtles all the way down”，现在我似乎能理解这个概念了。

\*在上程序课的时候老师说要养成“计算思维”，也就是要去思考计算机会怎么做某一件事，我觉得是有道理的。

==<font color="#800000">'''创建词条'''</font>==

[[Import]]，[[Export]]

</font>

==<font color="#800000">'''参考文献'''</font>==

<font color="#808080">

\*网络词条

\*\*隐喻[http://baike.baidu.com/subview/117044/9422190.htm#viewPageContent]

\*\*概念隐喻[http://baike.baidu.com/link?url=pH2sZRETaCklOPHB-MkvtMm5q1LRxloldbgMrBcpeUInIPU3uv2GhezvkzNyDGp\_oKhbv1XL5dYoHGq403qicK]

\*[http://nand2tetris.org/02.php| Nand2Tetris project4]

\*[[《超越学科的认知基础》2015秋孙一乔学习报告-第六周]]

\*[[2016年秋-Nand2X-计算思维和系统设计基础]]

\*顾学雍，BooleanFunctions，2016

</font>

==版权声明==

[[File:BYNCND.png | link=http://creativecommons.org/licenses/by-nc-nd/3.0/cn/ ]] [http://creativecommons.org/licenses/by-nc-nd/3.0/cn/ 署名-非商业性使用-禁止演绎 3.0 中国大陆 (CC BY-NC-ND 3.0 CN)]