实验八 状态机及键盘输入

实验报告

181860085 汤昊

数字电路与数字系统二班

邮箱：1174639585@qq.com

2019.10.16

一．实验目的

学习状态机的⼯作原理，了解状态机的编码⽅式，并利⽤ PS/2键盘输⼊实现简单状态机的设计。

二．实验原理

状态机由状态寄存器和组合逻辑电路构成，能够根据控制信号按照预先设定的状态进行状态转移，是协调相关信号动作、完成特定操作的控制中心。有限状态机主要分为2大类：

第一类，若输出只和状态有关而与输入无关，则称为Moore状态机。

第二类，输出不仅和状态有关而且和输入有关系，则称为Mealy状态机。

状态机的设计可以有两种方法，一种是对状态进行逻辑抽象和化简，再进行状态分配和编码，推导出输出方程和状态转移方程，优势还需要进行自启动检查。另一种简化的方法是先进行逻辑抽象和状态设定，之后分析各个状态的可能输入以及各个输入下的输出和下一状态。对于本实验ps2键盘实验将采用此种方法进行复杂的状态机设计。

三．实验环境及器材

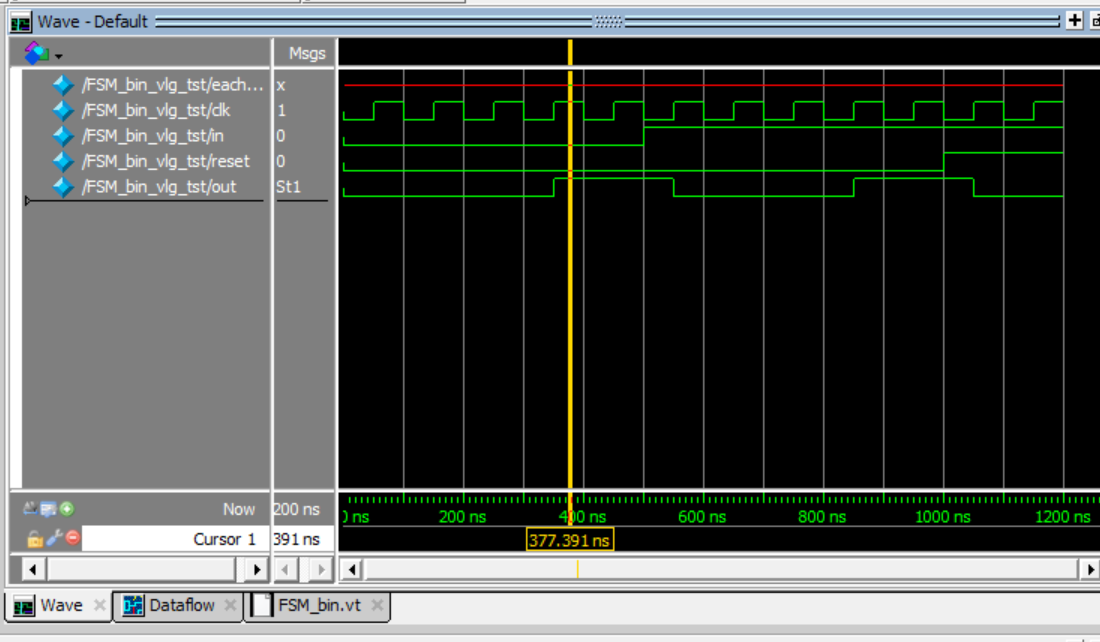
开发软件：quartus prime 17.1

开发器材：DE-standard 开发板

ps2键盘

四．简单状态机

Moore型状态机仿真模拟结果



可以看到在in=0保持了四个上升沿后，out=1，同样in=1保持了四个上升沿后out=1，且输出在in改变之前一直保持。另外out并不是立即响应in的变化，而是等到下一个上升沿状态变化才改变，这体现了Moore型状态机的特点。

问题：⽶⾥型状态 的设计与此有何不同？

答：米里型的设计输出不仅与状态有关，还和当前输入有关，因此在决定输出时还要考虑当前的输入。

五．ps2接口控制器及键盘输入

(1)模块划分

数据

数据

键盘输入处理

键盘控制器

键盘输入

数据

读信号

数码管及LED

显示

对应quartus项目中的

Keyboard：键盘输入处理

单个数码管显示

Ps2\_keyboard：键盘控制器

Display：显示

HEX：单个数码管显示

(2)状态分析及功能设计

* 读信号处理

Ready=1

Ready=0

一个周期

读完nextdata=0置零一个周期：

键盘处理模块设置寄存器data1，在ready有效时接收键盘控制器的数据data0，同时将nextdata\_n置0，每次上升沿都判断nextdata\_n是否为0，是则置为1，由于采用非阻塞式赋值，读完数据nextdata\_n置0的结果要在下一个周期才能体现，故实现了读完数据将nextdata\_n置0一个周期的要求

* 按键按下和释放

接着输入0x12

输入非F0

F0

字母/数

接着输入0x14

输入非F0

其他输入

对断码的处理：

接受到F0时，设置标志位break=1，这样下一次就不是正常读取，而是根据扫描码决定显示的变化，如果是shift，ctrl则对应LED灭；而对caps（不是松开就灭，而是松开代表一次切换结束）要设置可以改变的标志，以便下次按下caps可以改变信号，其余把给显示模块的使能信号en置0，低四位数码管全灭。

* 按键按下次数计数

初始/断码后/两次扫描码（非断码）不一样

首次输入

持续按下只计数一次：

设置标志位countable，在第一次读取时计数，countable置0，接收到断码后countable置1。但这样shift，ctrl键就不能和字母数字键同时按下，因此将每次的扫描码还送到一个temp寄存器保存，和送来的扫描码比较，这样上一次的扫描码和这一次的扫描码不一样时也可计数一次，当字母和功能键一起按下时，必然会出现一次不一样的情况。

这里不需要考虑断码，断码会进入break=1的状态，其扫描码不会被记录，不会计数，只会影响显示。

Caps松开后首次按下

* 大小写控制

Caps持续按下

Caps松开

Caps松开

caps持续按下

Caps松开后首次按下

实际的大小写切换是按下即切换状态，一直按下保持不变，松开后再按一下切换状态。因此使用标志位change\_caps，为1代表状态可切换，为0代表状态不可切换，在第一次读取时就将其置0，在接收到断码后再将其置1。

* 显示模块的设计

（1）扫描码显示：

送来的扫描码已经被顶层模块过滤掉了断码，因此直接给数码管显示。

（2）ASCII码显示：

用case语句将扫描码转换为ASCII码

（3）编码显示设置：

对于扫描码和ASCII码都采用十六进制显示，而按键次数采用十进制显示，因此在显示模块中给数码管传入一个进制信号，决定按什么进制显示

16进制的数码管显示如下（9之前与十进制相同，省略）：

4'b1010:HEX0=7'b0001000;

4'b1011:HEX0=7'b0001100;

4'b1100:HEX0=7'b1000110;

4'b1101:HEX0=7'b0100001;

4'b1110:HEX0=7'b0000110;

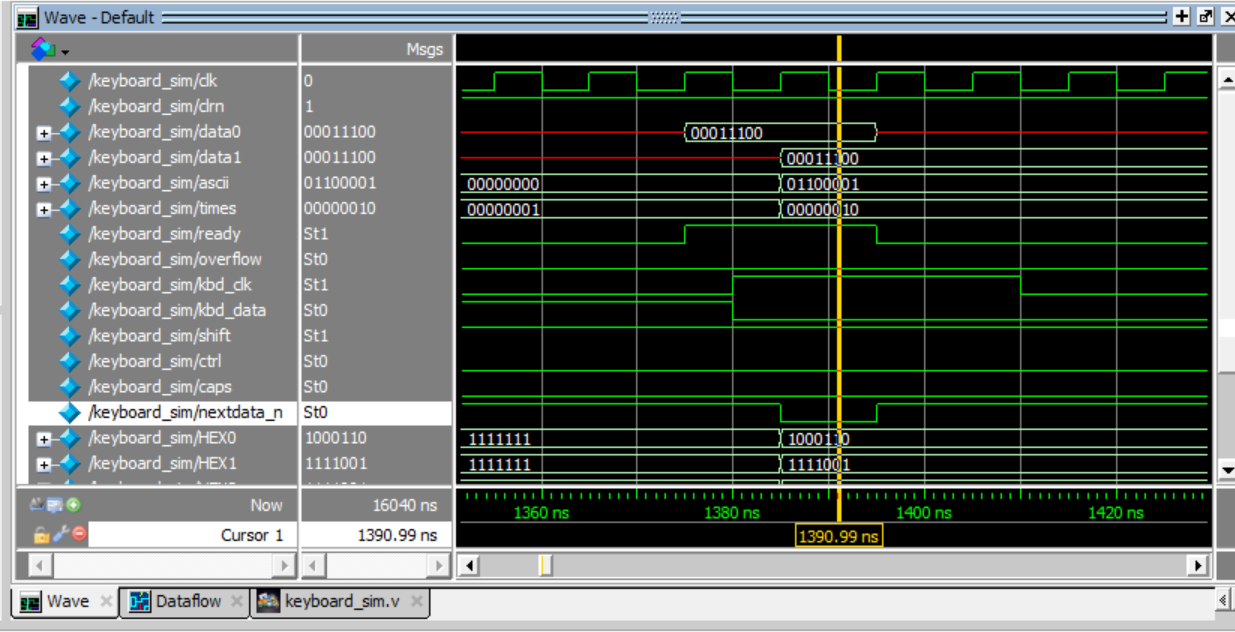
4'b1111:HEX0=7'b0001110;

（4）使能端设置：

每个HEX模块都有使能端，在输入断码时，处理模块发送显示使能无效的信号，通过显示模块给HEX模块。

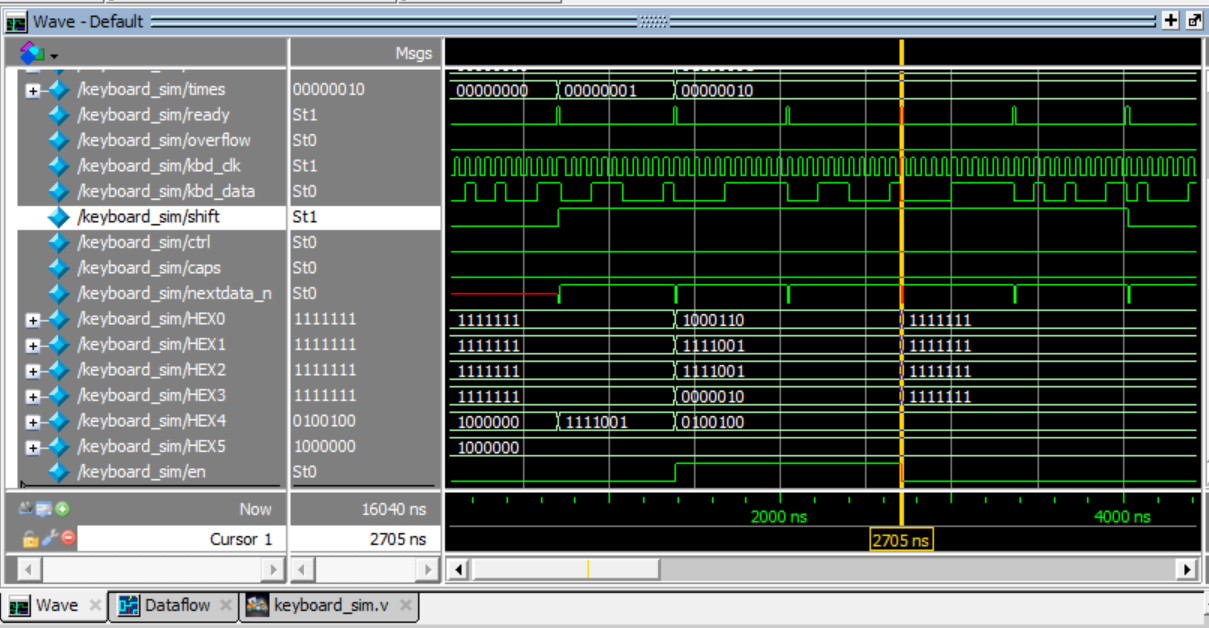
（3）仿真模拟

* Nextdata\_n置为0一个周期



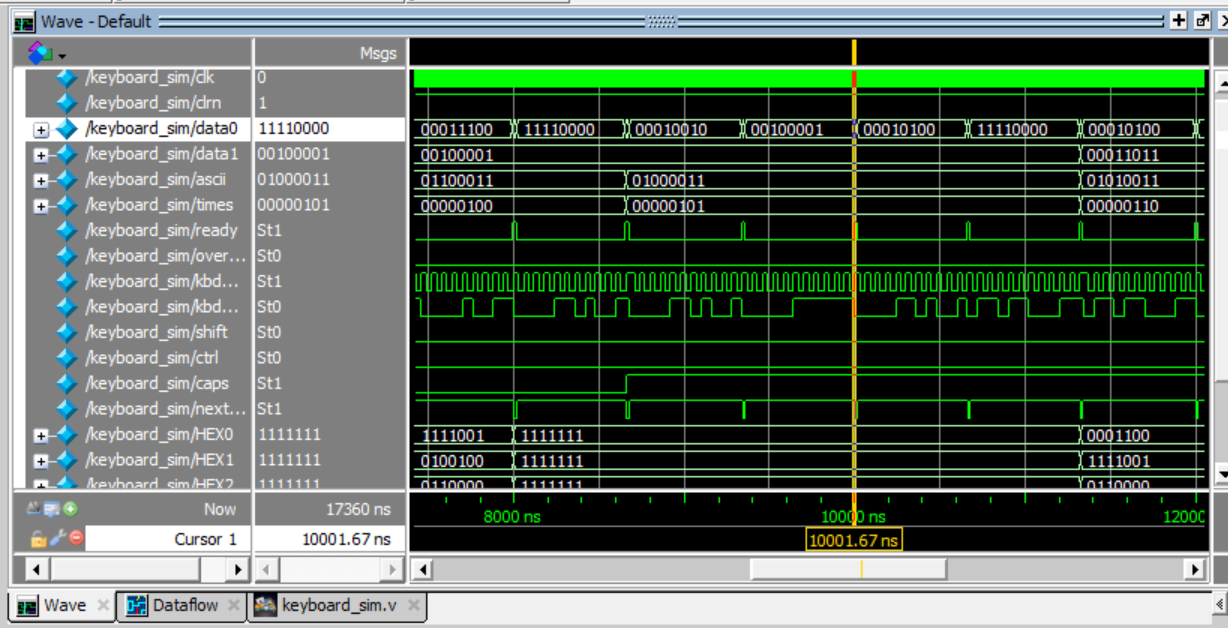
* Shift和A同时按下，HEX4，HEX5表示按键次数，可以看到显示分别在shift被按下和A被按下时变化，此时shift标志有效，HEX0，HEX1显示的是A的扫描码0x1C，HEX2,HEX3显示的是A的ASCII码0x61。

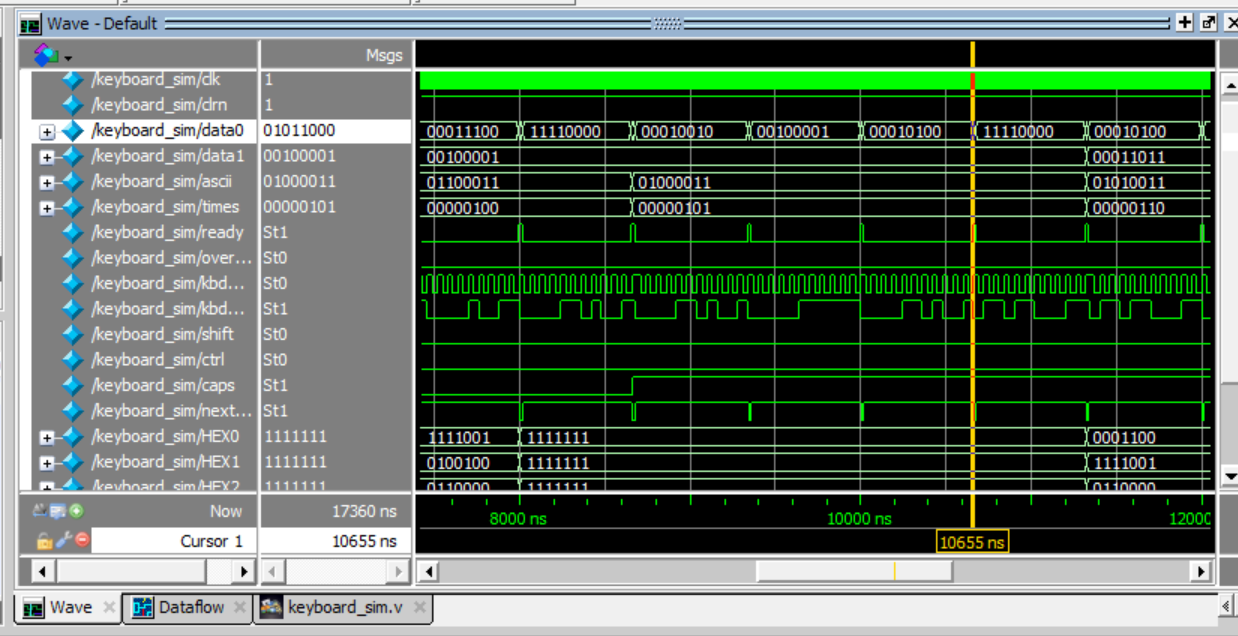
A松开后，低四个数码管都为全1，即全灭，最后shift也松开。



* 下面两张仿真图显示在接到caps断码时的反应（第一张data0:0XF0，第二张 data0:0x58）

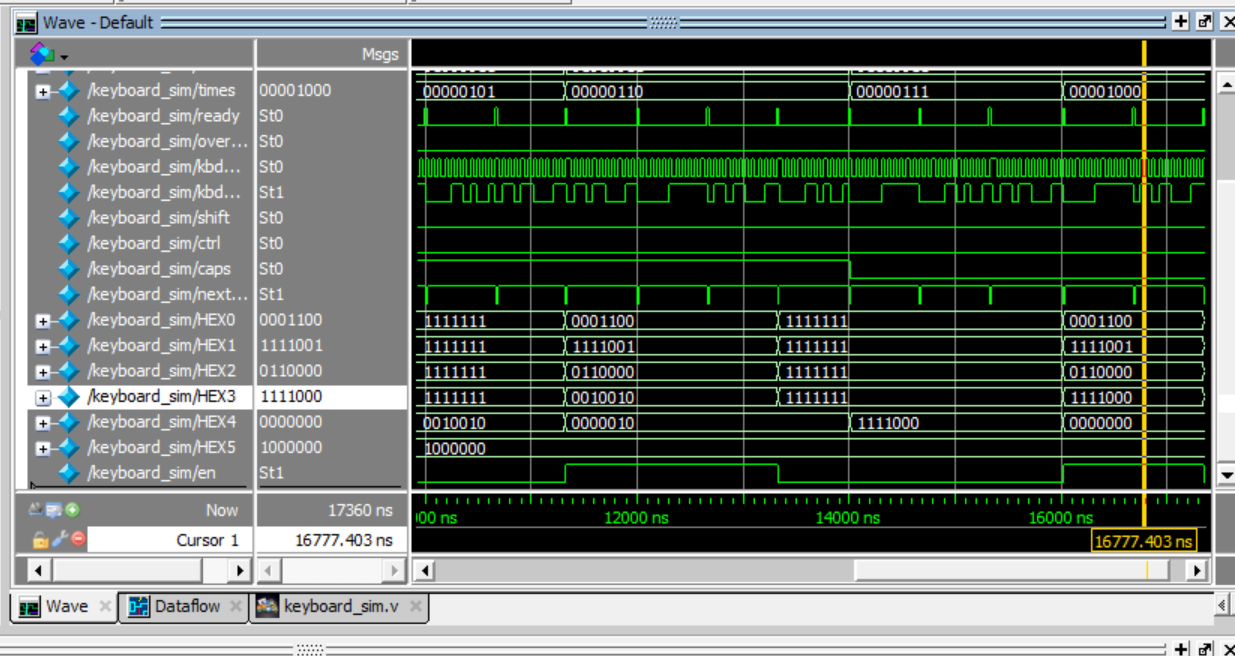
而caps保持不变





* 按下两次S，当caps有效时，HEX3，HEX2代表的ASCII码为0x53（HEX3=7’b0010010）第二张caps无效，HEX3，HEX2代表的ASCII码为0x73(HEX3=7’b111100)，实现了大小写的切换。

两次按下S都是按住不放，可以看到计数器只在第一次按下时计数（HEX4的值变化）

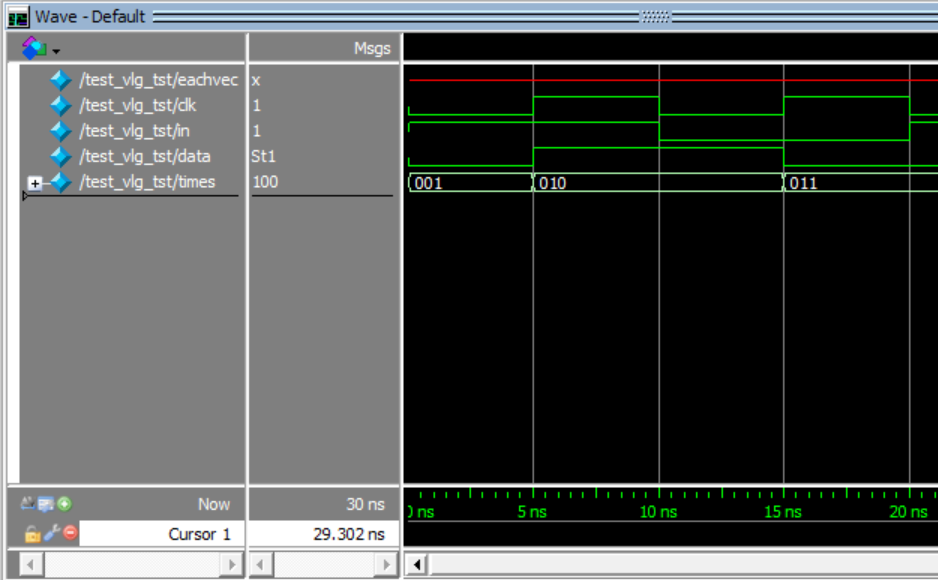


六．实验反思与收获

* 开始划分模块时，计数功能也由显示模块完成，在数据变化时计数，但这样在模拟时发现在开始时就计了一次数，猜测可能是变量初始化也被算作一次数据变化，因此编写了简短的程序测试

测试方法为：data在时钟上升沿接受输入in，每次data变化，times计数（设定为三位）

可以看到在模拟开始times就变为001，此时clk还未变化，in还没有改变data的值，故只可能是赋初值导致了times的变化。



因此后来将计数部分放在键盘数据处理模块，这样模块划分也更加清晰

* 建立了较为完整的数码管显示模块，支持使能端，十六进制和十进制，可以为之后实验复用。
* 自己在.v文件编写的测试模块只要添加好模块名和文件也可像testbench一样测试。
* 了解了在顶层模块和底层模块使用FIFO队列缓存数据的设计方法。