



设计模式期末项目  
Slow6502 小组

陈 晨  
孟 宇

戴仁杰  
杨淳屹

姜文渊  
杨孟臻

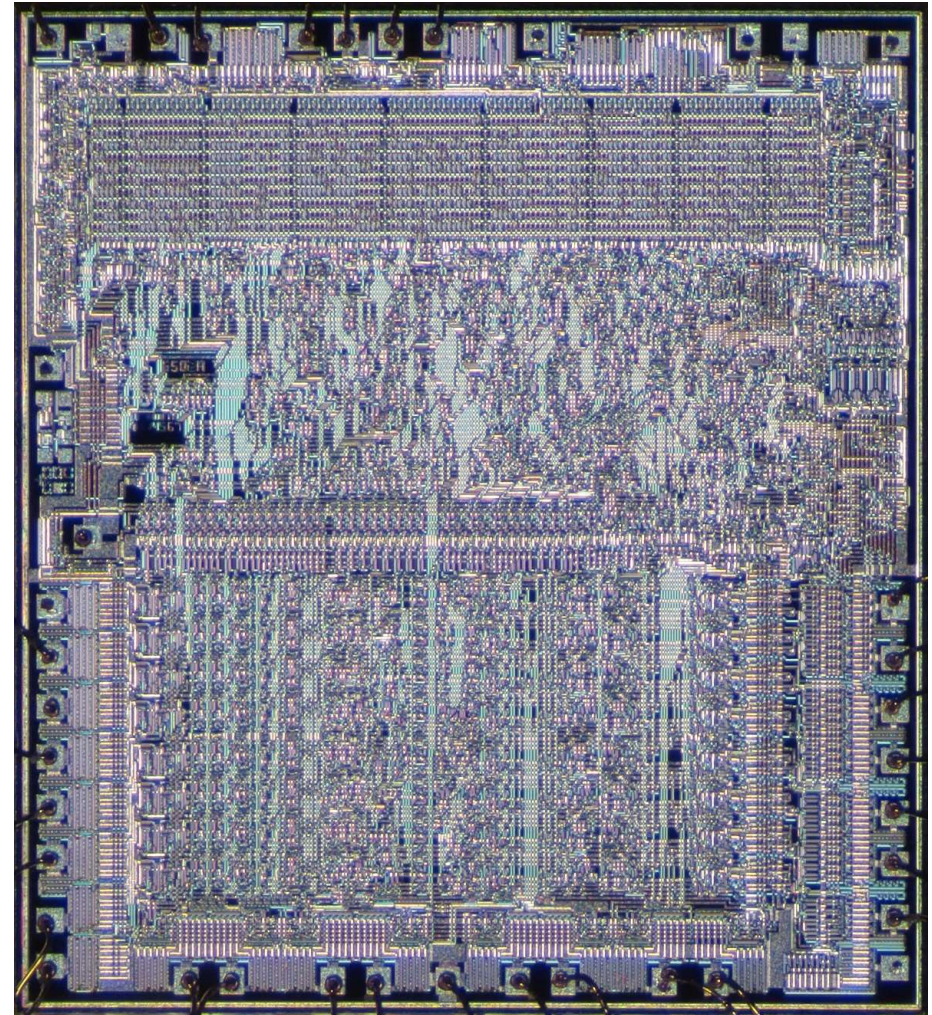
李乐天  
杨 鑫

# Outline

- 项目背景
- 产出概览
- 项目演示
- 为何 用设计模式
- 如何 用设计模式

# 关于 MOS 6502

- 来自 1975 年的成熟技术
  - 仅售 \$ 24.99
- 1.00 MHz 主频
  - 最大睿频 3.00 MHz
- 高达 3,510 个晶体管
  - 数据总线 8 bit, 地址总线 16 bit
  - 最大 64 kB 内存支持
  - 极其先进的精简指令集 (56条指令)
  - 多达 6 个寄存器
- 主流的 40 针 双列直插封装
- 主流外设兼容















# 小霸王® 中英文电脑学习机 SB-926

特别推荐 **认知码** 学习软件

认	知	码	入	门	指	南
认	知	码	词	组	练	习
认	知	码	速	度	练	习
认	知	码	单	字	练	习
认	知	码	综	合	练	习
认	知	码	填	字	游	戏

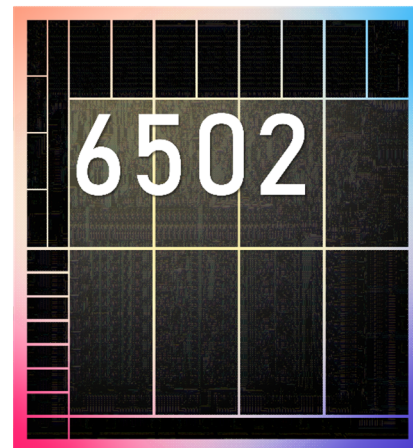


全国中小学计算机教育研究中心

特别推荐

# 产出概览

- 一个 MOS 6502 系统的模拟器
  - 包括 MOS 6502 CPU 和其变种
  - 包括串口终端和CRT在内的多种外设
  - 不同内存布局的支持
- Java 1.8 实现
- 使用了 31 种设计模式
  - 包括全部的 23 种 GoF 设计模式
  - 外加 8 种非 GoF 设计模式
- 友好的图形界面






# 目录

<b>1 项目简介</b>	<b>6</b>
1.1 工作内容简介	6
1.2 选题背景	7
<b>2 设计模式详述（GoF 模式）</b>	<b>9</b>
2.1 单例模式（Singleton）	9
2.1.1 单例模式简介	9
2.1.2 单例模式在项目中的应用	9
2.2 工厂方法模式（Factory Method）	10
2.2.1 工厂方法模式简介	10
2.2.2 工厂方法模式在项目中的应用	11
2.3 抽象工厂模式（Abstract Factory）	11
2.3.1 抽象工厂模式简介	11
2.3.2 抽象工厂模式在项目中的应用	12
2.4 建造者模式（Builder）	12
2.4.1 建造者模式简介	12
2.4.2 建造者模式在项目中的应用	13
2.5 代理模式（Proxy）	13
2.5.1 代理模式简介	13
2.5.2 代理模式在项目中的应用	14
2.6 适配器模式（Adapter）	15
2.6.1 适配器模式简介	15
2.6.2 适配器模式在项目中的应用	15
2.7 桥接模式（Bridge）	15
2.7.1 桥接模式简介	16
2.7.2 桥接模式在项目中的应用	16
2.8 装饰器模式（Decorator）	16
2.8.1 装饰器模式简介	17
2.8.2 装饰器模式在项目中的应用	17
2.9 外观模式（Facade）	18
2.9.1 外观模式简介	18
2.9.2 外观模式在项目中的应用	19
2.10 原型模式（Prototype）	19
2.10.1 原型模式简介	19

2.10.2 原型模式在项目中的应用	20
2.11 享元模式（Flyweight）	20
2.11.1 享元模式简介	21
2.11.2 享元模式在项目中的应用	21
2.12 组合模式（Composite）	21
2.12.1 组合模式简介	21
2.12.2 组合模式在项目中的应用	22
2.13 模板方法模式（TemplateMethod）	23
2.13.1 模板方法模式简介	23
2.13.2 模版方法模式在项目中的应用	24
2.14 策略模式（Strategy）	24
2.14.1 策略模式简介	24
2.14.2 策略模式在项目中的应用	25
2.15 命令模式（Command）	25
2.15.1 命令模式简介	25
2.15.2 命令模式在项目中的应用	26
2.16 责任链模式（Chain of Responsibility）	26
2.16.1 责任链模式简介	26
2.16.2 责任链模式在项目中的应用	27
2.17 状态机模式（State）	27
2.17.1 状态机模式简介	27
2.17.2 状态机模式在项目中的应用	28
2.18 观察者模式（Observer）	29
2.18.1 观察者模式简介	29
2.18.2 观察者模式在项目中的应用	30
2.19 中介者模式（Mediator）	30
2.19.1 中介者模式简介	30
2.19.2 中介者模式在项目中的应用	31
2.20 迭代器模式（Iterator）	31
2.20.1 迭代器模式简介	31
2.20.2 迭代器模式在项目中的应用	32
2.21 访问者模式（Visitor）	32
2.21.1 访问者模式简介	32
2.21.2 访问者模式在项目中的应用	33
2.22 备忘录模式（Memento）	34
2.22.1 备忘录模式简介	34

2.22.2	备忘录模式在项目中的应用	35
2.23	解释器模式 (Interpreter)	35
2.23.1	解释器模式在项目中的应用	36
3	设计模式详述 (非 GoF 模式)	36
3.1	过滤器模式 (Filter)	36
3.2	MVC 模式	37
3.3	MVP 模式	38
3.4	数据访问对象模式	39
3.5	传输对象模式	40
3.6	空对象模式	41
3.7	业务代表模式	42
3.8	前端控制器模式	43



Load Program...

Load ROM...

Preferences...

Quit

Load a program into memory

[illegible][illegible]

Preferences

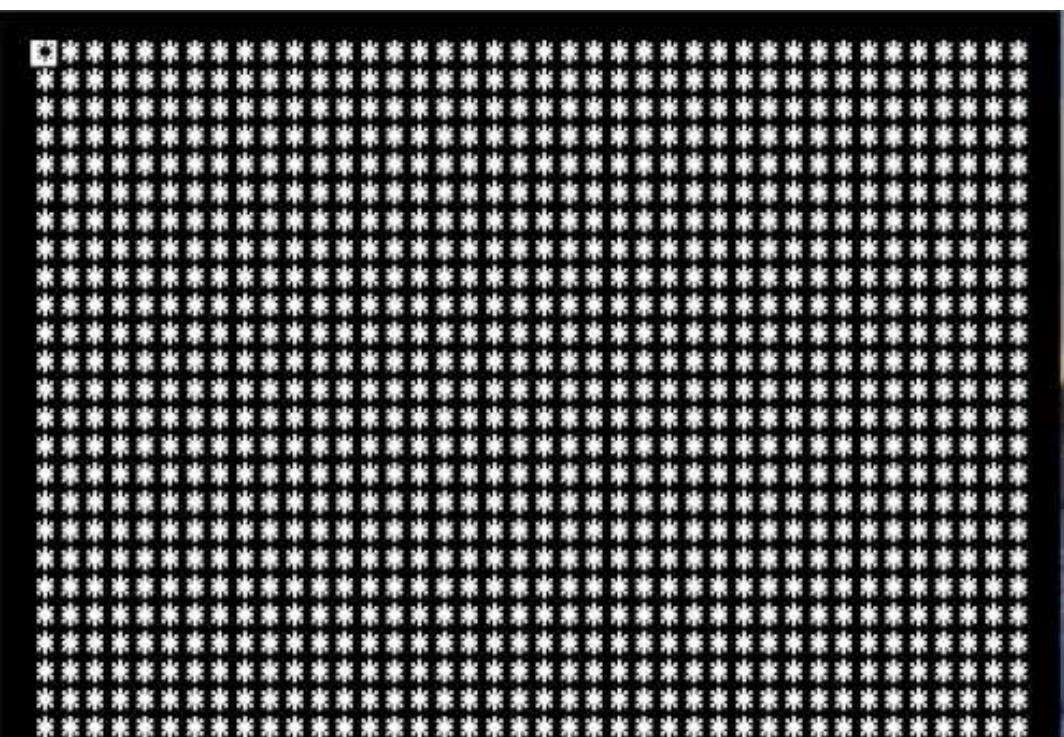
Halt on BRK ☒

Program Load Address 0300

Apply Cancel

Address	Inst
\$0565	BRK
\$7878	BRK






File View Simulator

Enhanced BASIC 2.22

Ready

```
100 VB = 28672
101 H = 25
102 W = 40
103 FOR I = 0 TO H - 1
104   FOR J = 0 TO W - 1
105     X = (J - 20)/10
106     Y = (I - 13)/8
107     T1 = X^2 + Y^2 - 1
108     T2 = X^2 * Y^3
1009   T = T1^3 + T2
1010   CH = 32
1011   IF T < 0 THEN CH = 43
1012   VA = VB + I*W + J
1013   POKE VA, CH
1014   NEXT J
1015   NEXT I
1016 END
```

Ready

Flags: 

Next IR: BEQ \$EA

PC: \$FF50

SP	A
\$F8	\$00

X	Y
\$00	\$09

Stop Step 1 ⏸ Soft Reset Hard Reset

# Why Design Patterns

- 方便写代码
  - 整块的逻辑行数较少
  - if, switch, for 等嵌套更浅
  - 减轻心智负担
- 方便测代码
  - 单元测试粒度更细
  - 集成测试问题定位更精准
- 方便改代码
  - 设计模式带来的灵活性

# How Design Patterns

## 1. 求同存异

- X、Y、A、SP、PC、FLAG 寄存器
- 各类外设

## 2. 化整为零

- CPU 的构成
- 机器和总线的构成

## 3. 统一管理

- 总线
- CPU State
- GUI 和消息传递

## 4. 生搬硬套



# Q&A

Thank you for your attention