

# LINUX 开机过程可视化

制作人：薛雨萌

同组人：无

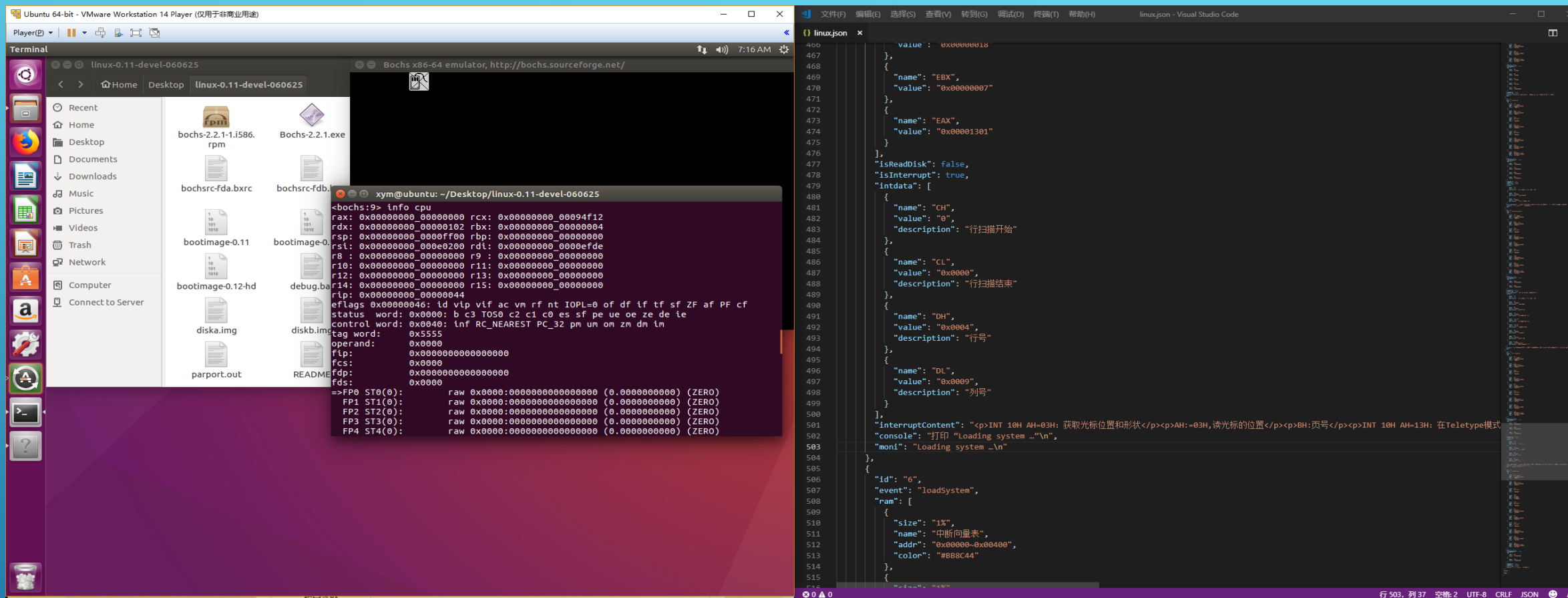
指导老师：杨兴强

- ▶ 阅读了第三章，第四章，第五章，第十章等重点章节，确立了要进行开机部分的可视化
- ▶ 反复仔细阅读bootsect.S setup.S head.S以及main.c，确定要提取的数据，对比去年杨浩然的数据提取，发现有很多做的不够好的地方，在这些地方进行改进。
- ▶ 进入main.c之后主要涉及到各种函数调用，宋振华陈宇翔组的可视化框架可以做这一部份工作，因此主要可视化BIOS执行阶段和另外三个汇编程序执行阶段。
- ▶ 三个文件看似不多，但其中的行为并不具备重复性，也就是说这是一个固定的过程，导致关键帧非常多，每个关键帧都有不小的差别，这导致可视化变得很繁琐

## 阅读代码

- ▶ 因为关键帧多达几十个，而且基本很多相同点差不多，因此我分析了关键帧中哪些没有变化，哪些变化少，哪些变化太大以至于必须做出专门的组件才能可视化
- ▶ 总结了三份分析报告，帮助确定关键帧

## 确立关键帧



# 提取数据

- ▶ 由于进入main.c之前其它的一切方法都不可用，只能使用bochs手动提数据，基本相当于自己一行代码一行代码的看，一行代码一行代码的调试，非常的复杂繁琐，下一级同学参考我的数据就足够了，基本这部分数据都让我提取出来了。
- ▶ 和上一级杨浩然的相比，我使用的都是真实的数据，而不是只是为了展现效果的描述，这部分看我的数据就知道了。

## 提取数据

```
1856 ],
1857 "isCarry": false,
1858 "isChangeRegister": false,
1859 "isReadDisk": false,
1860 "isInterrupt": true,
1861 "interruptContent": "开启分页, 设置了5个页表",
1862 "console": "开启分页, 跳转到 main()",
1863 "moni": ""
1864 }
1865 ]
1866 }
1867 }
1868
```

```
285 }
286
287 {
288   "module": "HEAD",
289   "event": "setpage",
290   "provider": "yhr",
291   "time": 22,
292   "data": {
293     "description": "开启A20地址线"
294   }
295 }
296
```

左图为我的数据行数，右图为去年的

```

    },
    {
      "size": "10x",
      "name": "显示缓冲区",
      "addr": "0x0000-0xc0000",
      "color": "#FA7921"
    },
    {
      "size": "12x",
      "name": "DEVICE",
      "addr": "0xc0000-0xf0000",
      "color": "#ED1E63"
    },
    {
      "size": "13x",
      "name": "ROM BIOS映射区",
      "addr": "0xf0000-0x100000",
      "color": "#673AB7"
    }
  ],
  "iscarry": true,
  "carry": {
    "from": 2,
    "to": 0,
    "times": 256
  },
  "isChangeRegister": true,
  "registers": [
    {
      "name": "DS",
      "value": "0x2000"
    },
    {
      "name": "ES",
      "value": "0x1000"
    },
    {
      "name": "ECX",
      "value": "0x00000000"
    },
    {
      "name": "EAX",
      "value": "0x00001000"
    }
  ],
  "isReadDisk": false,
  "isInterrupt": false,
  "console": "移动 system 模块到 0x00000\n",
  "mon1": ""

```

```

{
  "module": "SETUP",
  "event": "read-systemParameter",
  "provider": "yhr",
  "time": 20,
  "data": {
    "sysy": 300,
    "sysh": 30,
    "description": "将读到的信息保存到这个位置"
  }
}

```

左图为我某个数据的一部分，右图是相同功能的数据的全部内容

- ▶ 相比去年的数据，多了很多新的数据，详细丰富了很多，基本将整个开机阶段的数据完整提取了出来。
- ▶ 数据真实，完全是手动调试保存下来的，不仅包括代码里写好的对寄存器操作的值，更是连中断返回的寄存器的值都包括，中断返回值的意义也有说明。

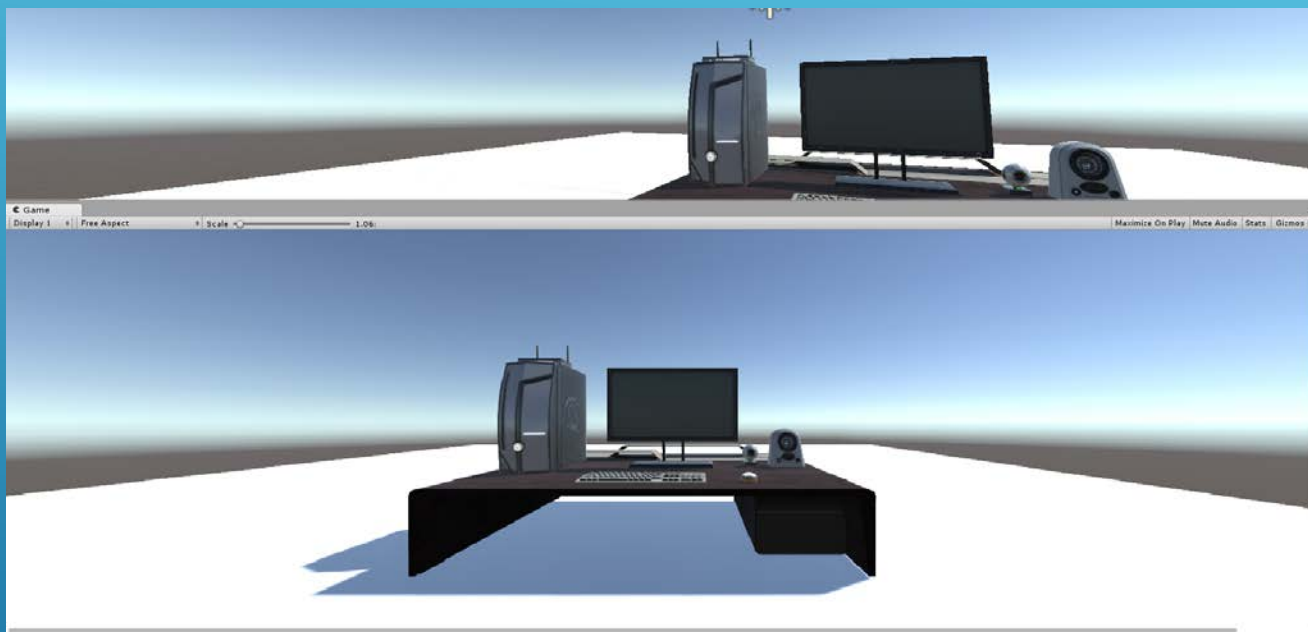
改进



- ▶ 在我看来，可视化不仅是做出能够可视化的东西。
- ▶ 比如做动画，虽然效果很好，但是不可复用，过程是死的，数据稍有变化就不能用了。
- ▶ 因此封装可视化的组件，适应各种数据的变化才是可视化的目的。
- ▶ 我的可视化页面的组件基本都可以复用，比如可视化内存布局的组件，按照我的数据格式写好，不同的数据都可以正常使用，可供下一级使用。

# 可视化

- ▶ 最初我打算用unity3d可视化，进bootsect之前它的效果还是不错的，但发现UGUI各种不变，于是换成了web



可视化（放弃的方案）

▶ 下面开始展示我的最终方案

可视化（最终方案）