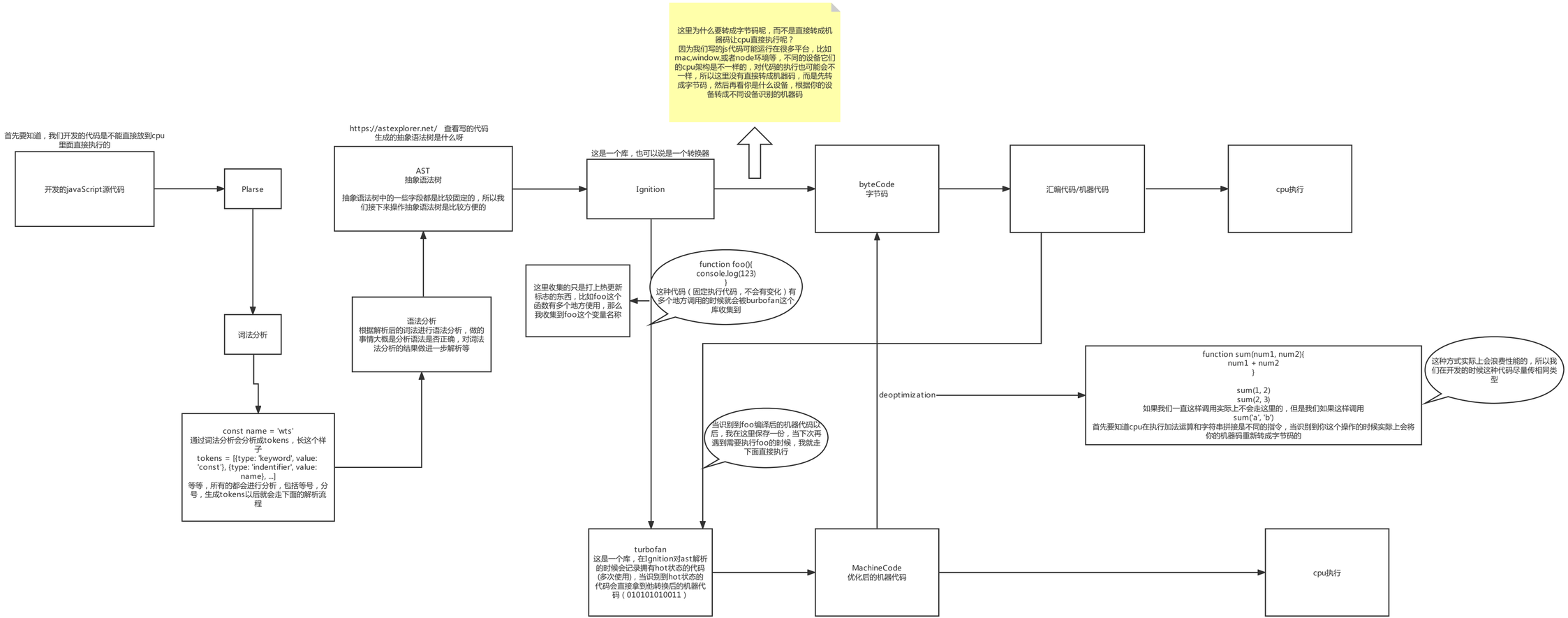
这里为什么要转成字节码呢，而不是直接转成机器码让cpu直接执行呢？
因为我们写的js代码可能运行在很多平台，比如mac,window,或者node环境等，不同的设备它们的cpu架构是不一样的，对代码的执行也可能会不一样，所以这里没有直接转成机器码，而是先转成字节码，然后再看你是什么设备，根据你的设备转成不同设备识别的机器码
首先要知道，我们开发的代码是不能直接放到cpu里面直接执行的
开发的javaScript源代码
Plarse
词法分析
const name = 'wts'
通过词法分析会分析成tokens，长这个样子
tokens = [{type: 'keyword', value: 'const'}, {type: 'indentifier', value: name}, ...]
等等，所有的都会进行分析，包括等号，分号，生成tokens以后就会走下面的解析流程
语法分析
根据解析后的词法进行语法分析，做的事情大概是分析语法是否正确，对词法法分析的结果做进一步解析等
https://astexplorer.net/ 查看写的代码生成的抽象语法树是什么呀
AST
抽象语法树
抽象语法树中的一些字段都是比较固定的，所以我们接下来操作抽象语法树是比较方便的
这是一个库，也可以说是一个转换器
Ignition
这里收集的只是打上热更新标志的东西，比如foo这个函数有多个地方使用，那么我收集到foo这个变量名称
function foo(){
console.log(123)
}
这种代码（固定执行代码，不会有变化）有多个地方调用的时候就会被burbofan这个库收集到
byteCode
字节码
汇编代码/机器代码
cpu执行
deoptimization
function sum(num1, num2){
num1 + num2
}
sum(1, 2)
sum(2, 3)
如果我们一直这样调用实际上不会走这里的，但是我们如果这样调用
sum('a', 'b')
首先要知道cpu在执行加法运算和字符串拼接是不同的指令，当识别到你这个操作的时候实际上会将你的机器码重新转成字节码的
这种方式实际上会浪费性能的，所以我们在开发的时候这种代码尽量传相同类型
当识别到foo编译后的机器代码以后，我在这里保存一份，当下次再遇到需要执行foo的时候，我就走下面直接执行
turbofan
这是一个库，在Ignition对ast解析的时候会记录拥有hot状态的代码（多次使用），当识别到hot状态的代码会直接拿到他转换后的机器代码（010101010011）
MachineCode
优化后的机器代码
cpu执行