基于ZYNQ的FPGA部分实现串口通讯和LC-3架构：可行性分析，任务目标，以及architecture的规划

1. 可行性分析

材料：拥有一块别人友情赞助的ZYNQ开发板，其有一块Xilinx 7000 的arm处理器，一块Artix 7 的FPGA。

由于ARM处理器架构过于复杂，我们这里暂时不使用，只使用FPGA的部分。

我们的目标是实现LC-3架构，其拥有16种指令，2^16位内存。

对于vivado的synthesis过程来说，16种指令开发难度较大，2^16位内存需要大量综合的时间。

我们打算让其拥有2^23位的空间—1MB足够存储一个较大程序，拥有内存，可以实现11种指令。

在与电脑的通讯方面，我们将参考中科大的digital lab代码。

[2.3 LC-3 处理器 - Digital Lab 2024](https://soc.ustc.edu.cn/Digital/2024/lab8/LC_3/)

关于LC-3的编译器

感谢[laser/src/label.c at testing · PaperFanz/laser · GitHub](https://github.com/PaperFanz/laser/blob/testing/src/label.c)开发的laser库使得LC-3转机器码成为可能。具体实现细节我们就按黑箱操作就行，不过此类编译器的基本原理都是刚开始创造一个变量名和地址的mapping，称为symbol table。第二遍再进行assemble，遇到变量就用symbol table上对应的值替换。

1. 任务目标

通过FPGA实现LC-3架构

Input

一个按钮负责切换显示哪个寄存器

一个按钮复位所有寄存器

Output

10条线控制一组（4个）七段数码管，显示当前寄存器的值

四个LED显示当前查看的是哪个寄存器，9个GPR和PC还有MDR和MAR，一共12个寄存器

因此，经过长时间的研究，这个项目的难度已经被降到了最低，接下来的任务和最大难点就是如何简化并实现这个LC-3结构。

1. Architecture的规划

指令集分析：

Operate ADD/AND/NOT

Control BR/JMP/JSR

Data transfer LD/LDR/LEA/ST/STR

接下来应该研究在原本的LC-3架构上我们要保留哪些，删除哪些

开发流程将分为三个部分—我们先实现三种最简单的操作指令，然后再是数据搬运指令，最后是控制指令，符合先易后难的原则。

有一个概念很重要：微指令

微指令序列相当于一个虚拟的状态机，一个指令（如ADD）由多个微指令组成。

微指令储存在ROM中，读取速度非常快（每条都要单周期内完成），由COE文件进行memory的预设。

微指令具有易于扩展的优点，同时其逻辑清楚适合初学者编写。

在我们的设计中，微指令将以此类格式出现：

0-1: alu\_ctrl

2-11: reg\_ctrl

1. xx: mem\_ctrl

xx-xx+2: COND

xx+3-xx+6: next address（j-field）

微指令拥有自己的内存条和地址系统，有一个专门的微指令计数器，保证指令每周期更新一位。

1. field是满足COND后会跳转的位置。

但是对于有些需要等待的COND类型，就有可能把J-field改为其他的地址入口。如内存等待微指令（R）就令地址不变，使得程序一直等待直到内存更新完毕。