

华中科技大学

电子技术课程设计[报告]

双通道数字示波器

院 系 人工智能与自动化学院

专业班级 自动化 1705 班

姓 名 裴熙隆

学 号 U201714286

指导教师 王贞炎

2020 年 11 月 1 日

摘 要

本双通道数字示波器以 FPGA 为核心，实现了双通道波形的采集与显示。通过高速 ADC 实时采样以获得较高的测量带宽，被测周期信号的频率范围为 10Hz~10MHz。同时系统还具有参数测量、光标测量、档位切换、储存与调出波形等功能。本系统人机交互友好，刷屏帧率高，功能全面，性能指标优良，很好地完成了课程设计任务。

关键词： FPGA；电子测量；数字示波器

目 录

摘要	I
1 系统方案设计	1
1.1 系统方案描述	1
1.2 方案比较和选择	1
1.2.1 采样逻辑设计方案	1
1.2.2 信号调理方案	1
1.2.3 主控选择方案	2
1.3 系统框图	2
2 相关原理和细节设计	3
2.1 ADC 选型及驱动	3
2.1.1 ADC 选型	3
2.1.2 ADC 驱动	3
2.2 FPGA 选型	4
2.3 采样触发控制	4
2.4 等精度频率测量	6
2.5 数据流接口设计	7
2.6 DMA 设计	7
2.7 液晶显示屏与人机交互	8
2.8 视频流直接存储器访问 (Vedio DMA)	10
2.9 时序控制器 (Video Timing Controller)	12
2.10 LCD 背光控制	12
2.11 波形数据存储	13
3 电路设计	14
3.1 信号调理电路	14
3.2 高速 ADC 采样电路	14
3.3 电源设计	15
4 程序设计	16
4.1 PS-PL 协同设计概述	16

4.2 嵌入式软件开发	16
4.3 FPGA 逻辑开发	17
4.3.1 FIFO 模块	17
4.3.2 直接存储器访问 (DMA) 模块	17
4.3.3 ZYNQ7 Processing System	18
4.3.4 液晶显示器驱动模块	18
5 系统仿真与调试	23
5.1 FPGA 开发流程	23
5.1.1 开发流程重要步骤说明与图示	23
5.1.1.1 新建工程	23
5.1.1.2 新建文件	23
5.1.1.3 功能仿真	24
5.1.1.4 引脚约束	24
5.1.1.5 时序约束	25
5.1.1.6 综合 RTL 电路图	27
5.1.2 综合实现报告与分析	27
5.1.2.1 时序报告分析	28
5.1.2.2 功耗分析	28
5.1.2.3 资源分析	29
5.1.2.4 电路实现版图	29
5.2 FPGA 调试	30
5.2.1 ILA 测试简介	30
5.2.1.1 Vivado ILA 简介	30
5.2.1.2 Vivado ILA 的使用	32
5.2.2 ILA 调试分析	32
5.2.2.1 ADC、采样、触发系统、FIFO、DMA 综合调试 .	32
5.2.3 液晶显示屏调试	32
5.2.4 人机交互界面调试	33
5.2.5 任务调度调试	33
5.3 系统功能测试	33

6 测试方案与测试结果	35
6.1 测试环境	35
6.2 测试方案	35
6.2.1 方波校准信号测试方案	35
6.2.2 单次触发扫描测试方案	35
6.2.3 扫描速度测试方案	35
6.2.4 垂直灵敏度测试方案	35
6.3 测试结果与数据	36
6.3.1 方波校准信号测试	36
6.3.2 单次触发扫描测试	36
6.3.3 扫描速度测试	36
6.3.4 垂直灵敏度测试	37
6.4 测试结果分析	37
7 问题分析与总结	38
7.1 问题分析	38
7.1.1 数字示波器硬件问题	38
7.1.2 数字示波器软件问题	38
7.2 心得感悟	39
8 参考文献	40
附录 A 【2007 国赛 C 题】数字示波器	41
附录 B 平台说明及主要元器件清单	43

1 系统方案设计

1.1 系统方案描述

本设计实现的数字示波器，以 Xilinx 的 Zynq-7020 系列 FPGA 为核心，利用搭载的 ARM Cortex-A9 处理器作为软件设计基础，实现了对被测信号的采样、显示、参数测量、光标测量、档位切换、触发功能切换、储存与调出等功能，通过高速 ADC 实时采样以获得较高的测量带宽。系统主要由信号调理、取样与保持、触发与控制逻辑四个部分组成，PL 部分使用 100MHz 时钟驱动；可测量周期信号的频率范围为 10Hz~10MHz，周期与电压测量误差均 $\leq 5\%$ ；0.5V/div、0.2V/div、0.1V/div 三个档位均能准确测量、显示波形；上升沿或下降沿触发方式、触发电平均可调节且验证无误；提供水平、垂直方向各两个测量光标，以及自动测量信号的频率、峰峰值、有效值、平均值、光标差值等参数，测量误差均 $\leq 1\%$ ；当前显示的波形数据可以存储在板载的 eMMC 存储器中，随时调用。人机交互友好，刷屏帧率高，功能全面，所有指标均达到设计要求。

1.2 方案比较和选择

1.2.1 采样逻辑设计方案

方案一：FPGA 逻辑根据当前选择的扫描档位来切换采样方式。

方案二：FPGA 逻辑根据当前需要的采样率来切换采样方式。

方案选择：方案一，逻辑简单，易于实现，可以符合题目指标，但可扩展性不强；方案二，逻辑较复杂，但可扩展性强，可设置较多的采样率，从而设置更多的扫描档位。综合考虑，使用方案二。

1.2.2 信号调理方案

方案一：使用一路放大器将各个档位的信号放大到合适值即可。

方案二：设置多档放大，分别设置不同的增益。

方案选择：方案一，部分档位输入信号过大，部分档位输入信号过小，不利于 AD 采集；方案二，电路复杂，但各个档位均能放大至 AD 输入最大范围，高

效利用 AD 的动态范围，提高了系统的精度。综合考虑，使用方案二。

1.2.3 主控选择方案

方案一：FPGA 负责底层逻辑，用 MCU 做主控制器，它们之间通过 SPI 等串行通信接口通信。

方案二：FPGA 负责底层逻辑，用软核处理器做主控制器。

方案选择：方案一，人机交互界面设计简单，但是 FPGA 与 MCU 之间数据通信不够可靠，通信速率成为了系统的瓶颈；方案二，软核与底层逻辑数据通信通过 AXI 总线，稳定可靠。虽然人机交互界面设计较为复杂，但是依然可以通过很多方式实现。综合考虑，使用方案二。

1.3 系统框图

系统框图如图 1-1 所示，两个通道的输入信号经过信号调理和换挡电路后由 AD9288 采集，通过并行数据总线输入 FPGA，PL 部分最终将数据通过 DMA 存储在 DDR 中，供 PS 部分进行读取。PS 部分完成采样逻辑的控制和波形显示等 人机交互部分，通过 VDMA 驱动显示器显示。

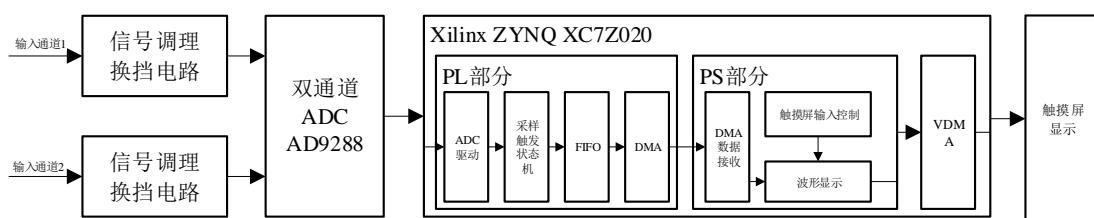


图 1-1 系统框图

2 相关原理和细节设计

2.1 ADC 选型及驱动

2.1.1 ADC 选型

题目要求最高水平速度为 1us/div, 共 10 个 div, 计算可得采样率要达到 10MHz。垂直分辨率最高共 100 个点, 因此需选用至少 7 位的 ADC。将功能拓展为 0.1us/div, 并且双通道采样, 则最高采样率为 100MHz。综合考量, 选用 ADI 公司的 AD9288 作为采样 ADC, 此 ADC 为双通道 8 位并行 ADC, 符合垂直分辨率的要求。

2.1.2 ADC 驱动

ADC 驱动负责产生 AD9288 的采样时钟、控制逻辑以及接收数据。

AD9288 为双通道 8 位并行 ADC, 最高可达 100MHz 的采样速率。示波器采用最高采样率, 因此直接输出 FPGA 的系统时钟 100MHz 到 ADC 上作为采样时钟。由于输入端为时钟下降沿接收数据, 所以输出时钟端口先经过反相器输入, 输入信号则可用 100MHz 系统时钟上升沿触发接收数据, 直接得到两个通道的采样数据, 数据格式为偏移二进制, 因此还要将 MSB 取反, 得到补码形式的数据。

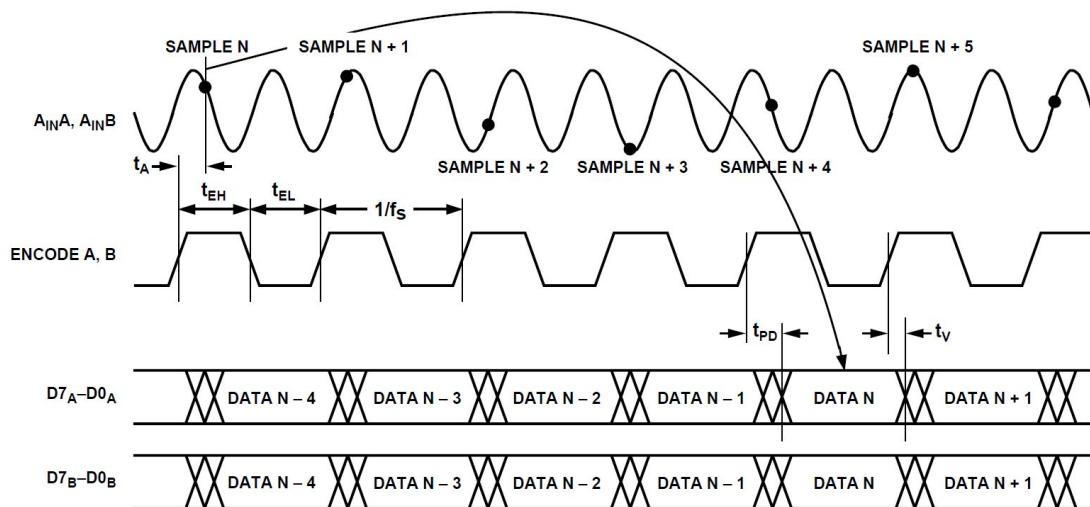


图 2-1 AD9288 采样时序图

为避免输入的亚稳态效应，ADC 并行数据进入 FPGA 后要先经过两级触发器，再送给下一级进行处理。AD9288 的采样时序图如图 2-1 所示。

2.2 FPGA 选型

FPGA 采用 Xilinx ZYNQ 系列的 XC7Z020 作为主控，其框图如图 2-2 所示。该 FPGA 具有两个 ARM Cortex-A9 高性能应用级 ARM 处理器，PL 部分有 53.2k 的 LUT，106.4k 的触发器，并且有 1GB DDR3 SDRAM，4GB eMMC，满足题目使用要求。

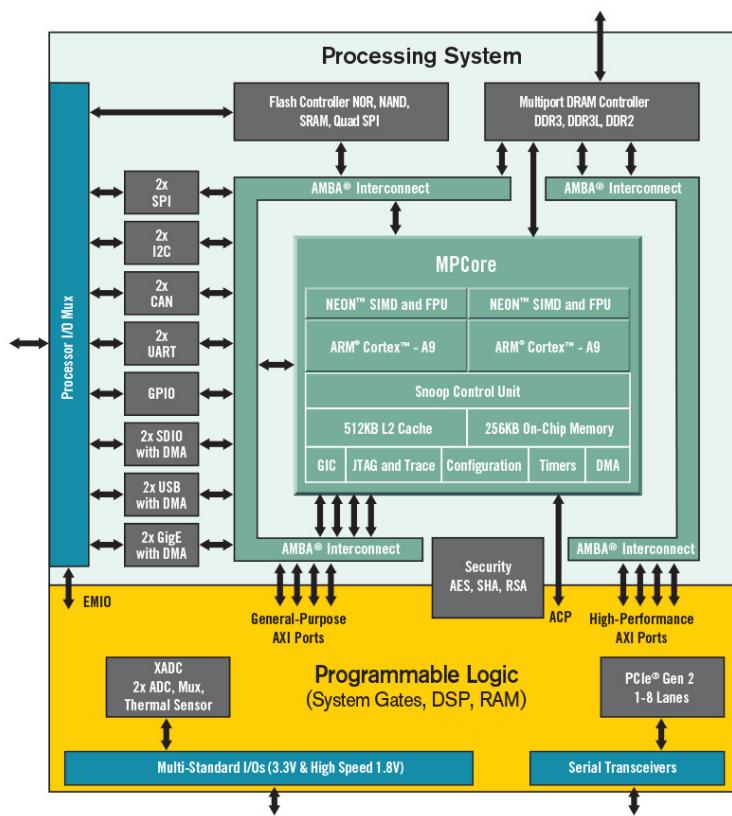


图 2-2 Zynq-7000 SoC 框图

2.3 采样触发控制

示波器采样与触发系统负责对 ADC 的信号进行采样触发控制。

触发系统同时还有水平档位切换功能，上升/下降沿触发，通道 1/2 触发控制，触发电平控制功能。采样系统一般持续接收数据。数据量超过设定的一帧数量，丢弃最早的数据，存入最新的数据，而在触发事件发生年后，在持续接收一

定量的数据后停止。

水平档位从小到大分别为 $0.1\mu s/div$, $0.2\mu s/div$, $0.5\mu s/div$, $1\mu s/div$, $2.5\mu s/div$, $5\mu s/div$, $10\mu s/div$, $25\mu s/div$, $50\mu s/div$, 使用一个计数器产生使能信号控制速率, 在 $100MHz$ 的系统时钟下, 计数器的模分别为 $0, 1, 4, 9, 24, 49, 99, 249, 499$, 将使能信号送到各个触发器块中, 模的值通过处理器送来的寄存器值进行设置, 从而实现在处理器中控制水平速率。

上升/下降沿触发, 触发通道切换通过处理器中送来的寄存器值进行修改, 并带有 10 个单位的迟滞触发, 以抵抗噪声带来的误触发。

采样触发系统具有预触发功能, 使用状态机对各个状态的切换进行控制。触发采样系统可以抽象出以下六种状态:

- (1) 空闲状态 (IDLE): 不采集。
- (2) 触发前采集状态 (PRE): 确保能采满用户设定的需要在触发事件到来前保留的数据量, 在此状态下还需要对采集的数据计数, 以便达到数量转换状态。
- (3) 等待触发状态 (WAIT): 等待触发事件的到来, 同时也在采集数据, 在此状态下也需要对已采集的数据计数, 以便达到超时时间时转换状态。
- (4) 触发状态: 满足用户设定的触发条件, 开始触发后采样。
- (5) 超时状态: 等待出发状态经历 $0.5s$, 强制采集一帧数据。
- (6) 后采样状态: 满足触发条件或者已经超时, 将数据采集满后结束采集。

采样触发系统的状态转移如图 2-3 所示。

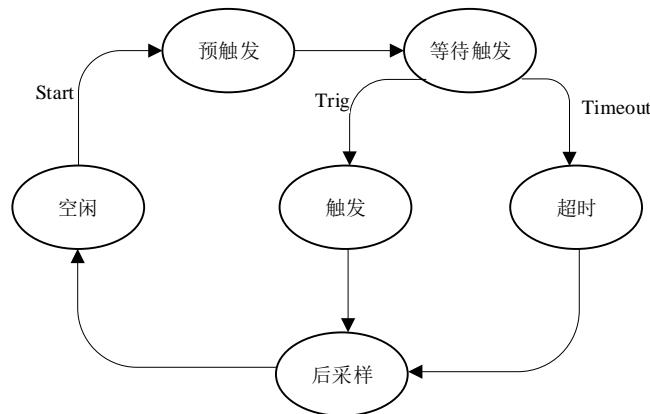


图 2-3 状态转移图

送给 FIFO 的数据共 32 位, 其数据格式为:

[7:0]: 通道 1 数据;

[15:8]: 通道 2 数据;

[16]: 触发点处为 1, 其他为 0;

[17]: 是否属于超时采样, 是的话为 1, 否则为 0;

[31:18]: 保留。

2.4 等精度频率测量

频率测量功能为了保证精确以及高性能, 采用 PL 部分使用等精度频率计的技术进行测量。

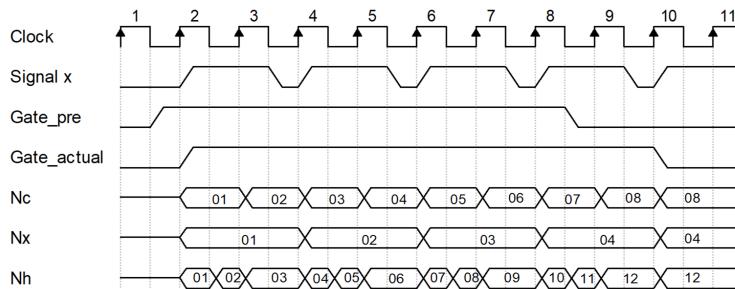


图 2-4 等精度频率测量原理

如图 2-4 所示, f_x 为待测信号频率, f_c 为标准时钟频率。两个计数器在同一闸门时间 T 内分别对待测信号、标准时钟进行计数, 计数值为 N_x 、 N_c 。由于 $\frac{N_x}{f_x} = \frac{N_c}{f_c} = T$, 则被测频率 $f_x = \frac{N_x}{N_c} \cdot f_c$ 。

利用 D 触发器使计数闸门信号与被测信号同步, 开门时间 T 等于被测信号周期的整数倍, 故 N_x 无量化误差。则测频误差:

$$\frac{\Delta f_x}{f_x} = \pm \left(\frac{T_c}{T} + \left| \frac{\Delta f_c}{f_c} \right| + \frac{0.32}{k} \times 10^{-\frac{R}{20}} \right) \quad (2.1)$$

式中 T_c 为 N_c 计数器在同步门 T 期间的 ± 1 量化误差, $\frac{\Delta f_c}{f_c}$ 为时钟 f_c 的不确定度, R 为待测信号的信噪比, k 为多周期倍率。

主要误差由 T_c 引起, 故 $\frac{\Delta f_x}{f_x} = \pm \frac{T_c}{T}$ 。时钟频率 200MHz 时, 则 $T_c = 50\text{ns}$, 闸门选为 0.2s, 则 N_c 计数误差 $\frac{T_c}{T}$ 为 5×10^{-9} 。实测可测得的频率范围为 DC~20MHz, 测量精度大于十万分之一。

将测得的数据通过 AXI Lite 总线将寄存器数据送到处理器中进行频率计算并显示。

2.5 数据流接口设计

经过采样触发控制系统后，得到的数据要通过一个 FIFO (First Input First Output) 储存器，即先进先出的队列，送给 DMA，从而与处理器进行数据同步。FIFO 存储器是一个先入先出的双口缓冲器，即第一个进入其内的数据第一个被移出，其中一个是存储器的输入口，另一个口是存储器的输出口。使用 Xilinx 的 IP 的 FIFO Generator 实现，将接口格式设置为 AXI Stream。

由于示波器具有预触发功能，在数据采样后 FIFO 数据量大于采样长度时要将 FIFO 中的数据读出，同时要将 FIFO 的读接口接给 DMA，即 FIFO 的 Read 端口连接两个部分，因此需要做逻辑控制，将 FIFO 的 Read 端口做切换，在示波器采样时将读端口送给示波器状态机做控制，在空闲时将读端口切换给 DMA 读取。其框图如图 2-5 所示。

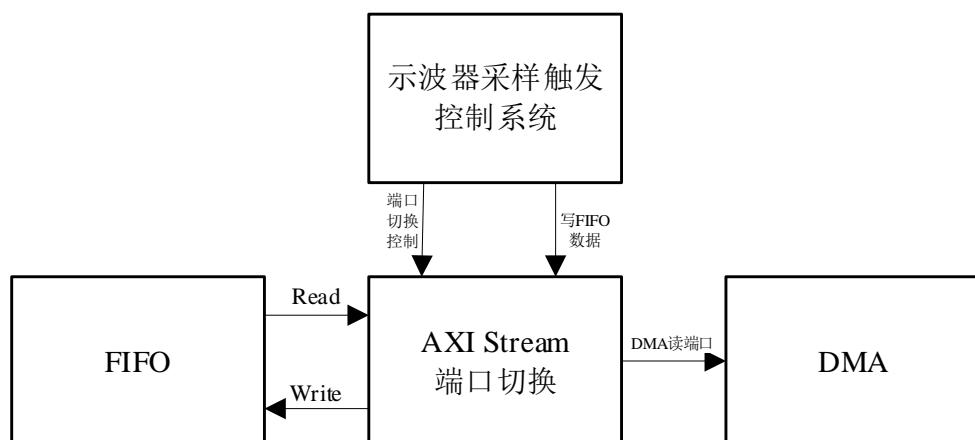


图 2-5 AXI Stream 接口框图

2.6 DMA 设计

为满足数据的高速读取操作，FIFO 与处理器之间需使用 DMA 将 FIFO 的数据转移到内存 DDR 中供处理器使用。

DMA (Direct Memory Access, 直接内存存取) 传输将数据从一个地址空间复制到另外一个地址空间。当 CPU 初始化这个传输动作，传输动作本身是由 DMA

控制器来实行和完成。典型的例子就是移动一个外部内存的区块到芯片内部更快的内存区。像是这样的操作并没有让处理器工作拖延，反而可以节省时间去处理其他的工作。

使用 Xilinx 提供了 IP 来实现 DMA，与 FIFO 通过 AXI Stream 接口读取数据，与处理器通过 AXI SmartConnect 连接到 PS 部分的 AXI HP 端口，将数据转移到内存中。FIFO 与 DMA 接口的时序图如图 2-6 所示。

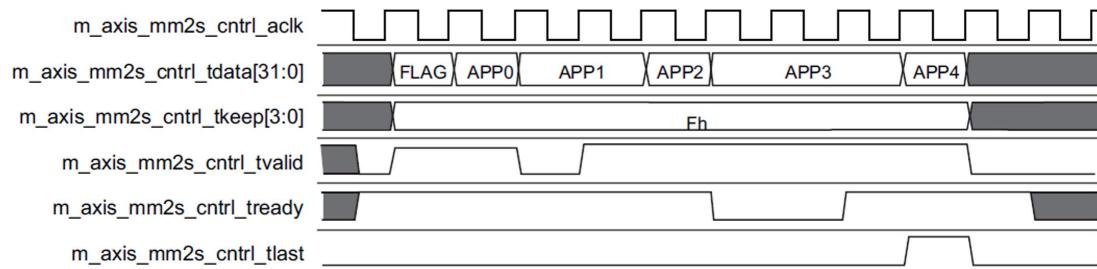


图 2-6 FIFO 与 DMA 的 AXI Stream 接口时序

2.7 液晶显示屏与人机交互

为了更好地实现人机交互，选用微雪电子 7 寸液晶显示屏进行显示，分辨率为 1024×600 ，可以很好地显示当前采样得到的波形，且自带轮询触摸检测，检测到触摸后，由屏幕内部控制芯片在 I2C 接口上产生对应的信息，传送到 FPGA 板上，再由顶层触发中断，解析数据，得到触摸点坐标并进行相应的操作。图 2-7 所示为显示屏的基本信息。



图 2-7 LCD 显示屏基本信息

触摸屏中断信号 INT 为低电平有效。当手指触摸电容屏时，INT 信号会以固定频率（默认为 60Hz）脉冲信号的形式输出，当手指离开电容屏，INT 信号

重新恢复高电平，如图 2-8 所示，本质上看就是一种 IIC 的传输方式，我们正是利用了这个信号，来控制触摸。

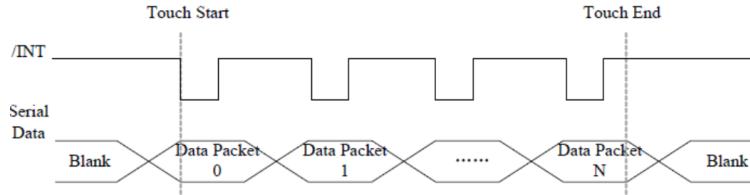


图 2-8 触摸屏中断时序图

得到触摸信号之后，根据 I2C 传来的串行寄存器数据来读取信息，查阅显示屏的数据手册，可以得到图 2-9 所示的寄存器表。

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access		
Op.00h	DEVIDE_MODE	Device Mode[2:0]										
Op.01h	GEST_ID	Gesture ID[7:0]										
Op.02h	TD_STATUS						Number of touch points[3:0]					
Op.03h	TOUCH1_XH	1 st Event Flag		1 st Touch X Position[11:8]								
Op.04h	TOUCH1_XL	1 st Touch X Position[7:0]										
Op.05h	TOUCH1_YH	1 st Touch ID[3:0]			1 st Touch Y Position[11:8]							
Op.06h	TOUCH1_YL	1 st Touch Y Position[7:0]										
Op.07h												
Op.08h												
Op.09h	TOUCH2_XH	2 nd Event Flag		2 nd Touch X Position[11:8]								
Op.0Ah	TOUCH2_XL	2 nd touch X Position[7:0]										
Op.0Bh	TOUCH2_YH	2 nd Touch ID[3:0]			2 nd Touch Y Position[11:8]							
Op.0Ch	TOUCH2_YL	2 nd Touch Y Position[7:0]										
Op.0Dh												
Op.0Eh												
Op.0Fh	TOUCH3_XH	3 rd Event Flag		3 rd Touch X Position[11:8]								
Op.10h	TOUCH3_XL	3 rd Touch X Position[7:0]										
Op.11h	TOUCH3_YH	3 rd Touch ID[3:0]			3 rd Touch Y Position[11:8]							
Op.12h	TOUCH3_YL	3 rd Touch Y Position[7:0]										
Op.13h												
Op.14h												
Op.15h	TOUCH4_XH	4 th Event Flag		4 th Touch X Position[11:8]								
Op.16h	TOUCH4_XL	4 th Touch X Position[7:0]										
Op.17h	TOUCH4_YH	4 th Touch ID[3:0]			4 th Touch Y Position[11:8]							
Op.18h	TOUCH4_YL	4 th Touch Y Position[7:0]										
Op.19h												
Op.1Ah												
Op.1Bh	TOUCH5_XH	5 th Event Flag		5 th Touch X Position[11:8]								
Op.1Ch	TOUCH5_XL	5 th Touch X Position[7:0]										
Op.1Dh	TOUCH5_YH	5 th Touch ID[3:0]			5 th Touch Y Position[11:8]							

图 2-9 LCD 寄存器表

其中，TD_STATUS 寄存器记录了触摸点数，因此接收到中断标志之后，首先读取寄存器地址，再读取数据，整个接口时序符合 I2C 标准。

2.8 视频流直接存储器访问 (Vedio DMA)

由于使用 VDMA 可以方便地实现双缓冲和多缓冲机制，所以本小节引入了帧缓存和缓冲机制的概念。另外，VDMA 可以很好地契合 Zynq 内部架构，缩短开发周期。再加上 VDMA 本身能够高效地实现数据存取，所以在基于 Zynq（也包括其他 Xilinx FPGA）的图像、视频处理系统中，VDMA 可谓是必不可少的。

而 VDMA 涉及到缓存的概念。缓冲存储器 (Frame Buffer): 简称帧缓存或显存，它是屏幕所显示画面的一个直接映象，又称为位映射图 (Bit Map) 或光栅。帧缓存的每一存储单元对应屏幕上的一个像素，整个帧缓存对应一帧图像。在开发者看来，Frame Buffer 是一块显示缓存，往显示缓存中写入特定格式的数据就意味着向屏幕输出内容。所以说 Frame Buffer 就是一块画布，系统在画布上绘制好画面之后，就可以通知显示设备读取 Frame Buffer 进行显示了。

在具体的项目实现中，对 VDMA 的调用经常采用双缓冲机制来加速显示以及提高帧率，双缓冲机制的原理图如图 2-10 所示。

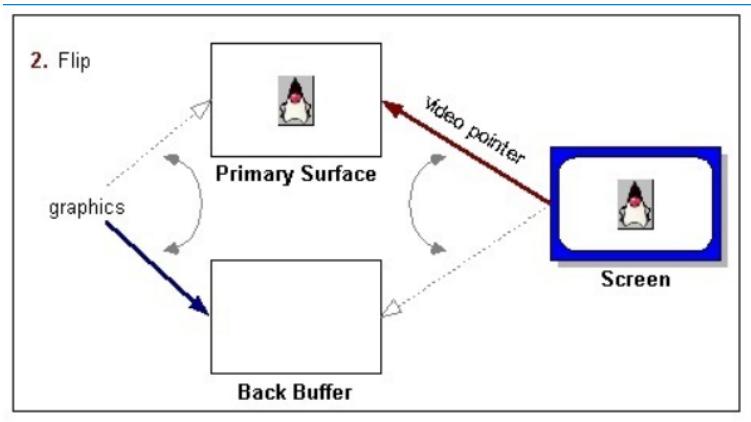


图 2-10 双缓冲示意图

最早解释多缓冲区如何工作的方式，是通过一个现实生活中的实例来解释的，这里花一些篇幅加以描述。

单缓冲：在一个阳光明媚的日子，你想将水池里的水打满，而又找不到水管的时候，就只能用手边的木桶来灌满水池。水桶满了之后，关掉水龙头，将水提到水池旁边，倒进去，然后走回到水龙头。重复上述工作，如此往复直到将水池灌满。这就类似单缓冲工作过程，当你想将木桶里的水倒出的时候，你必须关掉水龙头。

双缓冲：现在假设有人用两个木桶来做上面的工作。你会注满第一个木桶然

后将第二个木桶换到水龙头下面，这样，在第二个水桶注满的时间内，你就可以将第一个木桶里面的水倒进水池里面，当你回来的时候，你只需要再将第一个木桶换下第二个注满水木桶，当第一个木桶开始注水的时候你就将第二个木桶里面的水倒进水池里面。重复这个过程直到水池被注满。

很容易看得到用这种技术注满水池将会更快，同时也节省了很多等待木桶被注满的时间，而这段时间里你什么也做不了，而水龙头也就不用等待从木桶被注满到你回来的这段时间了。

当你雇佣另外一个人来搬运一个被注满的木桶时，这就有点类似于三个缓冲区的工作原理。如果将搬运木桶的时间很长，你可以用更多的木桶，雇佣更多的人，这样水龙头就会一直开着注满木桶了。

在计算机图形学中，双缓冲是一种画图技术，使用这种技术可以使得画图没有（至少是减少）闪烁、撕裂等不良效果，并减少等待时间。双缓冲机制的原理大概是：所有画图操作将它们画图的结果保存在一块系统内存区域中，这块区域通常被称作“后缓冲区（Back Buffer）”，当所有的绘图操作结束之后，将整块区域复制到显示内存中，这个复制操作通常要跟显示器的光栈束同步，以避免撕裂。双缓冲机制必须要求有比单缓冲更多的显示内存和 CPU 消耗时间，因为“后缓冲区”需要显示内存，而复制操作和等待同步需要 CPU 时间。

Xilinx 提供的 VDMA IP 核的几个重要信号，在这里列举。

Frame Buffers：选择 VDMA 缓存几帧图像，这里默认是写通道和读通道都设置相同的缓存帧数，具体设置多少帧合适一般根据应用来定，比如读写带宽相同，想用 DDR 作为一个乒乓 Buffer，那就可以设置成 2 帧，写第一个地址，读第二个地址，然后写第二个地址，读第一个地址。这里面设置几帧，就要在 VDMA 寄存器配置的时候设置几个帧起始地址。

Memory Map Data Width：代表数据到达 AXI4 总线上的位宽，比如这里设置成 64，那就代表 M_AXI_XX 总线上的数据位宽是 64bit，这时候如果 Stream 上的数据是 32bit，那 VDMA 内部会有一个带宽转换模块，把数据拼成 64bit。

Burst Size：AXI 总线上突发传输的长度，一般设置为 16。

Stream Data Width：VDMA 与 PL 逻辑部分通过 AXI Stream 协议交互数据，这里代表 Stream 数据位宽。

Line Buffer Depth：VDMA 内部会有一个行缓存 FIFO，Stream 数据会先写

入 FIFO，然后 AXI 总线逻辑会读出到总线上，这个深度就代表 FIFO 的深度。设置原则（个人理解）：如果 AXI 总线数据带宽是 Stream 总线数据带宽的 1.5 倍以上，这个 FIFO 深度可以设置的小一点，如果 AXI 总线带宽小于 1.5 倍的 Stream 总线带宽，那 FIFO 的深度至少要是图像一个有效行的一半。

2.9 时序控制器 (Video Timing Controller)

使用 VDMA 来传送显示的数据，还需要有正确的时钟信号来控制 VDMA 从 DDR 内存里搬移数据，以及转化成 RGB 接口输出。这一工作是由时序控制器 VTC 来完成的，Xilinx 官方有提供时序控制器的 IP 核，为节约开发时间，直接调用。

所有视频系统都需要管理视频定时信号，该视频定时信号用于同步各种过程。视频定时控制器具有检测和生成这些定时信号的功能。图 2-11 显示了包括定时信号的典型视频帧。

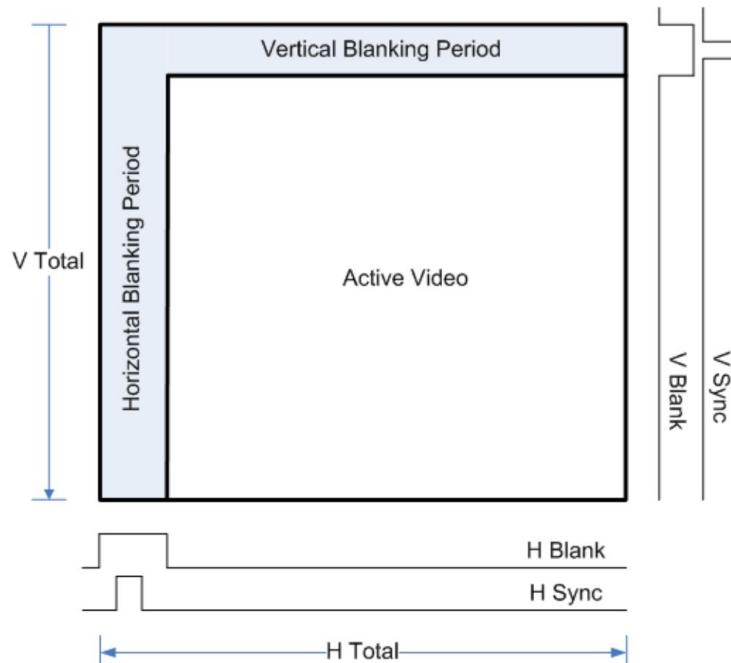


图 2-11 视频帧和定时信号示例

2.10 LCD 背光控制

LCD 显示屏需要用到背光控制，背光采用脉冲宽度调制进行控制。因此需要设计 PWM 模块生成脉冲宽度调制信号，输出给显示器。

为了减小开发周期，我们采用 Digilent 公司的 AXI PWM IP 核来生成脉冲宽度调制信号。另外，此 IP 核可以在软件中随时配置 PWM 信号的参数（如频率、占空比等等），使用较为灵活。

2.11 波形数据存储

为了存储示波器的当前波形，我们比较了 SD 卡和 eMMC 两种存储方式，最终决定使用 eMMC 来存储波形数据，原因在于 eMMC 拥有较快的读写速度，且配置起来较为方便，减少开发周期和调试难度。

嵌入式多媒体卡 (Embedded Multimedia Card) 是一种新的存储技术，由 MMC 协会所订立。该架构标准将 MMC 组件（闪存加控制器）放入一个球栅数组封装（BGA）中，是一种主要用于印刷电路板的嵌入式非易失性存储器系统。eMMC 有 100、153、169 个触点之分，并都基于 8 位并行接口。eMMC 与 MMC 的其他版本有明显不同，因为 eMMC 不是用户可随意移动的卡，而是永久性的电路板附件。如果 eMMC 出现内存或其控制器的问题，则可能需要更换整个 PCB（印刷电路板）来修复。

几乎所有在 2016 年之前生产的手机和平板电脑都使用这种形式的主存储器，直到 2016 年 UFS 开始渐渐占领市场。JEDEC 的最新的 eMMC 标准 (JESD84-B51) 是在 2015 年 2 月发布的 5.1 版本，该版本的 eMMC 速度可媲美 SATA 接口标准的固态硬盘 (400 MB/s)，表 2-1 简述了不同版本的 eMMC 的写入速度和读取速度。

表 2-1 不同版本的 eMMC 的写入速度和读取速度

	写入速度 (MB/s)	读取速度 (MB/s)
eMMC 5.1	125	250
eMMC 5.0	90	250
eMMC 4.5	50	140

由此看来，可以说 eMMC 的读写速度远超 SD 卡，综上所述，选用 eMMC 读写方案。由于 FPGA 开发板已经具备板载的 eMMC，只需要配置存储器接口即可使用，在 PS 配置中勾选 SD 卡接口，将其连接到 eMMC，即可实现复用。

3 电路设计

3.1 信号调理电路

信号调理电路主要由挡位切换电路以及抗混叠滤波电路组成。其中，挡位切换电路选用 TI 公司的精密运放 OPA656 搭建交流耦合电压跟随器，其后跟三路不同放大倍数的电阻分压电路，通道 A、B、C 分别对应放大 1 倍、0.5 倍、0.2 倍，电路图如图 3-1 所示。

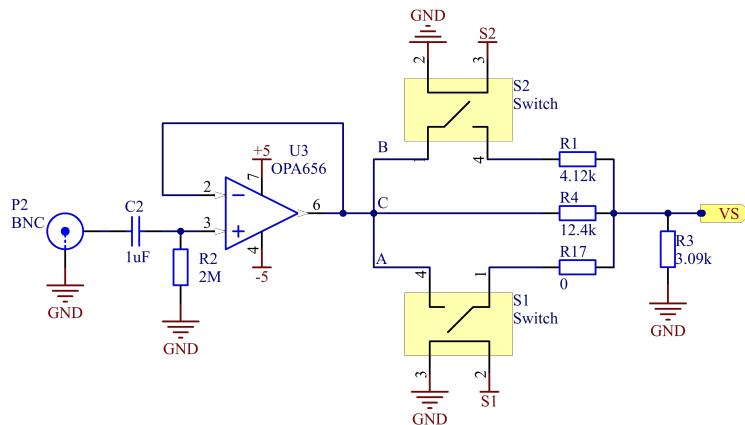


图 3-1 挡位切换电路

抗混叠滤波电路选用五阶无源椭圆滤波器进行抗混叠滤波，使 50MHz 时信号衰减达到 50dB，同时保证 20MHz 的有效信号输入范围。

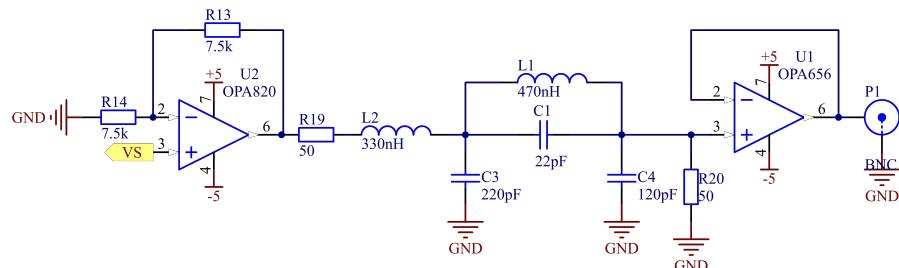


图 3-2 抗混叠滤波电路

3.2 高速 ADC 采样电路

选用 ADI 公司的 8 位双路高速模数转换器 AD9288，其采样率最大可达 100MSPS，可以满足题目要求。选用 ADI 公司的高速运算放大器 AD8132 将单端输入信号转成差分信号。

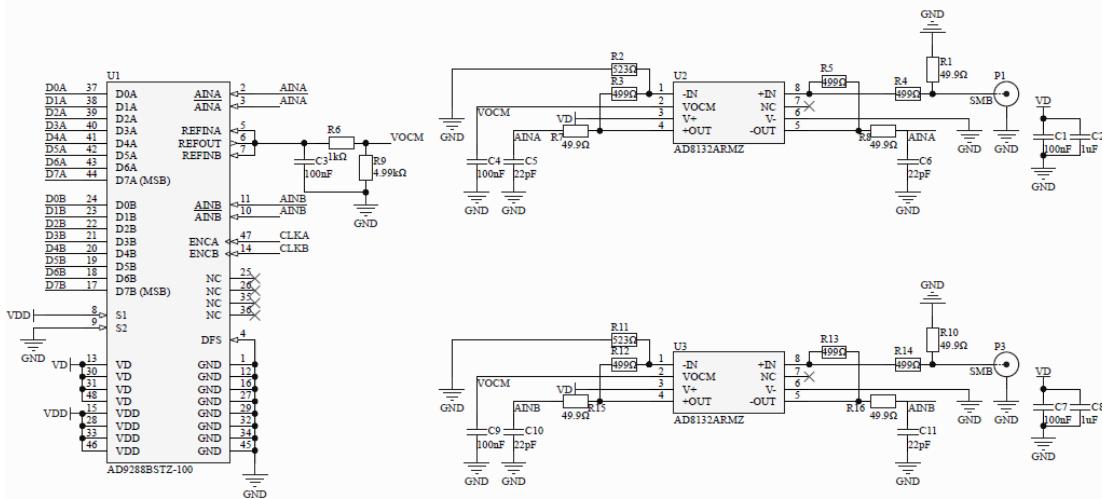


图 3-3 ADC 采样电路

3.3 电源设计

本系统使用了高精度高速 ADC 作为前级采样，要求电源有较高的稳定性，前级信号调理电路也对电源要求颇高。故设计了如图 3-4 所示的 LDO 稳压电路，得到纹波较小的 $\pm 5V$ 电压。经测试，本电源模块的纹波小于 20mV。

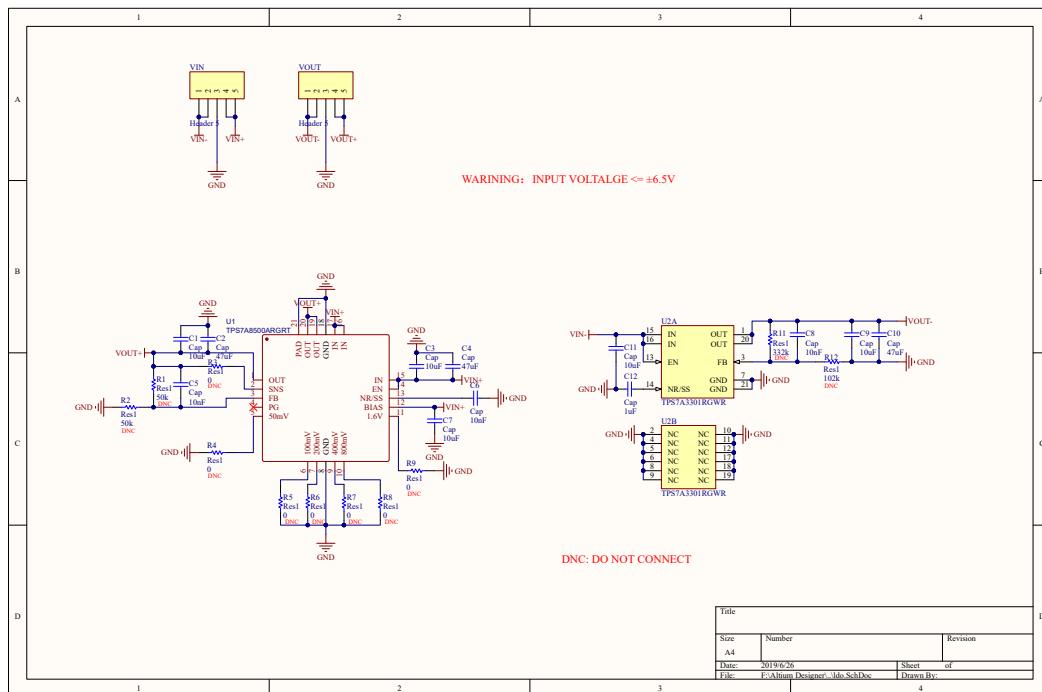


图 3-4 LDO 电源模块电路

4 程序设计

4.1 PS-PL 协同设计概述

本设计采用微雪 7 寸液晶显示屏来进行人机交互，该显示屏带有触摸功能，在检测到触摸时会触发 I2C 接口产生相应的信息，包含触摸点坐标等。因此，需要设计人机交互以及数据交互，故引入 Zynq 系列软硬件协同开发方案，即 PS(Process System)-PL(Programmable Logic) 交互，底层 PL 部分负责采样系统控制，顶层 PS 负责运行时调度、与 PL 数据交互以及与用户人机交互。

4.2 嵌入式软件开发

为了更方便地进行嵌入式软件部分开发，本设计移植了一套精美的嵌入式图形库 Littlevgl。这套嵌入式图形库不仅利于开发嵌入式 UI，还自带实时操作系统 RTOS，即使不用嵌入式操作系统（即裸机开发），也可以很好地达到任务调度的目的。

除此之外，该图形库还具有完善的 API，方便进行调用，同时也有基于 Eclipse 开发的 UI 仿真环境，便于调试。图图 4-1 为本设计的软件程序框图。

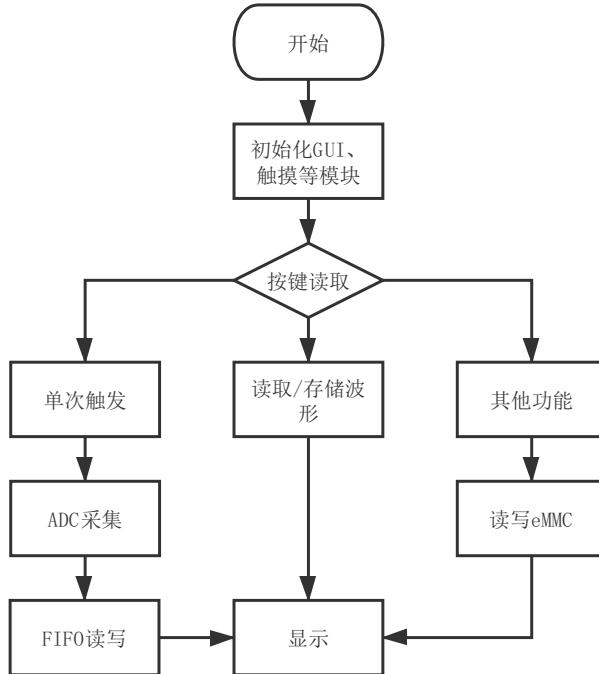


图 4-1 软件程序框图

4.3 FPGA 逻辑开发

如图 4-2 所示即为本设计的软硬件协同开发模块互联图 (Block Design)，将示波器采样控制模块、PS 部分、FIFO/DMA 数据交互模块、VDMA/VTC 显示控制部分全部整合到一起，实现了示波器采样控制、数据流控制、人机交互等功能整合开发。

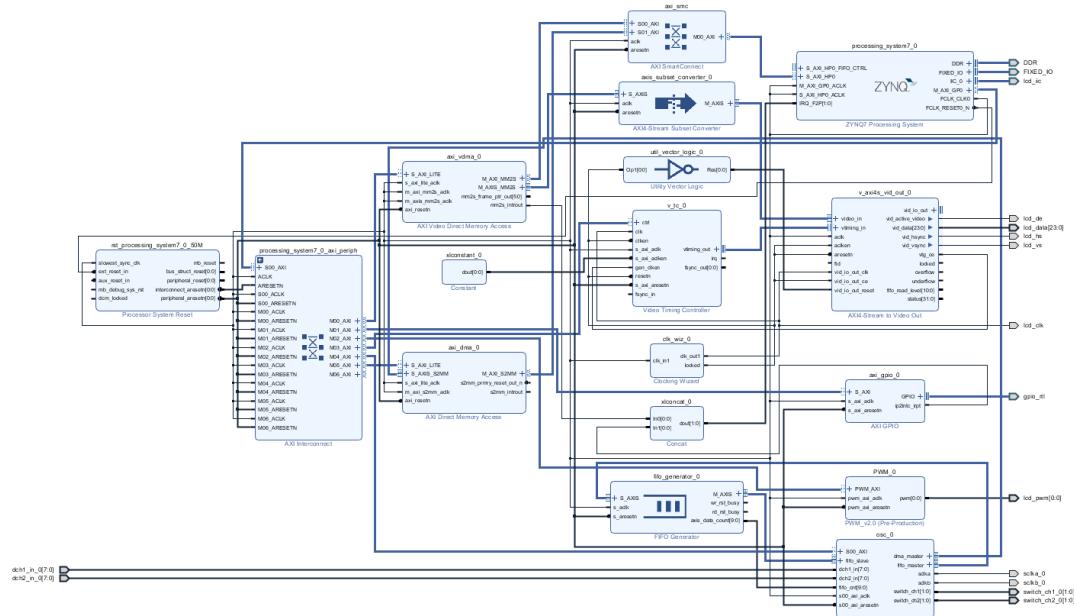


图 4-2 Block Design

下面分别介绍该作品使用到的 PL 部分模块，主要有 FIFO 模块、直接存储器访问 (DMA) 模块、液晶显示器驱动模块。

4.3.1 FIFO 模块

FIFO 模块使用 Xilinx 的 IP 进行配置，将数据宽度设置为 4 个字节，数据深度设置为 512，提供余量以保证不会溢出，并且打开 FIFO 中已经存入的数据量计数接口，提供给触发采样状态机，满足状态转换进行的条件。相关配置如图 4-3 所示。

4.3.2 直接存储器访问 (DMA) 模块

DMA 使用 Xilinx 提供的 IP 进行配置，只需要其中的 Write Channel 即可。相关配置如图 4-4 所示。

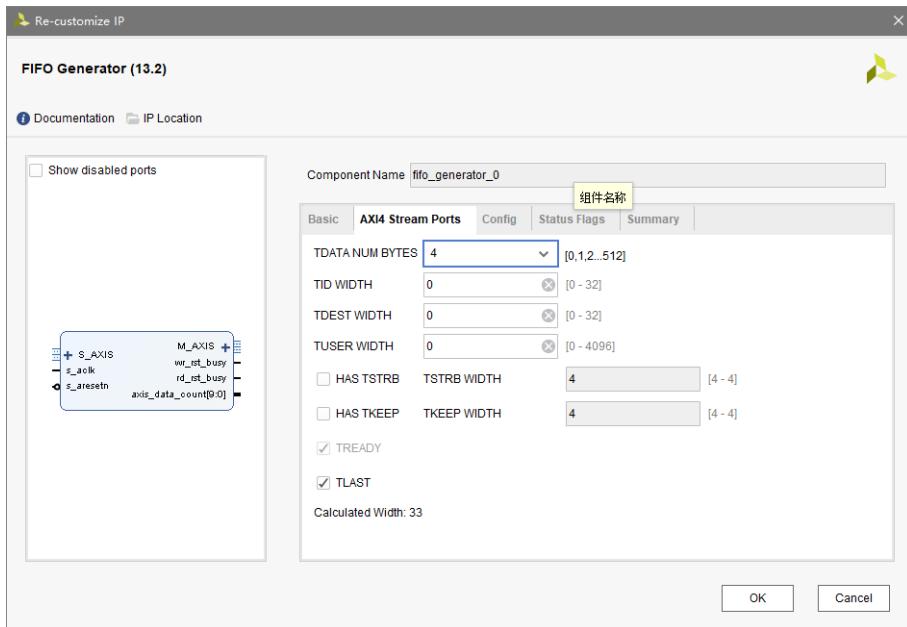


图 4-3 FIFO 配置

4.3.3 ZYNQ7 Processing System

软硬件协同设计的核心在于 PS 硬核处理器，这也是 Xilinx 公司的 Zynq 系列器件优势所在，能够比较方便地配置 ARM Cortex-A9 处理器，在 Block Design 中添加 ZYNQ7 Processing System，再对其进行配置即可，图 4-5 和图 4-6 分别截取了内存以及时钟的配置。

4.3.4 液晶显示器驱动模块

本系统使用了微雪公司的 7 寸液晶显示屏来显示界面，采用 RGB 并行数据传输的方式，通过视频流直接存储器访问（VDMA）模块来完成时序驱动。为了产生与数据流相匹配的时钟，我们采用了 Xilinx 公司的 Video Timing Control IP 核，与 VDMA 模块适配，完成场同步、帧同步等时序操作。该部分模块如图 4-7 所示。

打开 VDMA 的配置界面，如图 4-8 所示，根据原理分析中的显示器时序图，我们可以配置出 VDMA 的参数，分辨率自定义，设置为我们使用的屏幕的 1024×600 分辨率。

打开 Clocking Wizard 的配置界面，如图 4-9 所示，同理可得配置参数。注意要和 VDMA 的配置保持一致，否则将出现时序错误，显示器的驱动时钟为 51.2MHz，所以这里要做相应的配置。

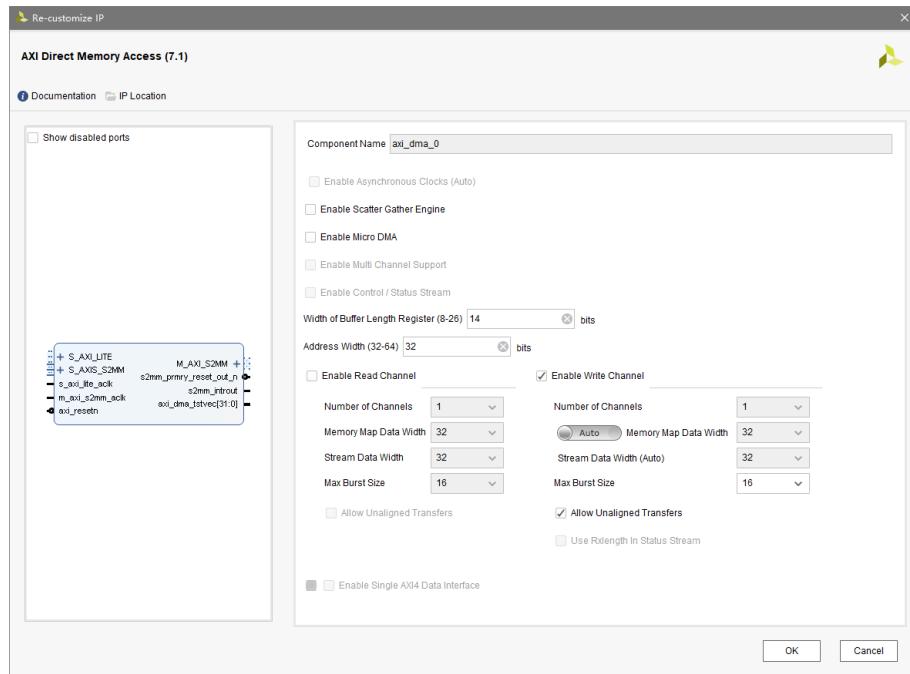


图 4-4 DMA 配置

最后，在连线时，VDMA 和 VTC 之间需要做位宽桥接，调用 Xilinx 的 IP 核来完成位宽转换。在连线时，需要把锁相环的 Lock 信号连接到各个显示屏驱动相关 IP 核的使能接口上，保证在锁相环产生了稳定的时钟之后，再触发传输。

如此便完成了液晶显示器驱动模块的设计，最后将配置好的 RGB 接口引出即可。整个显示器驱动模块的架构如图 4-10 所示。

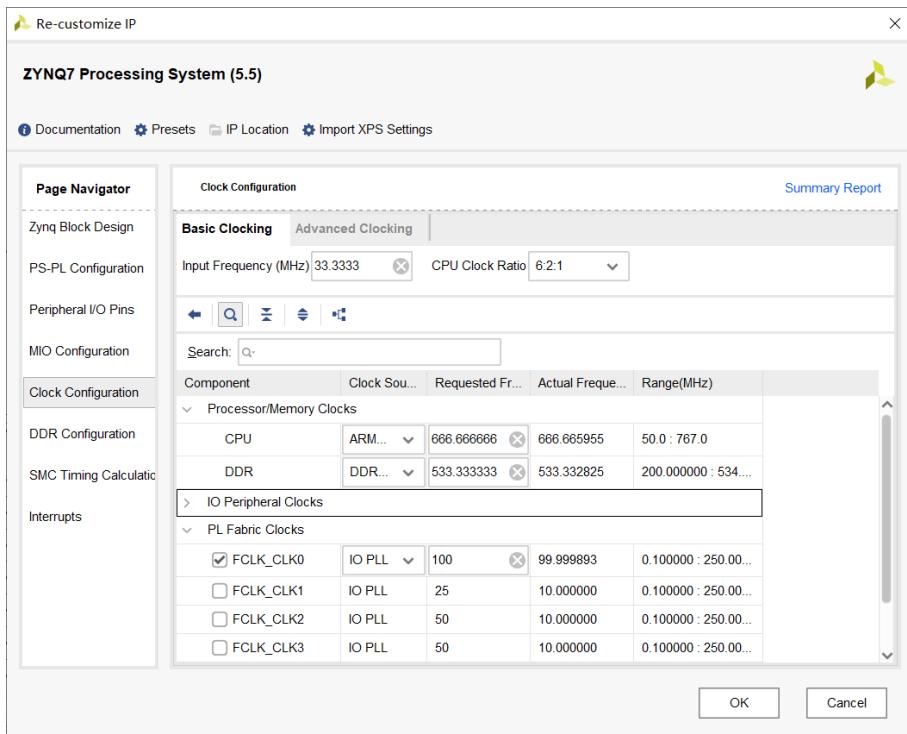


图 4-5 PS 时钟配置

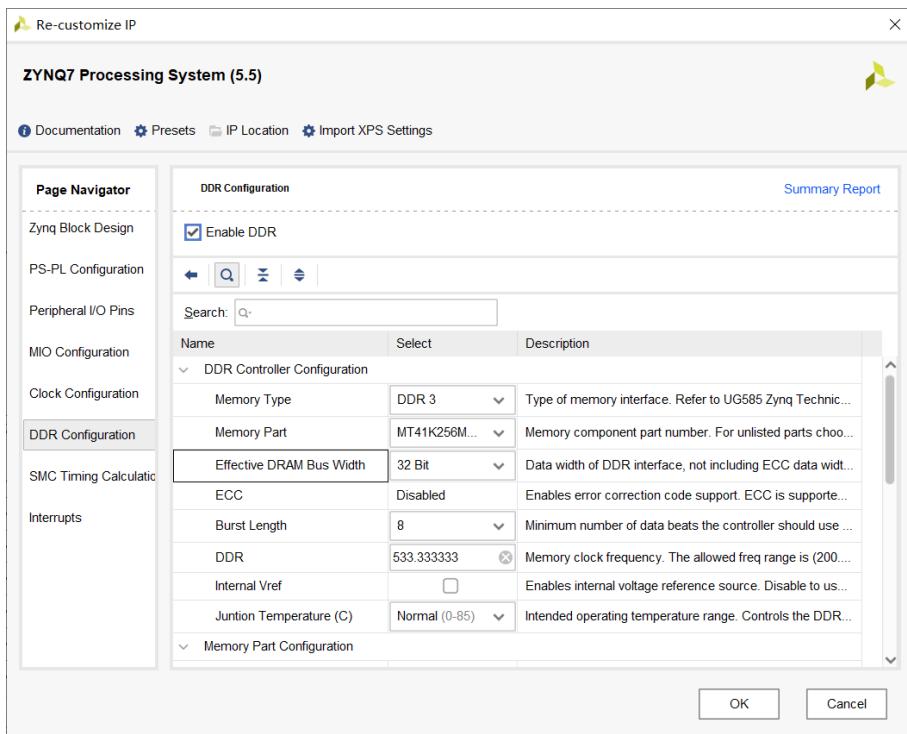


图 4-6 PS 内存配置

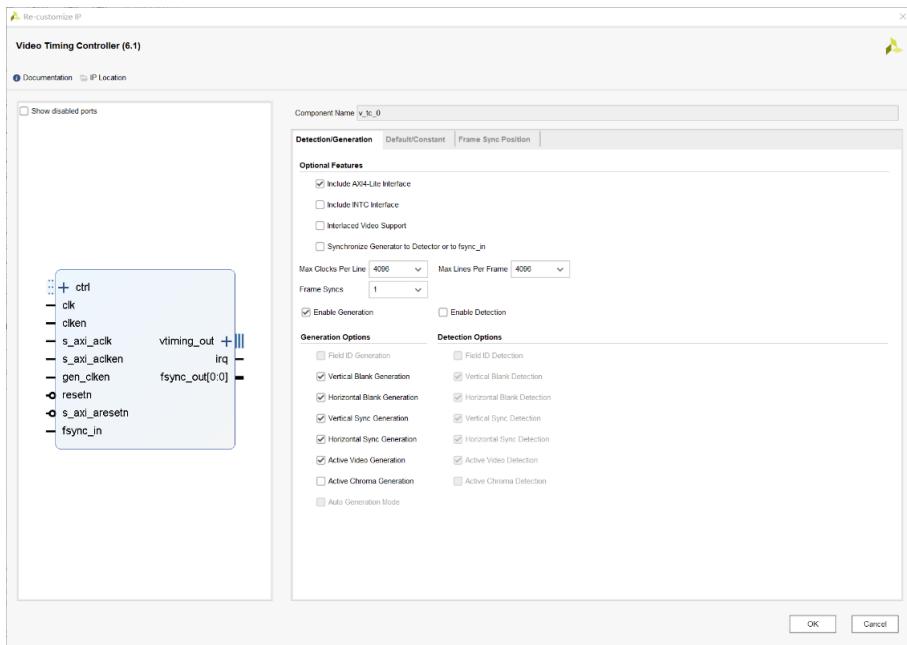


图 4-7 Video Timing Control IP 核配置

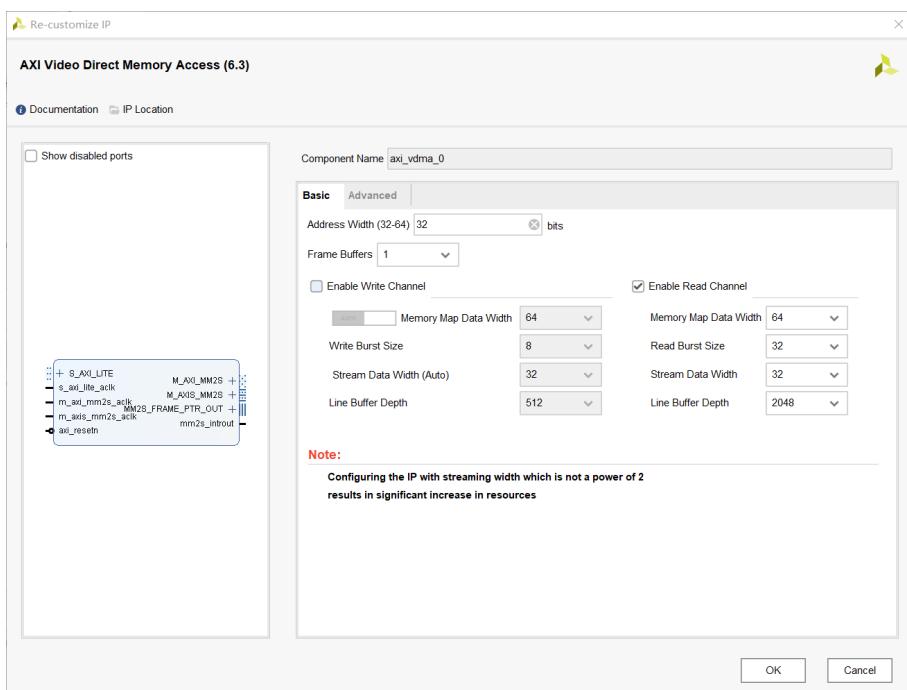


图 4-8 Video DMA IP 核配置

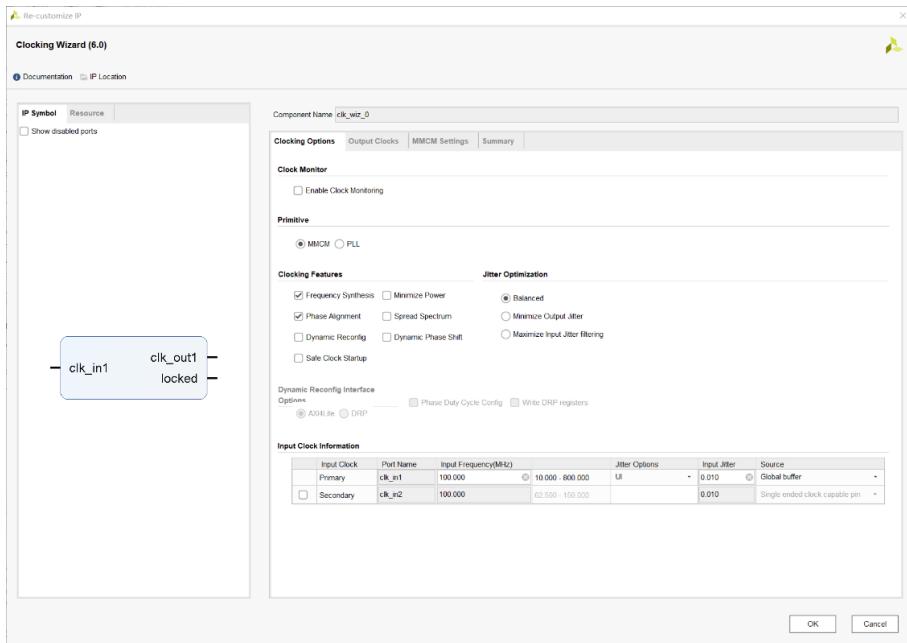


图 4-9 锁相环配置

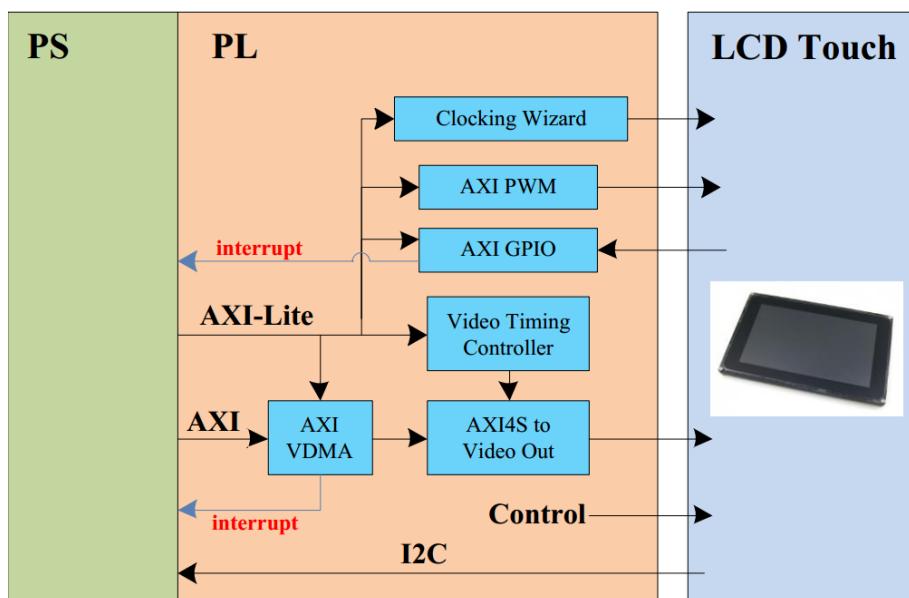


图 4-10 显示器驱动模块的架构图

5 系统仿真与调试

5.1 FPGA 开发流程

5.1.1 开发流程重要步骤说明与图示

5.1.1.1 新建工程

进入 Vivado 开发平台后，首先点击新建工程，如图 5-1 所示。

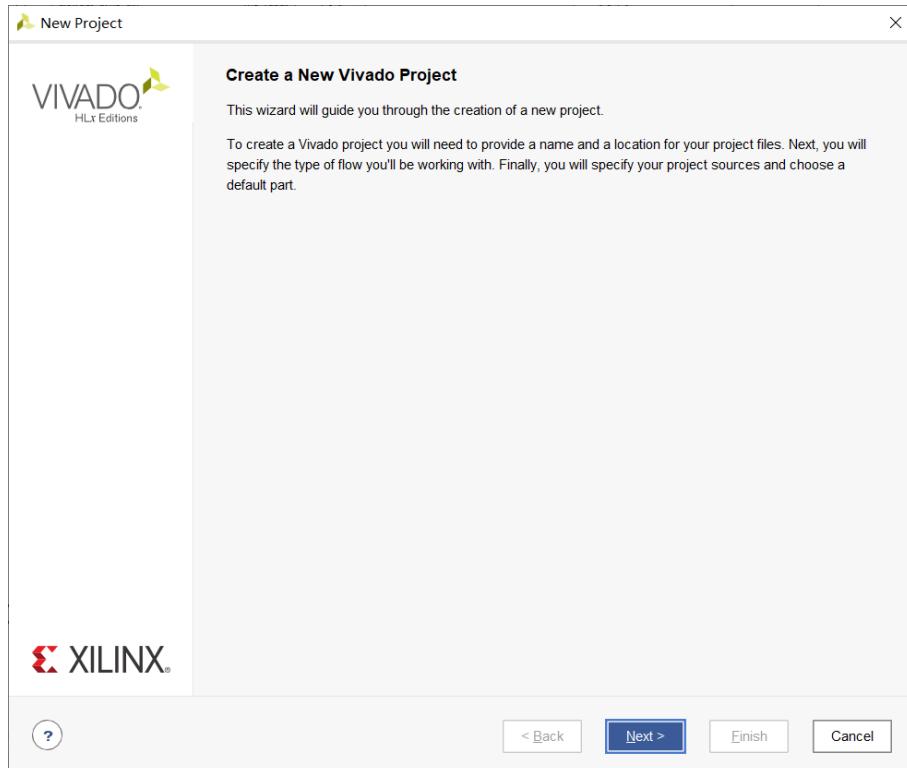


图 5-1 新建工程

确定工程的名字和路径之后，下一步是器件选型，本课程设计选用的 FPGA 型号为 Zynq-7020，因此芯片型号为 xc7z020clg400-2，如图 5-2 所示。

配置好器件型号之后，即完成了工程的创建，如图 5-3 所示。

5.1.1.2 新建文件

进入新的 Vivado 工程之后，我们选择添加 Block Design，因为本系统是软硬件协同的，需要用到处理器，因此选用 Block Design 模块化设计，如图 5-4 所示。

添加好之后即添加 IP、配置 IP 并连线、输出等等，完成后的设计图在上文

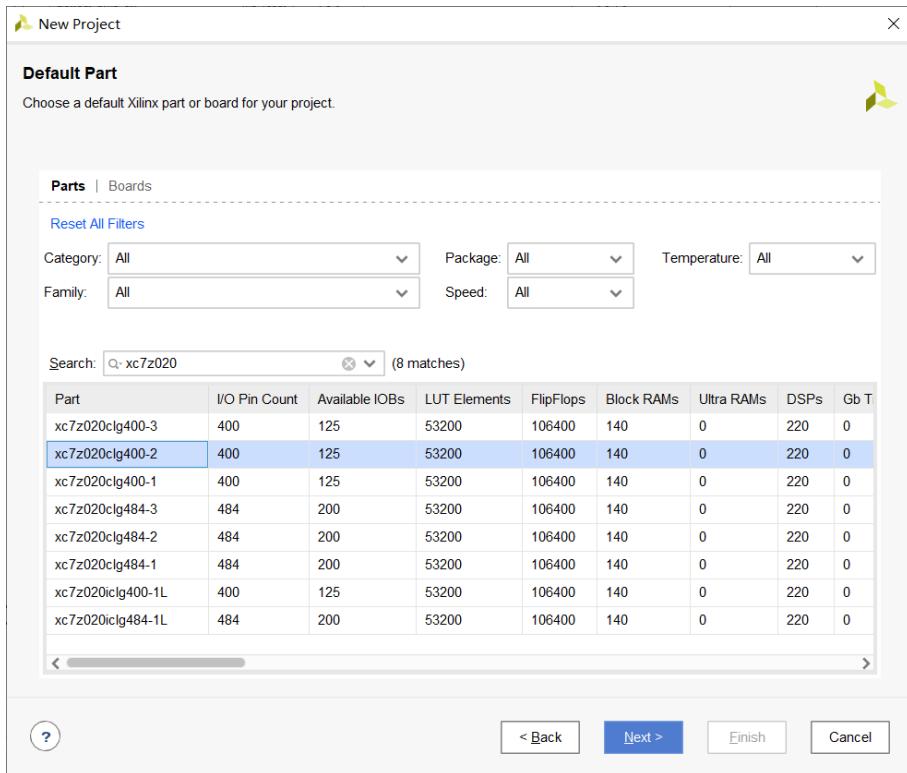


图 5-2 器件选型

中提到。选择让 Vivado 自动生成顶层 Verilog 代码，即可完成硬件部分的逻辑设计，顶层 Wrapper 将会自动生成。

5.1.1.3 功能仿真

在设计每一个模块时，需要对模块进行功能仿真，Xilinx 的 Vivado 平台自带仿真系统，也可以使用业界常用的 Modelsim 进行仿真。本设计使用 Vivado 进行综合前仿真，以验证功能正确性。

编写 testbench 时，使用 DDS（Direct Digital Synthesis，直接数字频率合成）生成模拟的正弦信号作为仿真信号源给状态机进行仿真。仿真模块实例化触发状态机以及 FIFO，DMA 部分由于缺少与处理器的交互没有加入仿真验证。经过修改验证，得到如图 5-5 所示的正确的仿真波形。

5.1.1.4 引脚约束

使用直接编写约束文件的方式，能大大提高开发效率，因为可以通过模板来快速编写约束文件，如图 5-6 所示为节选的约束文件。

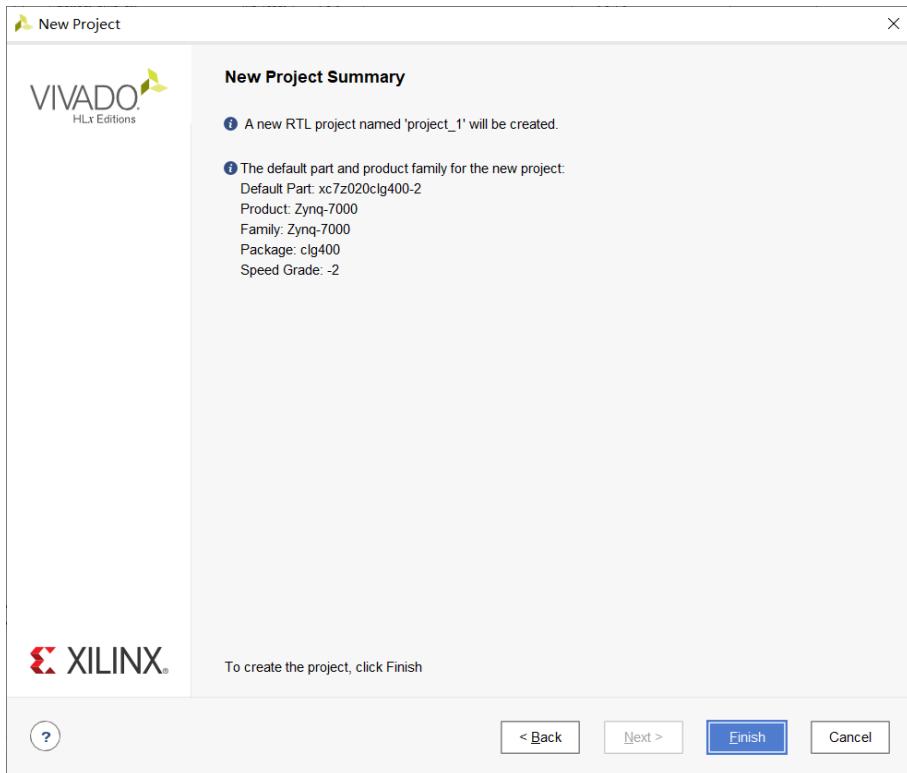


图 5-3 完成创建

5.1.1.5 时序约束

使用 Vivado 进行开发时，一般的时序约束方法有三种。

方法 1：自动创建基时钟和 PLL 输出时钟

这一方法能够自动地约束 PLL 的输入和输出时钟。ALTPPLL mega function 中指定的所有 PLL 参数都用于约束 PLL 的输入和输出时钟。自动更新了 ALTPPLL mega function 的修改。当创建 PLL 的输入和输出时钟时，不必跟踪 PLL 参数的更改或指定正确的值。为了自动约束所有输入和输出，要将 derive_pll_clocks 命

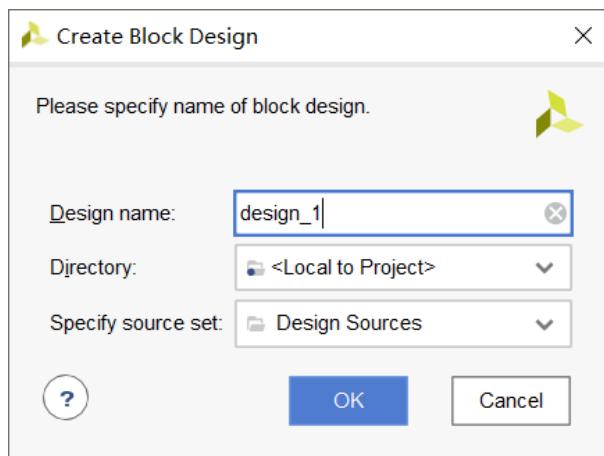


图 5-4 添加 Block Design

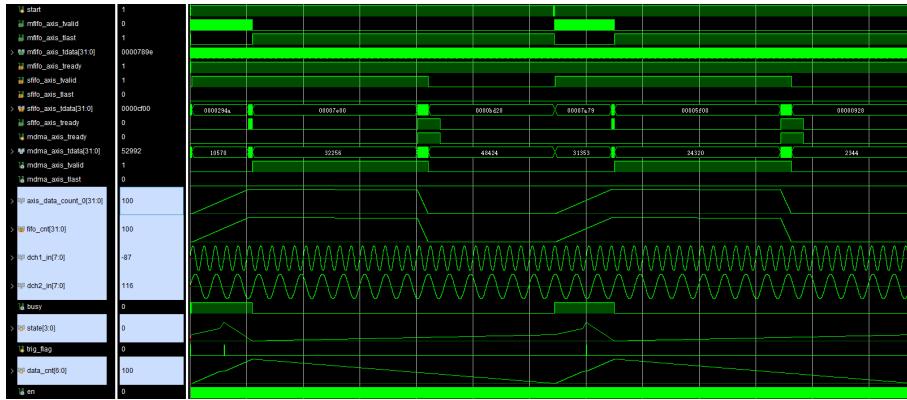


图 5-5 Vivado 功能仿真

```
# "LCD"
set_property PACKAGE_PIN T15 [get_ports {lcd_int_i}]
set_property PACKAGE_PIN U17 [get_ports {lcd_clk_o}]
set_property PACKAGE_PIN W16 [get_ports {lcd_de_o}]
set_property PACKAGE_PIN V16 [get_ports {lcd_hs_o}]
set_property PACKAGE_PIN P15 [get_ports {lcd_vs_o}]
set_property PACKAGE_PIN P16 [get_ports {lcd_pwm_o}]
set_property PACKAGE_PIN W15 [get_ports {lcd_wake_n_o}]
set_property PACKAGE_PIN U19 [get_ports {lcd_bl_en_o}]
set_property PACKAGE_PIN T14 [get_ports {lcd_iic_scl_io}]
set_property PACKAGE_PIN V15 [get_ports {lcd_iic_sda_io}]

set_property PACKAGE_PIN V20 [get_ports {lcd_r_o[0]}]
set_property PACKAGE_PIN T20 [get_ports {lcd_r_o[1]}]
set_property PACKAGE_PIN W20 [get_ports {lcd_r_o[2]}]
set_property PACKAGE_PIN U20 [get_ports {lcd_r_o[3]}]
set_property PACKAGE_PIN Y18 [get_ports {lcd_r_o[4]}]
set_property PACKAGE_PIN N20 [get_ports {lcd_r_o[5]}]
set_property PACKAGE_PIN Y19 [get_ports {lcd_r_o[6]}]
set_property PACKAGE_PIN P20 [get_ports {lcd_r_o[7]}]
```

图 5-6 引脚约束文件

令和 -create_base_clocks 选项一起使用。

方法 2：手动创建基时钟和自动创建 PLL 输出时钟

通过这种方法，可以手动约束 PLL 的输入时钟并且使 TimeQuest analyzer 能够自动约束 PLL 的输出时钟。除此之外，与 ALTPLL mega function 中指定的输入时钟频率相反，可以指定一个不同的输入时钟频率。通过使用 ALTPLL mega function 中指定的参数自动创建 PLL 输出时钟。

方法 3：手动创建基时钟和 PLL 输出时钟

通过这种方法，可以手动约束 PLL 的输入时钟和输出时钟。指定了所有的 PLL 参数并且参数值可以不同于 ALTPLL mega function 中指定的参数值。除此之外，可以尝试各种 PLL 输入和输出频率以及参数。也可以将该方法与 create_clock 和 create_generate_clock 命令的组合一起使用。由于软硬件协同设计中，PL 部分的时钟由 PS 内的锁相环提供，因此无需在约束文件中对 PL 时钟做过多约束。

5.1.1.6 综合 RTL 电路图

设计完成后，便可以进行综合，如果需要在线调试，还需要在综合结束之后进行配置以形成 ILA。图 5-7 是设计综合后的 RTL 电路图。

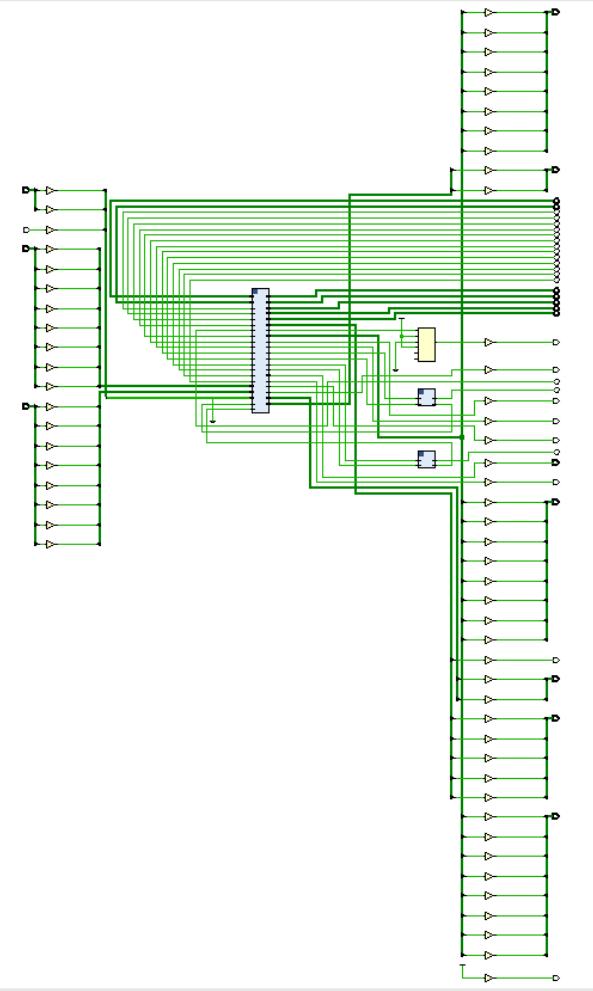


图 5-7 RTL 电路图

5.1.2 综合实现报告与分析

Verilog HDL 是一种硬件描述语言，其最终的目的是生成硬件电路，所以与 C 语言不同，编程时特别在意硬件资源的消耗、时序分析以及功耗分析，如果实现相同的功能，而硬件消耗过多的话，那么这个设计就基本上是失败的。同理，时序、功耗等参数也是分析设计的重点。

5.1.2.1 时序报告分析

在一个时序电路中，一定要满足信号的建立时间和保持时间，否则信号时序出错，数据不能正确的传达。

根据在数字电路基础这门课上学到的知识，可以得到以下的参数关系。

建立时间 (Ts: set up time) 是指在时钟沿到来之前数据从不稳定到稳定所需的时间，如果建立时间不满足要求，那么数据将不能在这个时钟上升沿被稳定地送入触发器；**保持时间** (Th: hold time) 是指数据稳定后保持的时间，如果保持时间不满足要求那么数据同样也不能被稳定地送入触发器。

由于建立时间与保持时间的和是稳定的一个时钟周期，如果时钟有延时，同时数据的延时也较小，那么建立时间必然是增大的，保持时间就会随之减小，如果减小到不满足 D2 的保持时间要求时就不能采集到正确的数据，这时即 $T - (T_{pd} - T_{co} - T_{2min}) < T_4$ ，就不满足要求了，所以 D2 的保持时间应该为： $T - (T_{pd} + T - T_{co} - T_{2min}) \geq T_4$ 即 $T_{co} + T_{2min} - T_{pd} \geq T_4$ 从上式也可以看出如果 $T_{pd} = 0$ 也就是时钟的延时为 0，那么同样要求 $T_{co} + T_{2min} > T_4$ ，但是在实际的应用中由于 T_2 的延时也就是线路的延时远远大于触发器的保持时间即 T_4 ，所以不必要关心保持时间。

图 5-8 为时序分析报告，可以看到，在 100MHz 时钟下，建立时间、保持时间以及脉宽都满足时序约束要求。

Design Timing Summary					
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	3.329 ns	Worst Hold Slack (WHS):	0.036 ns	Worst Pulse Width Slack (WPWS):	3.000 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	29520	Total Number of Endpoints:	29413	Total Number of Endpoints:	11538
All user specified timing constraints are met.					

图 5-8 时序分析报告

5.1.2.2 功耗分析

低功耗是集成电路发展的一个趋势，如果实现一个完好的功能，实现了较低的功耗，那么这样的产品无疑是极有竞争力的。因此需要研究系统功耗，在

Xilinx 的集成开发工具 Vivado 中，带有功耗分析的功能。图 5-9 所示为系统的功耗分析报告。

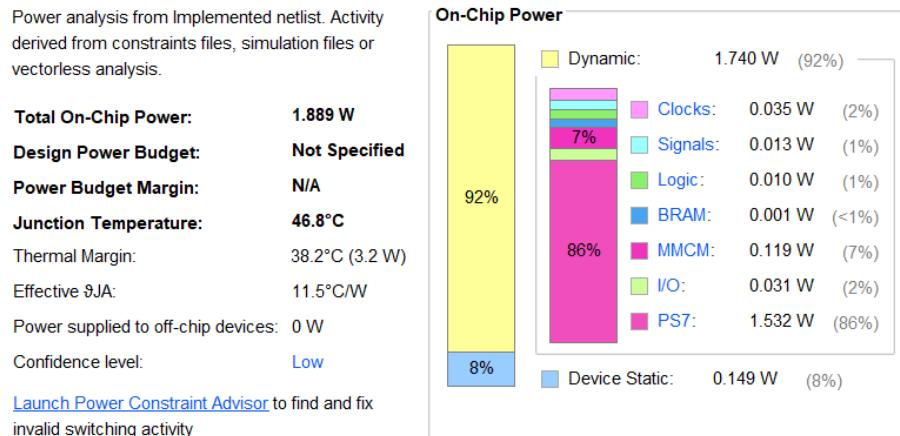


图 5-9 功耗分析报告

从图中可以看出，本设计的功耗主要用在 ARM 核上（即 PS 部分），达到了总功耗的 86%。本设计使用了很多数据传输的 IP 核，这些 IP 核都是需要高速时钟驱动的，而逻辑部分没有占用太多的功耗，可以说本系统在功耗上控制得不错。

5.1.2.3 资源分析

从综合到实现，Vivado 工具会帮我们做一些布局布线的优化，所以综合阶段的资源占用和实现阶段的资源占用是不一样的，因此选用实现阶段后的资源报告来作为最终的资源分析，如图 5-10 所示。

Name	1	Slice LUTs (53200)	Block RAM Tile (140)	Bonded IOB (125)	Bonded IOPADs (130)	OLOGIC (125)	BUFGCTRL (32)	MMCME2_ADV (4)	Slice Registers (106400)	F7 Muxes (26600)	Slice (13300)
system_wrapper	6375	5.5	63	130	1	3	1	10361	119	3262	
system_i (system)	6375	5.5	0	0	0	3	1		119	3262	

图 5-10 资源分析

5.1.2.4 电路实现版图

如图 5-11 所示为我们作品的最终 FPGA 版图。

可以看到，版图中用于实现逻辑的部分不算多，且逻辑占用大多用于通用的 FIFO、DMA、VDMA 等数据传输模块。众所周知，越通用的 IP 越会占用更多的资源去完成通用性，因此选用 Xilinx 的通用 IP 核只是为了提高开发效率，它会

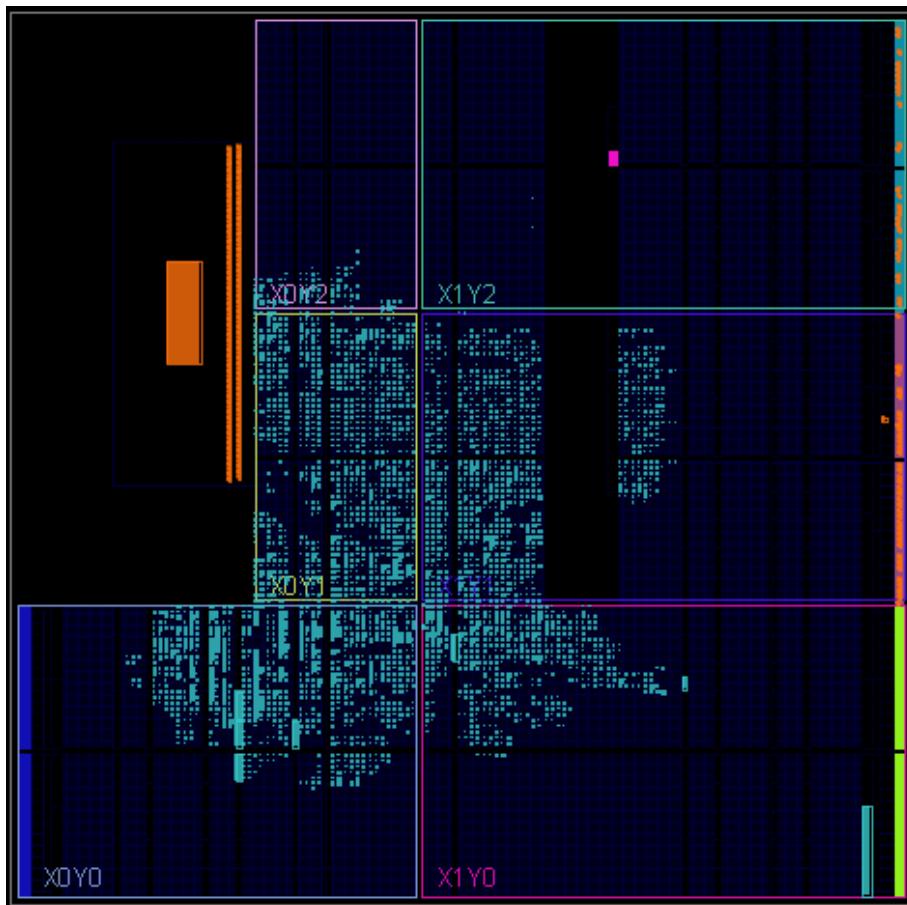


图 5-11 FPGA 实现版图

占用更多的资源。

5.2 FPGA 调试

5.2.1 ILA 测试简介

5.2.1.1 Vivado ILA 简介

逻辑分析仪是从测试设备上采集和显示数字信号的仪器，最主要作用在于时序判定。由于逻辑分析仪不像示波器那样有许多电压等级，通常只显示两个电压（逻辑 1 和 0），因此设定了参考电压后，逻辑分析仪将被测信号通过比较器进行判定，高于参考电压者为 High，低于参考电压者为 Low，在 High 与 Low 之间形成数字波形。

逻辑分析仪与示波器相同，是通过采集指定的信号，并通过图形化的方式展示给开发人员，开发人员根据这些图形化信号按照协议分析出是否出错。尽管图形化的显示已经给开发人员带来不少的方便，但是人工将一串串信号分析出

来不仅麻烦而且极易出错。在这个科技高速发展的社会，一切都在追求高效率。自动化、智能化已经成为协议分析的发展方向。在这个思想的指引下各种测试仪器的协议分析功能出现并发展起来。目前大多数开发人员通过逻辑分析仪等测试工具的协议分析功能可以很轻松的发现错误、调试硬件，加快开发进度，为高速度、高质量完成工程提供保障。

逻辑分析仪的工作就是数据采集、存储、触发、显示，由于它采用数字存储技术，可将数据采集工作和显示工作分开进行，也可同时进行，必要时，对存储的数据可以反复进行显示，以利于对问题的分析和研究。

然而，逻辑分析仪的成本太高，且使用起来麻烦，因此在大多数 FPGA 中都包含嵌入式逻辑分析仪，利用 JTAG 接口即可直接在 PC 端的开发平台上在线调试，观察 FPGA 运行时的内部信号波形与寄存器值等参数，加快了调试人员的开发速度，减少了开发周期。

在 Xilinx 的 FPGA 中，包含 ILA 这个嵌入式逻辑分析仪，在 Vivado 开发平台中即可对其进行配置与运行，与 Altera 公司的 Signal Tap 类似。但 Vivado 的 ILA 可以代码配置，也可以 IP 核配置，相比 Signal Tap 更灵活，如图 5-12 所示为它的 IP 核配置界面。接下来将介绍 Vivado ILA 的使用。

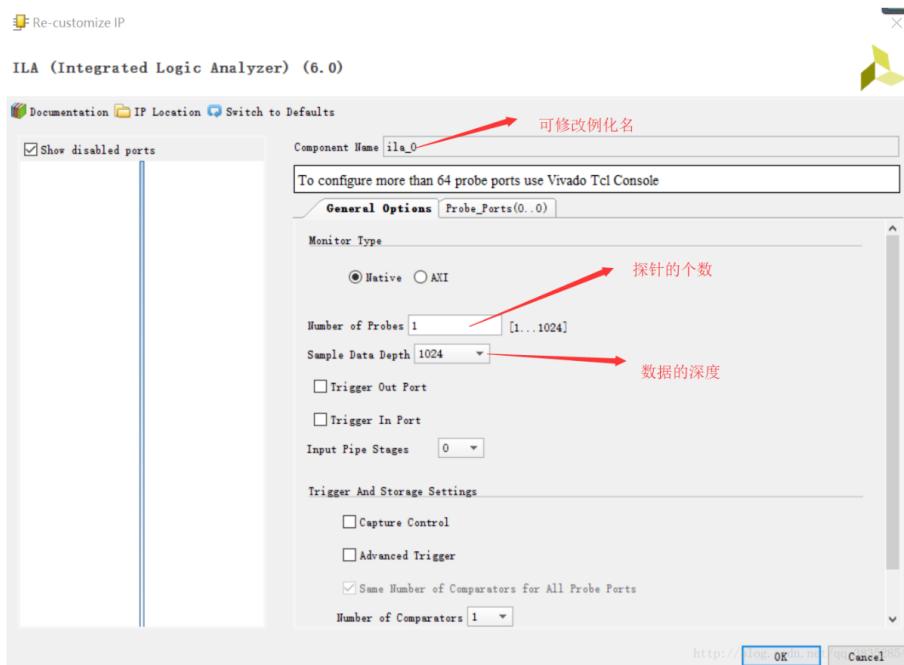


图 5-12 ILA IP 核配置

5.2.1.2 Vivado ILA 的使用

综合完成后，可以打开综合界面进行“debug setup”，即配置 ILA，可以在配置的时候选择采样点数、是否预触发采样、是否捕捉采样等等。一般要开启捕捉采样，方便调试。配置好 ILA 之后再选择实现硬件，生成比特流的操作，烧写到 FPGA 板子之后，便可以在运行时在线调试。

5.2.2 ILA 调试分析

5.2.2.1 ADC、采样、触发系统、FIFO、DMA 综合调试

使用 ILA 进行调试，使用信号源对 ADC 输入不同频率的信号，设置不同的水平档位，处理器中控制底层数据开始采样，设置 ILA 触发条件为状态机跳变为预触发的状态，对模块进行综合验证调试，从而测试 ADC 数据、采样系统、触发系统、FIFO、DMA 的功能正确性，得到以下调试图片。经过多次调试修改，验证功能正确。

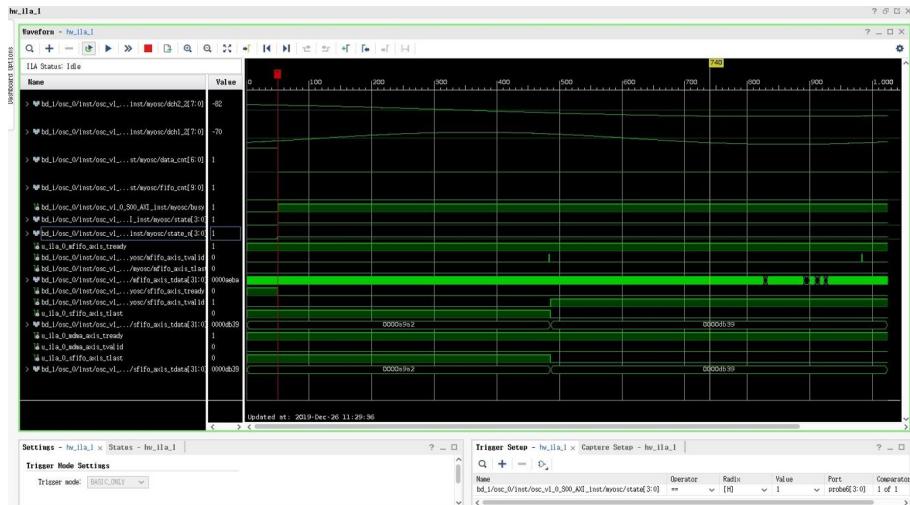


图 5-13 50us/div 档位 ILA 调试图片

5.2.3 液晶显示屏调试

由于液晶显示屏的时序较为清晰，故不采用嵌入式逻辑分析仪进行调试，而是直接观察输出效果。从软件顶层查看底层硬件的锁相环配置情况，即可得知显示屏的工作频率是否正确。在调试的过程中还是遇到了一些 bug，有时候屏幕会进入花屏，最终发现是锁相环没有约束住，最后干脆就不采用可重构的锁相环，



图 5-14 0.1us/div 档位 ILA 调试图片

直接在 PL 部分生成一个 51.2MHz 的锁相环，用来给显示屏提供时钟。

5.2.4 人机交互界面调试

人机交互也是本作品的一个核心亮点，除了显示以外，还必须做触屏操作的处理，因此在触屏终端服务函数设置断点，观察每次进入这个函数（即触屏）的时候，收到的值是否正确，是否能够收到正确的值（即是否满足 IIC 协议）。经验证无误后再编写具体的功能服务。

5.2.5 任务调度调试

本设计中采用了嵌入式实时操作系统来进行任务调度，为了验证任务调度的正确性，在调试中采用了“任务中断”的方法，对于低优先级任务，通过触屏、刷屏等高优先级任务，甚至中断，来打断它，测试优先级调度的准确性。

5.3 系统功能测试

在所有软件调试完成之后，我们进行了系统级的调试，主要是测试本作品的功能指标是否满足。

根据题意，我们联合调试了作品的波形显示、挡位切换、触发调节、存储与调用功能、触发电平、参数测量、光标测量等功能。并用信号发生器产生各种波形来进行功能测试。最终结果也较好的反映了示波器的所有正常功能，并有部分指标超出了题目要求。如图 5-15 所示即为我们的系统功能调试照片。

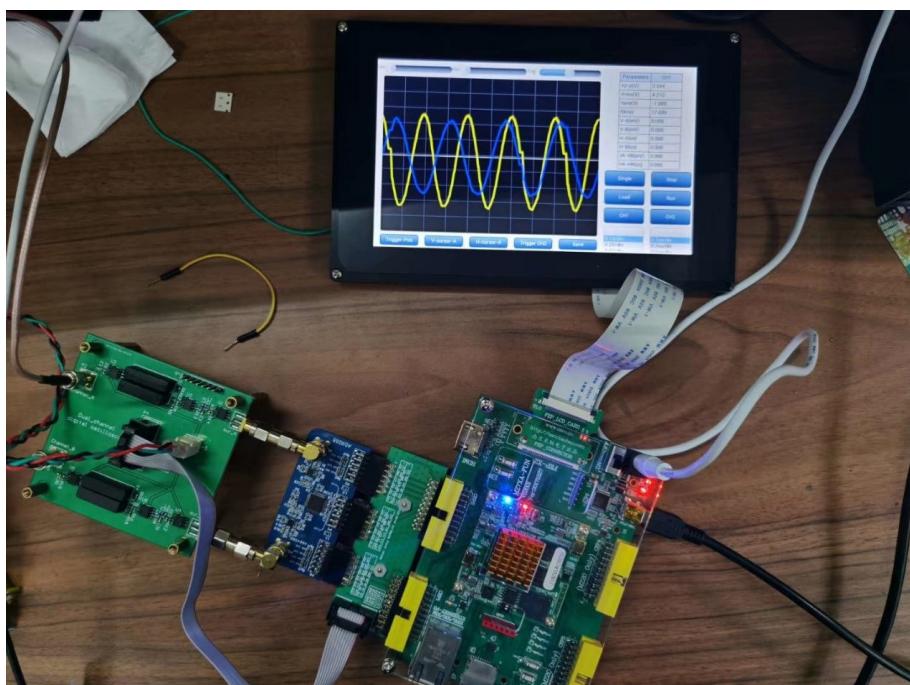


图 5-15 示波器整体照片

6 测试方案与测试结果

本节单独列出系统功能测试时所采用的测试方案与系统的测试结果。

6.1 测试环境

表 6-1 测试环境

仪器	品牌	型号
示波器	Tektronix	MDO2002B 型数字示波器
信号发生器	RIGOL	DG1062 型 160M 任意波形发生器
电源	ZhongCe	DF1743003C 型稳压源

6.2 测试方案

6.2.1 方波校准信号测试方案

使用示波器测试校准方波，测试其频率及幅值。

6.2.2 单次触发扫描测试方案

程控触发电平，观察简易数字信号示波器能否产生扫描电压，并在信号上升沿开始显示波形。

6.2.3 扫描速度测试方案

信号源输入峰峰值为 1V 频率可变的正弦信号，观察并记录专用示波器的周期读数，调整作品示波器各项挡位并记录数值，与标准值进行比较并计算误差。

6.2.4 垂直灵敏度测试方案

信号源输入频率为 100kHz 幅值可变的正弦信号，观察并记录专用示波器的幅值读数，调整作品示波器各项挡位并记录数值，与标准值进行比较并计算误差。

6.3 测试结果与数据

6.3.1 方波校准信号测试

经测试，本设计产生的方波信号频率 $f = 99.90\text{kHz}$ ，峰-峰值为 $V_{\text{p-p}} = 0.3\text{V}$ 。

6.3.2 单次触发扫描测试

观察结果显示简易数字信号示波器能够产生扫描电压，并显示波形。如图 6-1 所示为单次触发的显示波形。

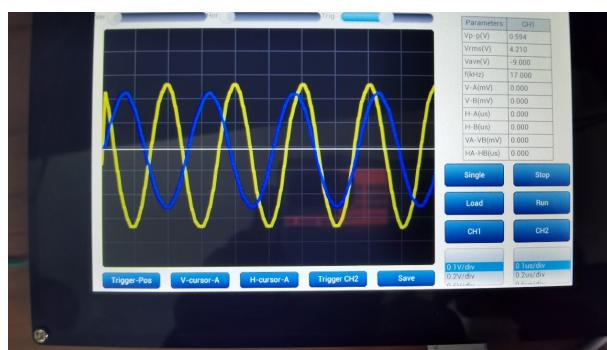


图 6-1 单次触发测试波形

6.3.3 扫描速度测试

* 信号源输入峰峰值为 $V_{\text{p-p}} = 1\text{V}$ 的正弦信号。

表 6-2 扫描速度（周期）测试表

扫描速度	周期		
	实际值	测量值	误差
10us/div	50us	49.0us	2.0%
	20us	20.4us	2.0%
	10us	10.0us	0
1us/div	5us	5.2us	4.0%
	2us	2.02us	1.0%
	1us	1.04us	4.0%
100ns/div	1000ns	982ns	1.8%
	500ns	516ns	3.2%
	100ns	96ns	4.0%

6.3.4 垂直灵敏度测试

* 信号源输入频率为 $f = 100\text{kHz}$ 的正弦信号。

表 6-3 垂直灵敏度测试表

垂直灵敏度	幅值		
	实际值	测试值	误差
0.5V/div	1V	1.01	1%
	2V	2.02V	1%
	4V	4.07V	1.7%
0.2V/div	400mV	412mV	3.0%
	800mV	787mV	1.6%
	1.6V	1.62V	1.3%
0.1V/div	200mV	204mV	2.0%
	400mV	410mV	2.5%
	800mV	822mV	2.7%

6.4 测试结果分析

根据题意，联合调试了作品的波形显示、挡位切换、触发调节、存储与调用功能、触发电平、参数测量、光标测量等功能。并用信号发生器产生各种波形来进行功能测试。最终结果也较好的反映了示波器的所有正常功能，并有部分指标超出了题目要求。

7 问题分析与总结

7.1 问题分析

7.1.1 数字示波器硬件问题

在数字示波器这个课设作品中，刘刚负责的主要是前端信号调理电路和 AD 采样电路这两个模块，整个设计的过程比较简单，需要注意的地方是信号调理电路中运放的选型，既要保证足够的带宽，使能处理的信号频带尽可能大，又要尽可能挑选单位增益稳定的精密运放。最开始刘刚选用了 TI 公司的精密运放 OPA820 构成第一级的跟随器，但是效果并不好，跟随后的信号波形出现了严重失真，最后经过换过几次运放后，选择了 OPA656，它的输入失调参数都较小，能够很好的实现有关功能。

7.1.2 数字示波器软件问题

课设的过程中，我们第一次尝试写数字示波器的代码，过程中我们遇到了很多问题，在此罗列一些在过程中遇到的问题。

1、在多个过程块中对同一变量赋值

开始我们按照自己的思路，将各个功能模块用 verilog 写了出来，但是刚开始我们的思维还没有完全从 C 语言、汇编这类框架中跳脱出来，也没有完全领悟到 verilog “硬件描述语言”的本质，它描述的是电路状态，最终也要被映射成实际电路，我在不同的过程块中对同一变量进行赋值，实际就相当于电路中的“线与”，这种情况是应该被避免的，而我们等到 modelsim 仿真编译时出现很多 error 才发现这个问题，然后又是重新捋清思路，一点点改代码，直到没有 error。

2、按键消抖的边沿触发问题

按键消抖模块的原理是每隔 50ms 分别读取按键的电平，然后对读出的值做异或运算。一般来说，一次合法的按键按下，会产生两个脉冲，将两个波形分别存储，分别过 D 触发器然后就会得到一个表示按键被按下的脉冲，持续时间为一个周期，然后可以用电平检测的方法来判断按键是否被按下。在写这段代码的过程中，按键按下的时候变为低电平，所以应该判断下降沿来决定是否产生第一个脉冲，然后过 D 触发器合成一个脉冲，结果分析 modelsim 波形图的时候却发现

现最后的指示脉冲出现在了测试代码里按键松开后，相当于多了一段延时，我们向同学请教才发现是最后产生脉冲时非逻辑的对象给错了，然后才改了过来。

3、其它一些问题

这次写整体的 verilog 代码，我们也犯了很多粗心的错误，比如阻塞性赋值与非阻塞性赋值出现在同一个过程块里、使用的端口未声明、编写测试代码是变量类型定义错误等等。出现这些问题主要还是因为经验不足，写代码不够熟练，我相信多经过几次练习，这种错误出现几率一定会大大降低。

7.2 心得感悟

这次电子课程设计我们完成的双通道数字示波器能够满足题目中的要求，而且采样率达到 100M，输入信号的范围也可达 5M，应该算是一个不错的作品。

整个过程中我最大的感悟：要学会 FPGA，一定要实践实践多实践，只有真正尝试去做了，才会明白过程中有多少坑，然后一点一点把不会的地方学会，把存在的问题解决，逐渐积累经验，最后完成一个作品时才能游刃有余。

8 参考文献

- [1]. 罗杰.Verilog HDL 与数字 ASIC 设计基础, 2008, 华中科技大学出版社.
- [2]. 王贞炎.FPGA 应用开发和仿真, 2018, 机械工业出版社.
- [3]. [美]Bruce Carter. 运算放大器权威指南(第四版), 2014, 人民邮电出版社.
- [4]. 谭浩强.C 语言程序设计, 2010, 清华大学出版社.
- [5]. 陈尚松. 电子测量与仪器(第四版), 2018, 电子工业出版社.

附录 A 【2007 国赛 C 题】数字示波器

数字示波器 (C 题)

【本科组】

一、任务

设计并制作一台具有实时采样方式和等效采样方式的数字示波器，示意图如图 1 所示。

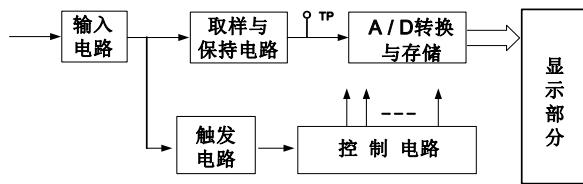


图 1 数字示波器示意图

二、要求

1. 基本要求

- (1) 被测周期信号的频率范围为 $10\text{Hz} \sim 10\text{MHz}$, 仪器输入阻抗为 $1\text{M}\Omega$, 显示屏的刻度为 $8\text{div} \times 10\text{div}$, 垂直分辨率为 8bits, 水平显示分辨率 ≥ 20 点/div。
- (2) 垂直灵敏度要求含 $1\text{V}/\text{div}$ 、 $0.1\text{V}/\text{div}$ 两档。电压测量误差 $\leq 5\%$ 。
- (3) 实时采样速率 $\leq 1\text{MSa/s}$, 等效采样速率 $\geq 200\text{MSa/s}$; 扫描速度要求含 $20\text{ms}/\text{div}$ 、 $2\mu\text{s}/\text{div}$ 、 $100\text{ ns}/\text{div}$ 三档, 波形周期测量误差 $\leq 5\%$ 。
- (4) 仪器的触发电路采用内触发方式, 要求上升沿触发, 触发电平可调。
- (5) 被测信号的显示波形应无明显失真。

2. 发挥部分

- (1) 提高仪器垂直灵敏度, 要求增加 $2\text{mV}/\text{div}$ 档, 其电压测量误差 $\leq 5\%$, 输入短路时的输出噪声峰-峰值小于 2mV 。
- (2) 增加存储/调出功能, 即按动一次“存储”键, 仪器即可存储当前波形, 并能在需要时调出存储的波形予以显示。
- (3) 增加单次触发功能, 即按动一次“单次触发”键, 仪器能对满足触发条件的信号进行一次采集与存储 (被测信号的频率范围限定为 $10\text{Hz} \sim 50\text{kHz}$)。
- (4) 能提供频率为 100kHz 的方波校准信号, 要求幅度值为 $0.3\text{V} \pm 5\%$ (负载电阻 $\geq 1\text{ M}\Omega$ 时), 频率误差 $\leq 5\%$ 。
- (5) 其他。

C-1

图 A-1 数字示波器-1

三、说明

1. A/D 转换器最高采样速率限定为 1MSa/s，并要求设计独立的取样保持电路。为了方便检测，要求在 A/D 转换器和取样保持电路之间设置测试端子 TP。
2. 显示部分可采用通用示波器，也可采用液晶显示器。
3. 等效采样的概念可参考蒋焕文等编著的《电子测量》一书中取样示波器的内容，或陈尚松等编著的《电子测量与仪器》等相关资料。
4. 设计报告正文中应包括系统总体框图、核心电路原理图、主要流程图、主要的测试结果。完整的电路原理图、重要的源程序和完整的测试结果可用附件给出。

四、评分标准

	项 目	应包括的主要内容	分 数
设计 报告	系统方案	比较与选择 方案描述	6
	理论分析与计算	等效采样分析 垂直灵敏度 扫描速度	12
	电路与程序设计	电路设计 程序设计	12
	测试方案与测试结果	测试方案及测试条件 测试结果完整性 测试结果分析	12
	设计报告结构及规范性	摘要 设计报告正文的结构 图表的规范性	8
	总分		50
基本 要求	实际制作完成情况		50
发挥 部分	完成第（1）项		22
	完成第（2）项		7
	完成第（3）项		7
	完成第（4）项		6
	其他		8
	总分		50

C-2

图 A-2 数字示波器-2

附录 B 平台说明及主要元器件清单

本设计使用的 FPGA 开发平台为米联客 MZ7XA XILINX FPGA 开发板，MZ7XA 是米联电子 Zynq-7000 系列开发平台一款全新的高端产品。其性价比高、资源丰富，拥有摄像头、HDMI、千兆以太网、FEP 等众多高性能接口，适合用于本设计的开发。FPGA 开发平台如图 B-1 所示。

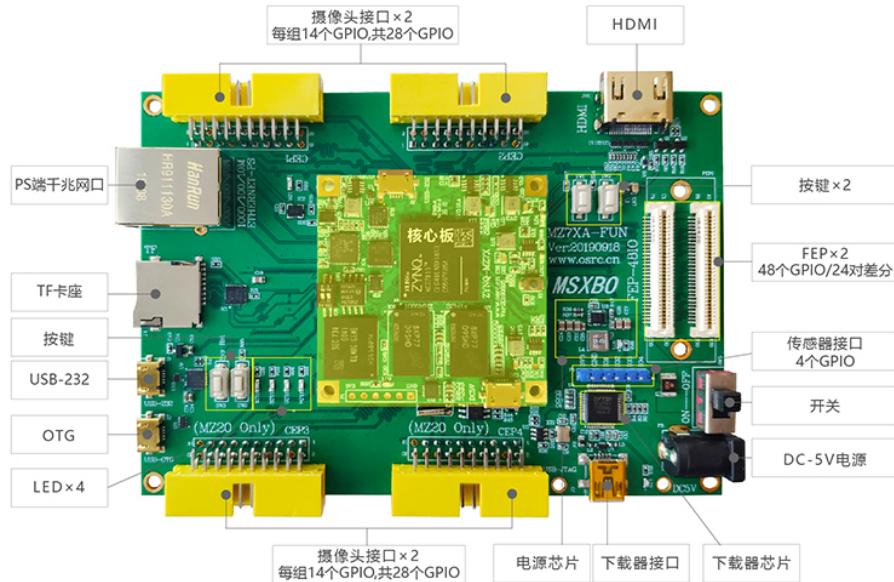


图 B-1 FPGA 开发平台

平台及主要元器件清单如表 B-1 所示。

表 B-1 平台及主要元器件清单

元件种类	元件型号
开发平台	米联客 MZ7XA
模数转换器	AD9288
运算放大器	OPA656
运算放大器	OPA820
高速差分放大器	AD8132