目录

济小台

——基于无剑 100 SoC 的安防云台

1 背景

近年来视频云台成为逐渐普及的产品，从安防类的某品牌水滴摄像头、某家摄像头、某度摄像头，到用于运动拍摄的 XPro 摄像头，再到手机、相机稳定器，如某疆公司的某眸手机云台、某影相机云台。根据调查，市面上的稳定器多用步进电机，而安防类摄像头多用直流无刷电机，原因在于步进电机响应快、控制精度高，价格高于低功率的直流电机。

相比于直流电机，步进电机除了具有上述的响应快、精度高的有点，还有扭矩大、相比于直流电机在高低负载下消耗相当的能量，步进电机在低负载运行状态下低能耗、控制程序简单灵活、损修率低的优点，使得步进电机在监控云台上具有绝佳的优势。

2 简介

作为一款高度灵活精确的电动云台，济小台可以在 360 度全方位的极大空间运转下，适用于多种应用场合，可以用在摄像机的精确快速监视，物体捕捉。其总体构造由中天微电子公司开发的核心为 Xilinx XC7A200T3B 的 FPGA 开发板 FMX7AR3B，其中搭载有平头哥半导体有限公司的无剑 100 嵌入式系统、主板供电模块、步进电机、电机驱动器以及蓝牙通讯单元组成。

本作品旨在设计一款智能安防监控云台，“济小台”。其具有如下特点：

- 控制响应快：高达 3m/s 的运行速度，25ms 的响应延时；
- 智能化控制：使用平头哥半导体公司的开源无剑 100 SoC 作为主控核心，搭配多种模块协同控制，实现济小台的百分百可编程；
- 低功耗：低功耗的无剑 100 SoC 使得济小台的工作能耗极低，每晚不用 0.1 度电；

- 蓝牙通讯协同：通过搭载在济小台的蓝牙通讯模块，使得手机、电脑等设备实时接入控制云台姿态，监视家中异常情况，后续开发可以接入蓝牙 mesh 协议，与家中网关通讯，实现全屋智能。

3 设计方案描述

3.1 总体方案概述

3.1.1 无剑 100 SoC 在 FPGA 上的实现

本节内容是在参考 Lianglonghui 发布在平头哥芯片开放社区（OCC）的博文[1]后得以成功实现，本团队撰写的英文版教程发布在 jiayi’ s blog 供参考。

平头哥在 GitHub 上发布的开源 wujian100_open 项目中包括了如下的目录树

```
Directory Structure
 |--Project                 //开源项目目录
 |--riscv_toolchain         //工具链
 |--wujian100_open          //
     |--case                //模拟实例
     |--doc                 //用户手册
     |--fpga                //FPGA 脚本
     |--lib                 //用于模拟的编译脚本
     |--regress             //回归结果
     |--sdk                 //软件开发套件（sdk）
     |--soc                 //SoC 的 RTL 源码
     |--tb                  //testbench
     |--tools               //模拟脚本与设定文件
     |--workdir             //模拟目录
     |--LICENSE
     |--README.md
```

图表 1 目录树

在 windows 系统下生成比特流文件需要使用到 vivado 软件，本设计使用的版本

[1] wujian100_open 的 FPGA 实现——如何用 vivado 生成 wujian100_open 的比特流文件. Lianglonghui.平头哥芯片开放社区

为运行在 windows10 环境下的 xlinx vivado 2018.3 HLX 版本，需要用到上述目录树中 soc 与 fpga 中的文件。

首先我们将在 fpga 文件夹中的*wjian100_open_fpga_top.v*（顶层文件）与 soc 文件夹中的所有文件（设计源文件）加入源文件目录；然后添加官方提供的 xdc 约束文件*XC7A200T3B.xdc*；随后将*apb0_params.v*、*apb1_params.v*、*timers_params.v*、*wdt_params.v*四个头文件类型改为“Verilog Header”头文件，并将*wujian100_open_fpga_top.v*文件置顶，作为首先编译的文件以避免一些编译错误。

由于官方提供的*XC7A200T3B.xdc*文件中不包含时序约束条件，我们同时也需要增加下述代码块中的时序约束代码至约束文件中以创建系统时钟。

```
create_clock -name {EHS} [get_ports PIN_EHS] -period 50 -waveform {0 25}
create_clock  -name {JTAG_CLK} [get_ports PAD_JTAG_TCLK] -period 1000 -waveform {0 500}

set_clock_groups -asynchronous -name {clkgroup_1} -group [get_clocks {EHS JTAG_CLK}]

set_false_path -through [get_ports PIN_EHS]

#set_clock_groups -name {Inferred_clkgroup_0} -asynchronous -group [get_clocks
{wujian100_open_top|PAD_JTAG_TCLK}]

set_property ASYNC_REG TRUE [get_cells
{x_aou_top/x_rtc0_sec_top/x_rtc_pdu_top/x_rtc_clr_sync/pclk_load_sync2_reg}]
set_property ASYNC_REG TRUE [get_cells
{x_aou_top/x_rtc0_sec_top/x_rtc_pdu_top/x_rtc_clr_sync/rtc_load_sync2_reg}]
set_property ASYNC_REG TRUE [get_cells
{x_aou_top/x_rtc0_sec_top/x_rtc_pdu_top/x_rtc_clr_sync/pclk_load_sync1_reg}]
set_property ASYNC_REG TRUE [get_cells
{x_aou_top/x_rtc0_sec_top/x_rtc_pdu_top/x_rtc_clr_sync/rtc_load_sync1_reg}]
set_property ASYNC_REG TRUE [get_cells {x_cpu_top/CPU/x_cr_had_top/A15d/A74/A10b_reg}]
set_property ASYNC_REG TRUE [get_cells {x_cpu_top/CPU/x_cr_had_top/A15d/A74/A18597_reg}]
set_property ASYNC_REG TRUE [get_cells {x_cpu_top/CPU/x_cr_had_top/A15d/A1862d/A10b_reg}]
set_property ASYNC_REG TRUE [get_cells
{x_cpu_top/CPU/x_cr_had_top/A15d/A1862d/A18597_reg}]
set_property ASYNC_REG TRUE [get_cells {x_cpu_top/CPU/x_cr_had_top/A15d/A75/A10b_reg}]
set_property ASYNC_REG TRUE [get_cells {x_cpu_top/CPU/x_cr_had_top/A15d/A75/A18597_reg}]
```

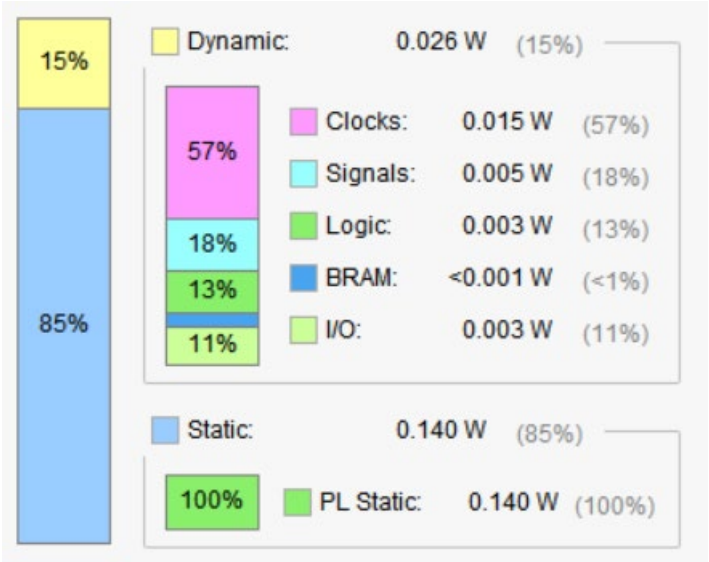
图表 2 时序约束语句

随后，根据上述博客中指出的问题，需要改正官方约束文件中的某一时钟管脚的变量名，由*PAD_JTAG_TCLK_c*更正为*PAD_JTAG_TCLK*。

最后综合生成比特流文件，在 Xlinx XC7A200T3B 开发板下的资源消耗如图表 3 所示，动态与静态功率消耗如图表 4 所示，可以看到总体资源使用率少于 25%，动态功率低于 0.1w，总功率低于 0.2w。作为一款低功耗 SoC，wujian100 展示出了极其优秀的功率以及低资源利用的特性。

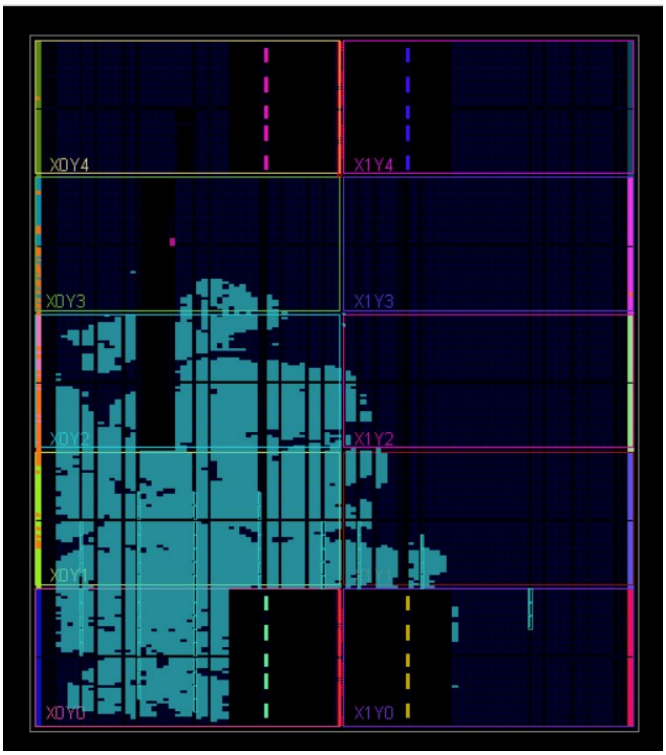
图表 3 wujian100 的资源消耗统计表

资源名称	使用数量	总共可用	利用率（百分比）
LUT	26744	133800	19.99
LUTRAM	72	46200	0.16
FF	13415	267600	5.01
BRAM	64	365	17.53
IO	62	285	21.75
BUFG	2	32	6.25



图表 4 功率消耗示意图

在实现中可以看到如图表 5 所示的版图设计，同样可以发现其器件布局仅占到版图中不到 25%的区域，也说明了 wujian100 的低资源消耗水平。



图表 5 版图

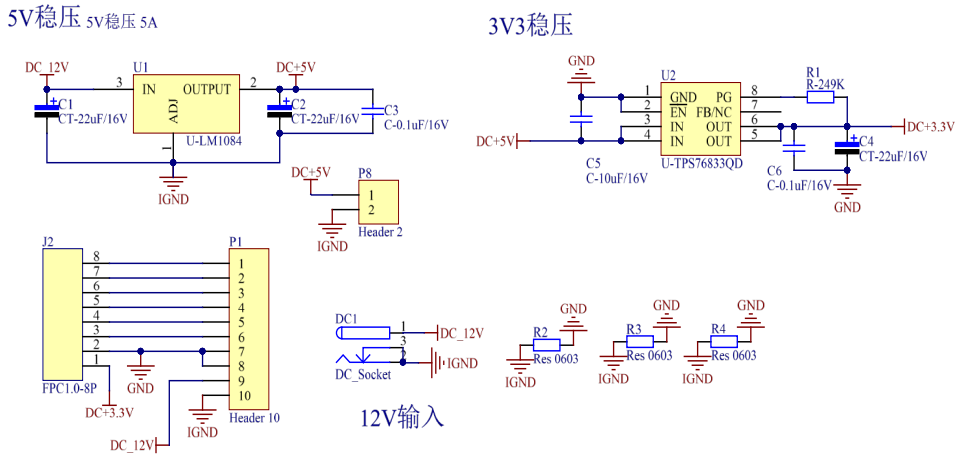
3.1.2 硬件方案

3.1.2.1 电源方案

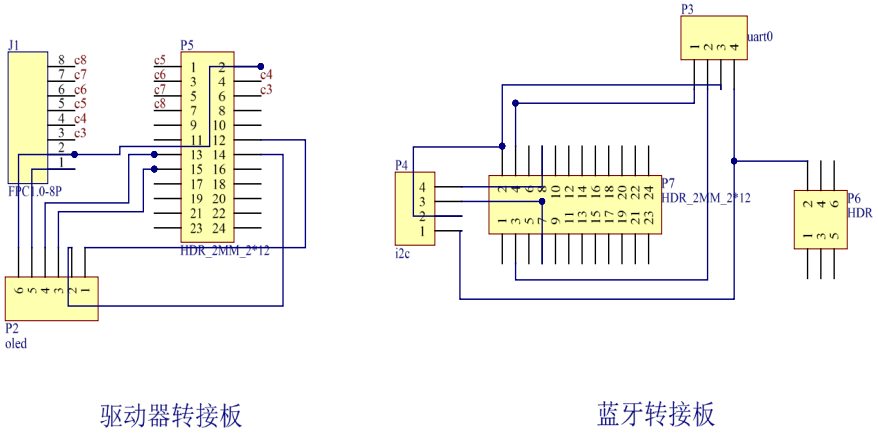
整个系统可分为两部分电源，一是为步进驱动器和电机供电的直流 12V 电源，另一个是为 FPGA 开发板供电的直流 5V 电源。两部分电源共用一个$AC{\sim}220V$转$DC12V-8A$的电源适配器。

步进驱动器和电机由 12V 直流电源直接供电，经检测总电流在 2A 以内，满足电源适配器的功率要求。FPGA 开发板 XC7A200T3B 由直流 12V 电源经降压电路降至 5V 后供电，降压电路采用 LM1084 稳压，最大输出电流可达 5A，可以满足 FPGA 3A 电流供电需求。

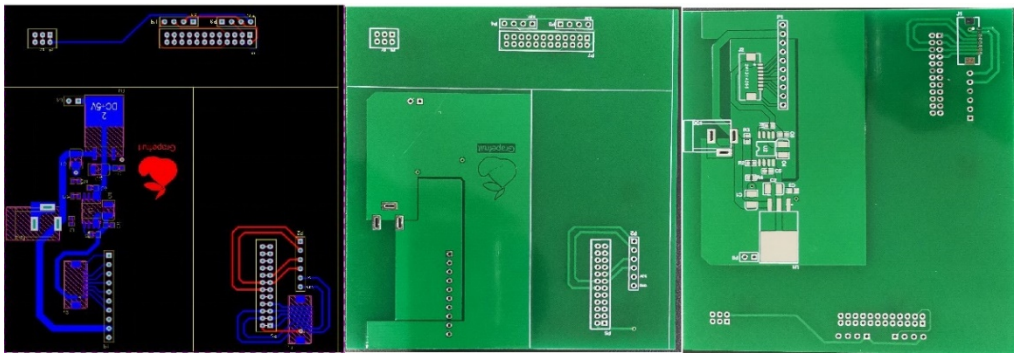
原理图如图表 6、图表 7 所示，PCB 设计图及实物图如图表 8 所示



图表 6 稳压供电模块原理图



图表 7 转接板原理图

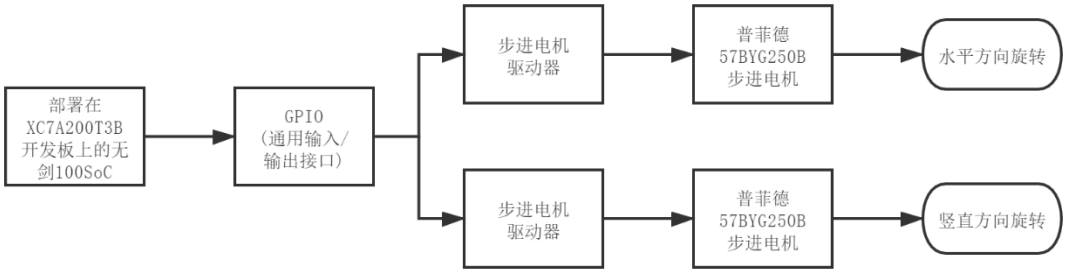


图表 8 稳压供电及转接板 PCB 设计图与实物图

3.1.2.2 云台驱动方案

如图表 9 所示，动力系统的主要模块由两个普菲德 57BYG250B 步进电机构成，其

扭矩为$2.3N\cdot m$，步距角为$1.8°$，驱动方式为二相直流驱动，单相最大电流为$3A$，最大功率 160W（控制器最大功率），其电气参数如图表 10 所示。分别实现两个旋转方向（水平、竖直）自由度，通过两个细分高达 25600，电流范围为$1.0\sim4.2A$，工作电压为$DC\ 9\sim24V$的 DM542 步进驱动器与无剑 100 SoC 连接，步进电机驱动器具有较强的抗干扰性，噪音低的良好特性。

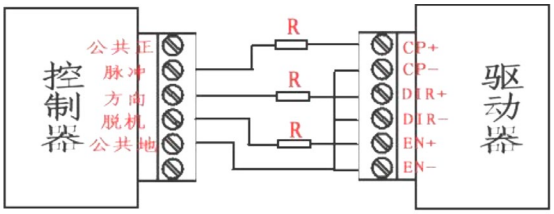


图表 10 驱动方案系统框图

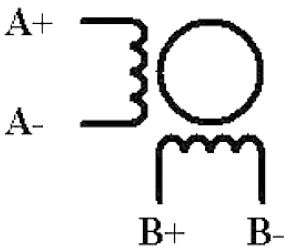
图表 11 步进电机电气参数

1	步距角	每步1.8°
2	电压	$4.7V$
3	电流	每相 $3A$
4	电阻	每相 $1.36\pm10\%\ \Omega$
5	电感	每相 $3.75\pm20\%\ mH$
6	保持力矩	$25kg\cdot m$
7	扭矩	$2.3N\cdot m$
8	绝缘等级	B

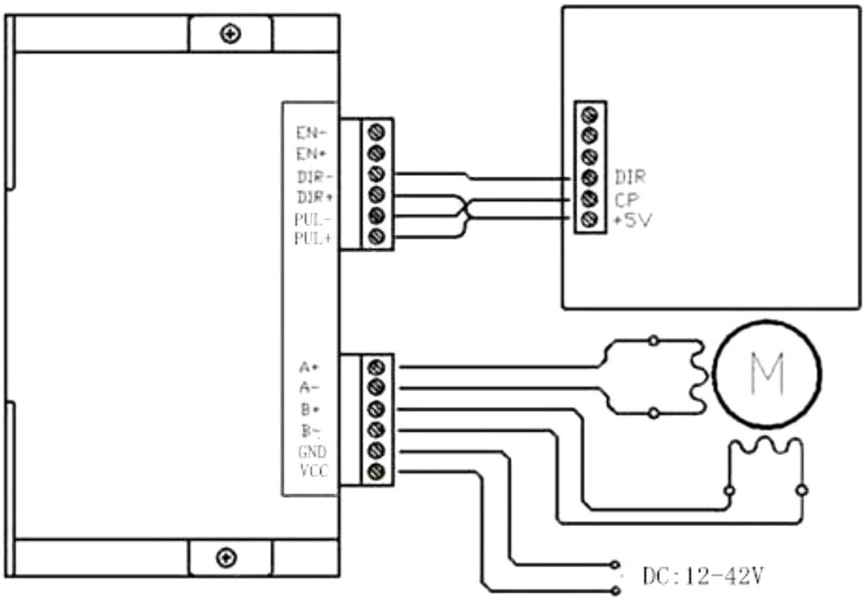
其中搭载在 XC7A200T3B 开发板上的 wujian100 SoC 与步进电机驱动器的连接方式为共阴极连接，如图表 12 所示；步进电机驱动器与步进电机的连接方式如图表 13 所示。控制器使用 1 细分，即步距角为实际电机步距角，带负载电机角度控制精度在水平和竖直方向分别为，$3.1°$和$2.2°$，即竖直方向可实现每次$2.2°$的旋转，水平方向每次可实现$3.1°$的旋转。



图表 14 连接方式示意图



图表 15 57 步进电机四线内部连接方式



图表 16 控制器与电机的总体连接方式

3.1.2.3　中央控制器和交互外设

• 中央控制器

中央控制器采用平头哥开源 wujian100 SoC，是一个基于 FPGA 的 SoC 平台，支持通过 EDA 工具进行前端仿真和 FPGA 片上测试。本设计使用中央控制器根据用户指令生成对应的固定数量的脉冲宽度调制信号（PWM），通过安全稳定并且简约美观的 FPC 排线把步进驱动器的 PWM 信号以及方向信号输入端口和中央控制器的 PWM 信号和控制信号生成端口相连。

• 交互外设

交互外设包括用于显示控制器内部运行状态的有机发光二极管（OLED）显示模块以及用于无线通讯的蓝牙模块。OLED 使用 SPI 协议进行指令控制和数据读写；

蓝牙模块通过串口与其他模块进行透传模式通信，即插即用，方便灵活。两者均通过转接板与中央控制器连接，避免使用杜邦线连接而导致的数据干扰、断连等造成传输不稳定的潜在风险。

图表 17 OLED 显示屏

图表 18 蓝牙模块

3.1.3 软件方案

本设计软件部分使用无剑 100 开发平台配套的 SDK 以及图形化开发套件-C-Sky Develop Kit(CDK)进行开发，其集成了代码编辑、在线仿真调试、模拟器调试、在线编程、性能分析等功能。

图表 19 CDK 开发平台

3.1.3.1 运行流程

1.系统初始化流程

连接电源，按下RE_PROG键后，首先通过板载的微控制器（MCU） $stm32-PM$

读取 SD 卡中的比特流文件$cfg.bit$并配置 FPGA.配置成功后，连接调试器并载入程序。

2.程序运行流程

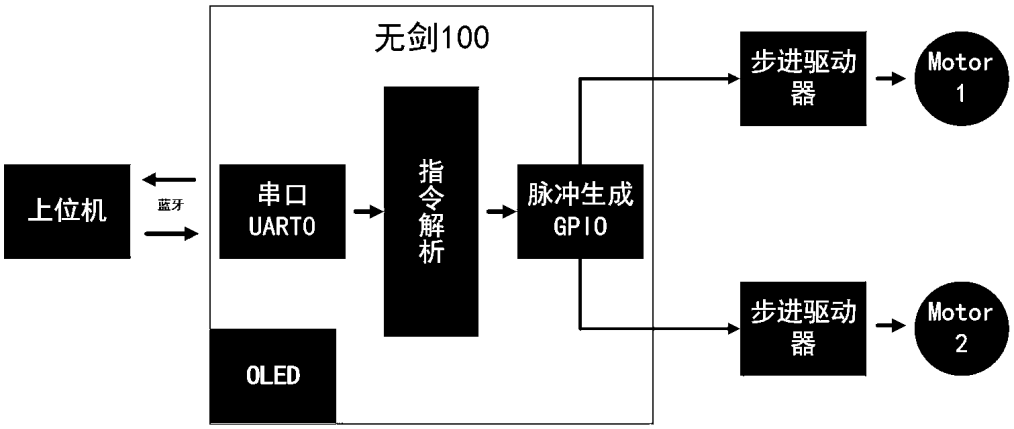
主程序$main.c$使用循环执行指令接收、处理和产生信号。

主程序中使用串口$UART0$通过轮询的方式读取来自蓝牙的指令，当无剑 100 的异步串列传输模块（UART）接收到来自上位机发送的指令后，首先解析指令翻译成电机片选序号及其旋转方向和脉冲数量。

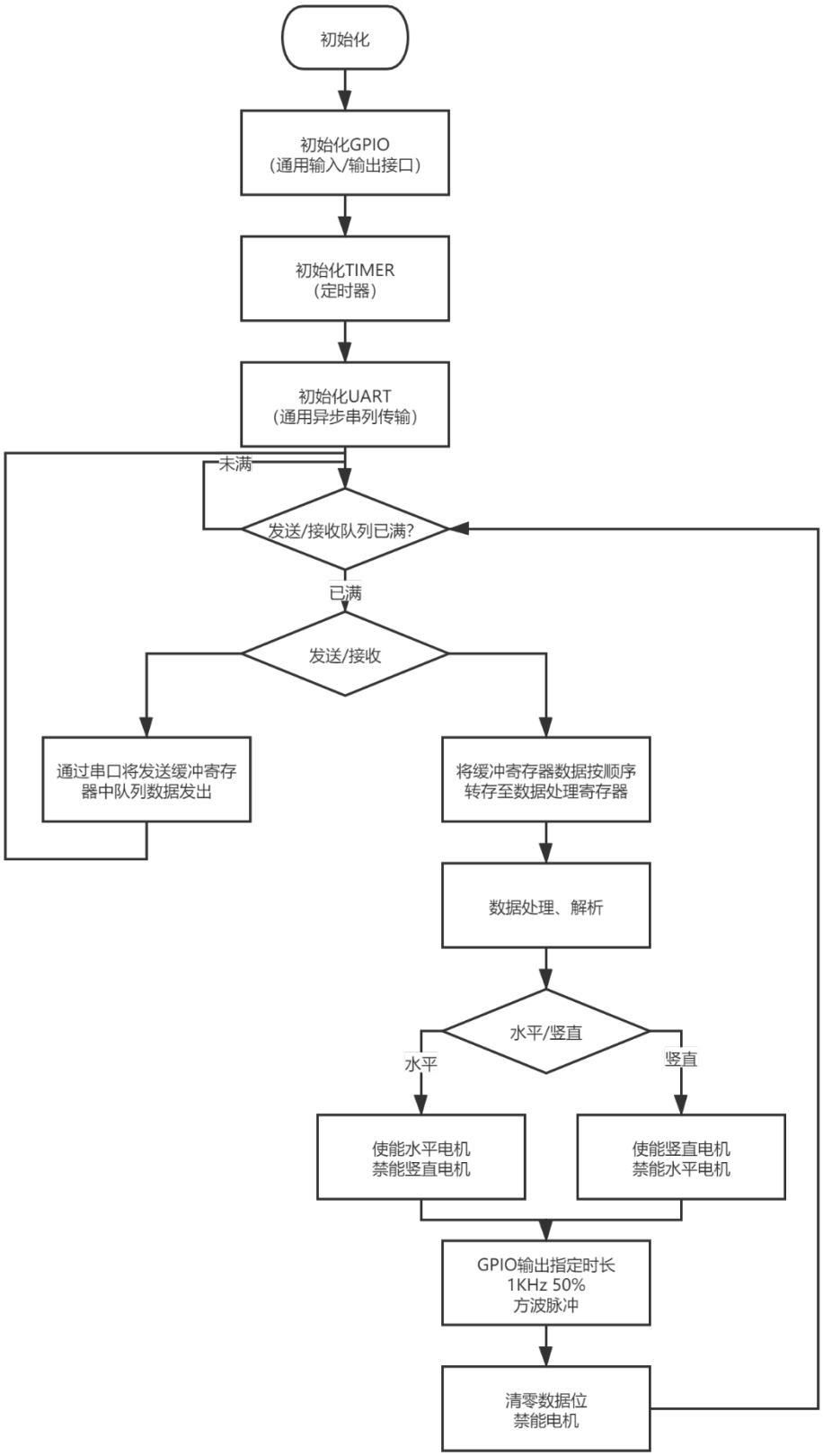
随后使用通用输入/输出接口(GPIO)模拟脉冲宽度调制（PWM）模块产生固定数量的脉冲和方向控制信号输入到步进驱动器中，达到控制目的。

最后使用 GPIO 模拟的串行外设接口（SPI）通信方式向 OLED 发送当前目标电机的位置，从而在 OLED 上显示出两个电机的绝对位置，可以起到验证指令解析是否正确和记录云台位置的作用。

系统框图如图表 20 所示，程序流程图如图表 21 所示。



图表 22 系统框图


初始化
初始化GPIO
（通用输入/输出接口）
初始化TIMER
（定时器）
初始化UART
（通用异步串列传输）
未满
发送/接收队列已满?
已满
发送/接收
通过串口将发送缓冲寄存器中队列数据发出
将缓冲寄存器数据按顺序转存至数据处理寄存器
数据处理、解析
水平/竖直
水平
竖直
使能水平电机
禁能竖直电机
使能竖直电机
禁能水平电机
GPIO输出指定时长
1KHz 50%
方波脉冲
清零数据位
禁能电机

图表 23 程序流程图

3.2　各模块工作方式

3.2.1　电机工作方式（驱动程序以及控制算法）

步进电机的驱动信号可以分为脉冲信号和方向控制信号。其中方向控制信号使用两个 GPIO 口产生，一个端口输出高电平，一个输出低电平，可使得电机向预设正方向旋转，则两个端口电平反转后即可使得电机向反方向旋转。步进电机按步长方式进行运动，每一步及对应一个脉冲。此电机步距角为 1.8 度，也即在细分数为 1 时，输入 1 个脉冲会驱动电机旋转 1.8 度。设细分数为 n，则每个脉冲驱动电机旋转角度为：

$$Angle = \frac{1.8}{n}(n = 1,2,3,\dots)$$

由于驱动器最高细分数为 25600，即理想情况下，驱动精确度上限输出一个脉冲驱动驱动电机选旋转$^{1.8°}/_{25600} = 0.00007°$。而脉冲的频率决定了步进电机的运行速度，相当于步进电机最终的响应速度。

设步进电机频率为f，细分数为n,则一个脉冲周期为$1/f$，步进电机旋转运动一周需要脉冲数为

$$N_{pulse} = \frac{n * 360}{1.8}$$

即，

$$N_{pulse} = 200n \quad (n = 1,2,3,\dots).$$

则所需时间为

$$t = \frac{200n}{f}s,$$

步进电机的运动速度w为

$$W = \frac{f}{200n}rps.$$

故根据实际应用调节频率即可满足不同运行速度的需要，在本设计中细分数取 1，即$n = 1$，输入的 PWM 频率为$f = 5kHz$,代入上式可得步进电机运行速度为

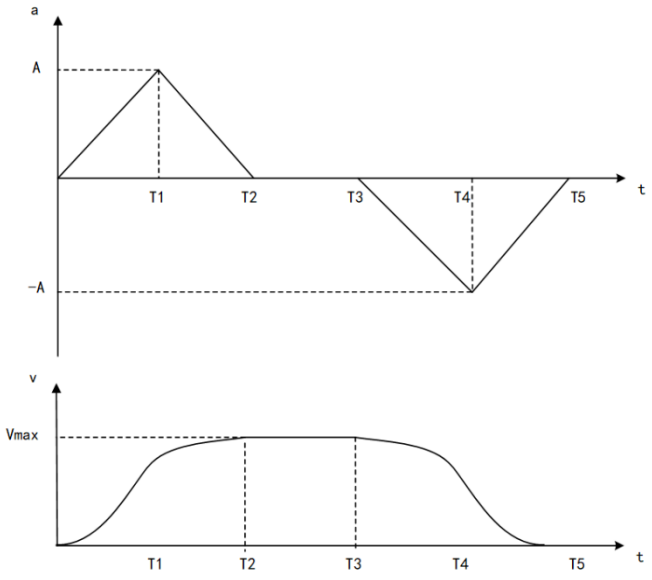
$$W = \frac{f}{200n} = \frac{5000}{200} = 25rps,$$

即每秒 25 转。

3.2.2　步进电机算法（S 曲线算法）

步进电机是一种把脉冲信号转换为角位移的机电元件，具有控制精度高，控制简单等特点，即使是开环控制也能获得较高的控制精度。但是由于固有原因，步进电机在启动时，由于此时需要的加速电流较大，线性度不够，会出现启动慢、启动丢步的现象；同理，在停止时也会出现过冲的现象，这是影响步进电机精度的一个较大原因，所以对步进电机启动和停止时的加速度，速度作合理的规划有很大的实际意义。

综合考虑后本方案选择使用 S 曲线加减速控制算法。S 曲线算法下的电机加速度和速度变化较为平稳且收敛较快，具有较好的效果。S 曲线算法如图表 24 所示，这里采用的是七段式 S 曲线，即在从启动到停止整个运动过程中电机的加速度可分为 7 个线性段。也可以看出来速度变化较为平稳。



图表 25 S 曲线算法示意图

设v为步进电机的稳定运动速度，也即 S 曲线中的最大速度，对应加速度为 0 的时段；最大加速度为A,最小加速度为$-A$, 最大速度为V_{max}, 加速度线性系数为k. 则整个运动过程可分为 5 段，如图表 26 所示。

图表 27 运功过程中加速度与速度的变化

	加速度$A(m/s^2)$	速度$v(m/s)$
1	kt	$\frac{1}{2}kt^2$
2	$A-kt$	$\frac{1}{2}At_1{}^2+\frac{1}{2}(2A-kt)\cdot(t-t_1)$
3	0	$V_{max}=\frac{1}{2}At_2$
4	$-kT+kt_3$	$V_{max}-\frac{1}{2}(t-t_3)(kt_3-kt)$
5	$kt-kt5$	$V_{max}-\frac{1}{2}A(t_4-t_3)-\frac{1}{2}(kT-kt_5+A)(T-t_4)$

步进电机启动加速度的大小和脉冲频率有着直接关系，所以可以假设脉冲频率和加速度存在线性相关关系，可以使用脉冲频率f代替加速度。经过实际带载测试和迭代求解，本方案中关于 S 曲线中各参数的详细配置为：

步进驱动器选择 6400 细分，计算可得 1 个脉冲对应的步进角为

$$\frac{1.8}{6400}=2.8125\times10^{-4}\circ$$

假设$F_{max}=5kHz$，则$t_1=50ms$，$t_2-t_1=50ms$．$t_4-t_3=50ms$，$t_5-t_4=50ms$。以$0\sim t_1$区间为例选择 100 个频率点，范围为$0\sim F_{max}$，时间步长为$50/100=0.5ms$，则脉冲频率步长为

$$\frac{F_{max}}{100}=\frac{5000}{100}=50.$$

3.2.3 蓝牙工作方式(传输和指令解析)

本方案设计中使用了无剑 100 中的 UART0 模块和蓝牙进行通信进而实现数据的无线发送和接收。上位机发送的是用 5 个字符即 10 个字节表示的自定义指令，指令格式定义如图表 28。

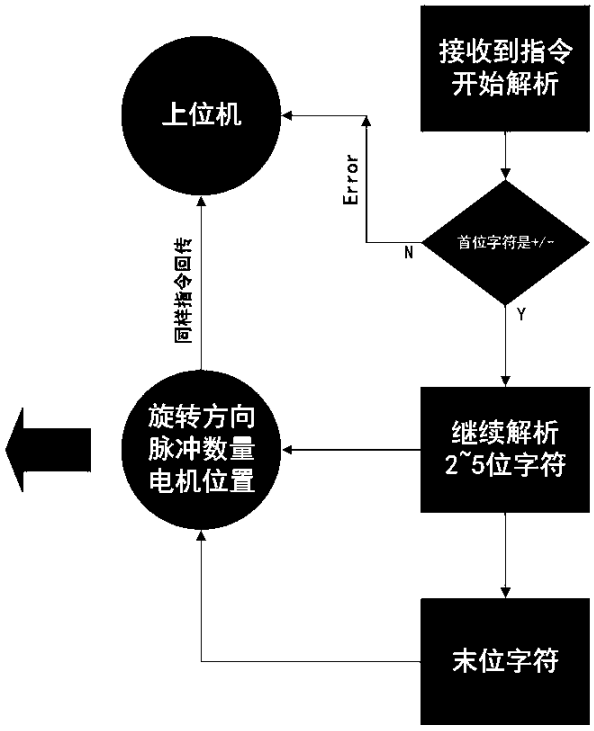
图表 29 蓝牙指令格式

指令（5 个字符）	意义

+/-（首字符）	+：定义方向顺时针旋转 -：定义方向逆时针旋转
xxxx(中间 4 个字符)	脉冲数量
1/2	1：top motor 2：bottom motor

当中央控制器接收到来自上位机发送的指令后首先进行校验，通过判断首位字符是是‘+’或者‘-’。如果不符合规范则回传上位机“error!”讯息，如果正确进行下一步解析。

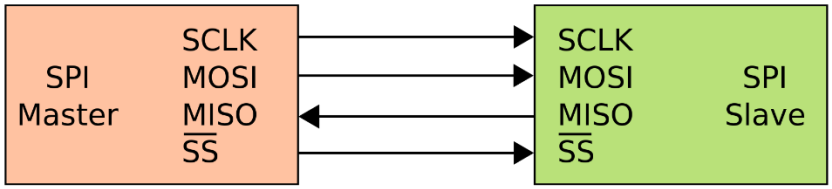
解析首先要把 4 个代表脉冲数的字符转化为对应的整数，结合首位字符，生成一个有符号数（signed integer），代表脉冲数和方向。之后解析末位字符，判断出目标电机的位置。在解析完整条指令后，回传给上位机同样的指令以及“success”消息表明解析成功。指令解析流程图如图表 30 所示。



图表 31 蓝牙指令解析流程图

3.2.4 有机发光二极管显示器（OLED）的工作方式（SPI 通讯）

有机发光二极管（OLED）显示屏使用串行外设接口（SPI）协议与主控制器进行通信作为人机交互界面，显示控制器内部变量的运行状态。无剑 100 SoC 虽然具有硬件 SPI 功能可以用来控制 OLED，但是整个系统不同硬件资源比如 GPIO、UART、SPI 等 I/O 端口的布局在本设计中无法统一。



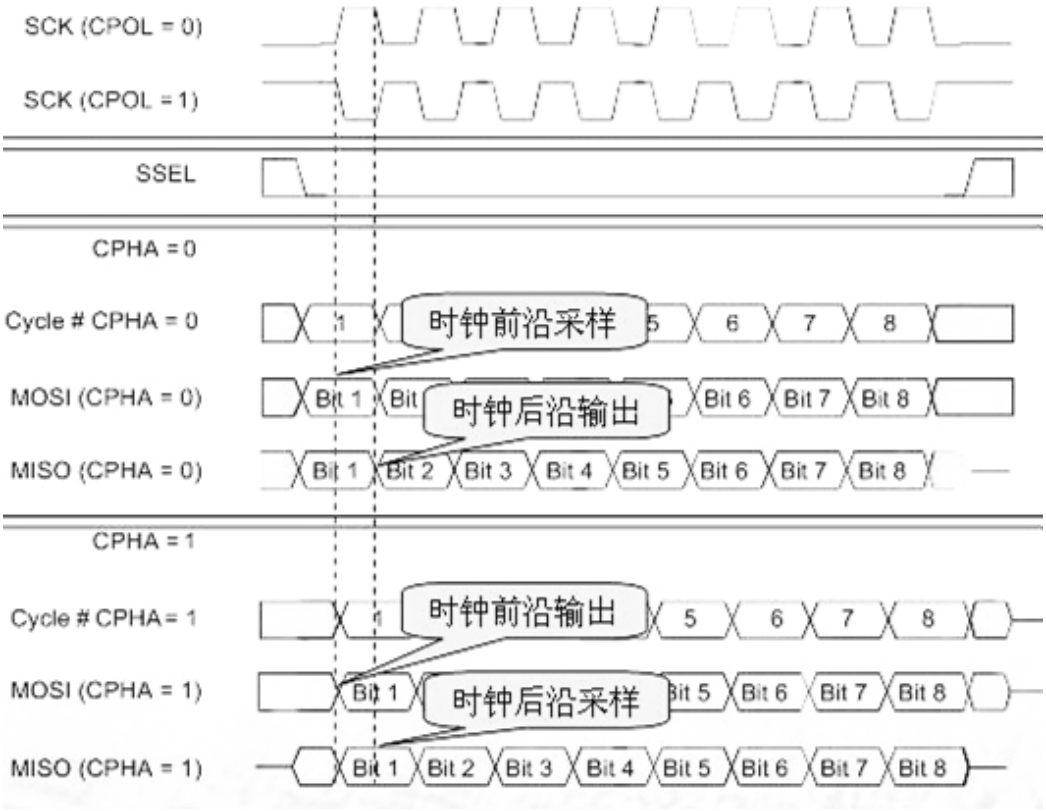
图表 32 SPI 主从机示意图

为了能够更集中利用板上资源，以及根据 SPI 协议的特性，最终本设计选择了使用 GPIO 模拟 SPI 的通信方式，SPI 主从机连接示意图如图表 33 所示。具体方法是根据 SPI 协议通信时时钟线和信号线的时序，用 GPIO 模拟通信端口和时钟线反转电平并进行合适的延时以模仿 SPI 的行为，可以实现使用任何 4 个通用端口即模拟串列时脉（SCLK）、主机输出从机输入信号（OSI）、主机输入从机输出信号（MISO）和片选信号（SS），即可达到 SPI 通信协议的要求。除了 SPI 通信所要求的 4 个信号线。除此之外，OLED 显示屏还需要一个复位信号（Reset），使用单独的 GPIO 即可，具体使用的 GPIO 端口定义如图表 34 所示。

图表 35 用于与 OLED 连接的 GPIO 端口定义

	SPI_CLK	SPI_NS	SPI_MISO	SPI_MOSI	RST
OLED	PA17	NC	PA9	PA12	PA11

SPI 协议的时序图如图表 36 所示，显示了不同模式下的信号时序图。本模拟 SPI 采用的是$CPOL = 0$且$CPHA = 0$的传输模式，即时钟相位为 0，时钟前沿数据采样，时钟后沿数据输出。



图表 37 SPI 通讯协议时序

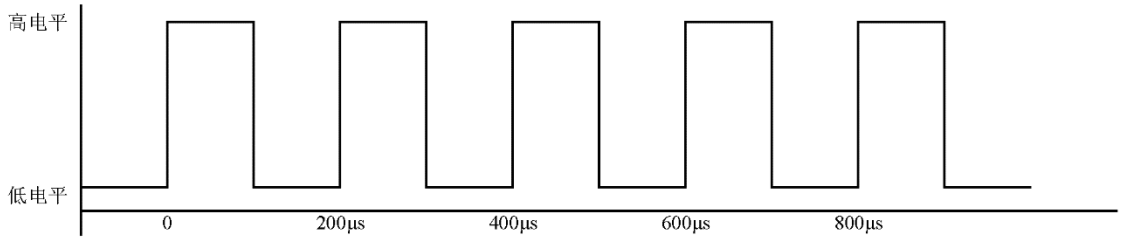
3.2.5　其他片上硬件资源的工作方式（GPIO、UART）

3.2.5.1　　通用输入/输出接口（GPIO）

可以看到，本设计中通用输入/输出接口（GPIO）是使用最多的资源，其中包括驱动器的脉冲宽度调制（PWM）信号、方向控制信号以及 OLED 的通信信号。本部分主要描述 PWM 信号的产生，其余部分在前文中已有详述。

产生 PWM 波形有三种方案：一是使用内置的 PWM 发生器，比如无剑 100SoC 提供共 12 个 PWM 通道，只需要配置好相应的寄存器即可产生需要的 PWM 波，可以应对如对频率和占空比等比较高且需长时间保持脉冲的情况；二是使用定时器中断产生电平翻转信号，同样有频率高的特性，但使用定时器中断产生的 PWM 波形的脉冲宽度通常不可调；三是可以使用通用输入/输出接口（GPIO）产生 PWM 波形，这种方式的优势在于脉冲数可控、频率与占空比即时调节，但受限于 GPIO 的主线频率（多为 50kHz 以下）无法产生高频率的脉冲。由于本设计中的步进电机需

要固定脉冲数量，如若使用内置硬件 PWM 则需要进行边沿判断来提取脉冲数，使得程序设计冗杂，效率降低。因此本方案选用上述的方案三，无剑 100 SoC 主频可达$20MHz$ ，故 GPIO 的电平可以依赖高速总线实现快速反转的功能以满足步进驱动所需频率，其产生的脉冲示意图如图表 38 所示。

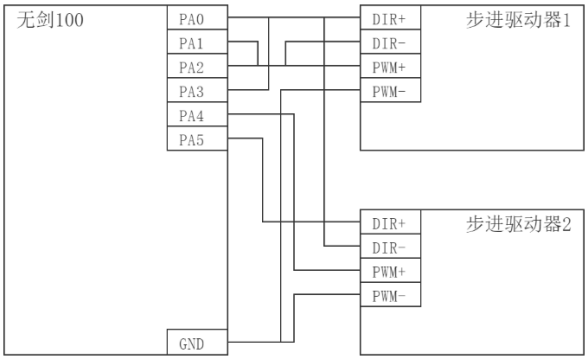


图表 39 脉冲示意图

具体而言，只需通过电平反转和延时保持即可实现 PWM 的生成。延时时间决定脉冲频率，通过有限次数的循环反转和精确延时以决定脉冲数量。GPIO 持续高电平$100\mu s$，之后反转持续低电平$100\mu s$作为一个脉冲，即脉冲频率为$5kHz$。GPIO 端口定义如图表 40 所示，通用输入/输出接口与步进驱动器的连线如图表 41 所示。

图表 42 用于与步进电机驱动器连接的 GPIO 端口定义

	Motor1	Motor2
PWM+	PA1	PA4
PWM-	GND	GND
DIR+	PA3	PA5
DIR-	PA2	PA0



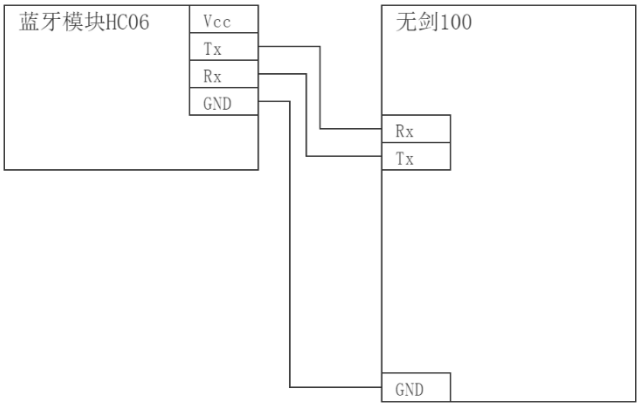
图表 43 通用输入/输出接口与步进驱动器的连线

3.2.5.2 异步串列传输（UART）

无剑 100 SoC 的同步串列传输模块（UART）可以进行同步串行发送接收数据。本设计使用端口 UART0 外接嘉源电子蓝牙串口模块 HC06（连接方式如图表 44 所示）与上位机（手机、电脑等设备）通信，其配置参数如图表 45 所示。

图表 46 异步串列传输配置表

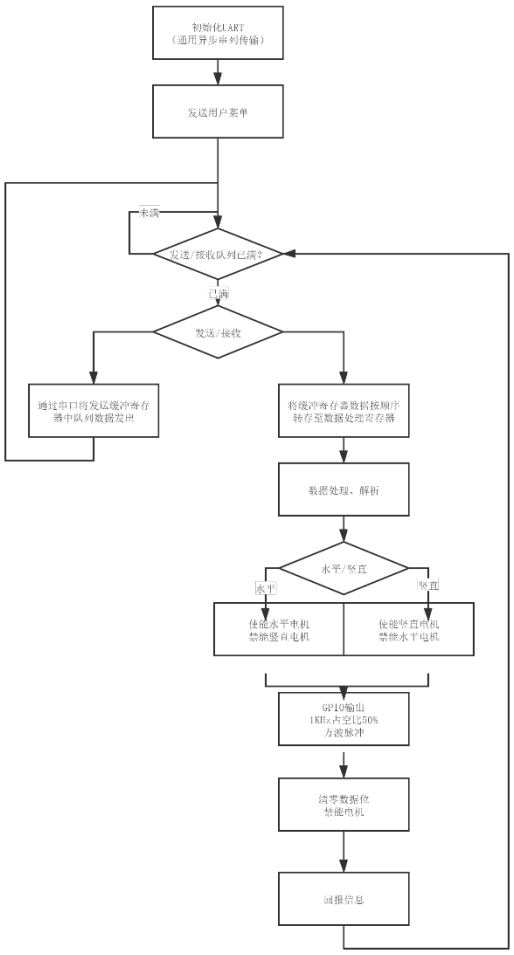
波特率（Baud Rate）	115200
传输模式(Mode)	异步传输
校验位(Parity)	无
停止位(Stop Bit)	1 位
数据位(Data Bit)	8 位



图表 47 蓝牙模块 HC06 与无剑 100SoC 的连接方式

程序设计方面，数据采用轮询方式（polling）接收，即不断检测缓冲寄存器（Rx buffer）是否已满。当主控接收到数据后，先会储存在缓冲寄存器中，将缓冲寄存器已满信号置位后通过程序读取缓冲寄存器中数据进行操作。

上位机在连接蓝牙后，可以看到串口输出首界面如图表 48 所示，包括指令解读和使用方式。用户发出指令后，程序将在主循环中进行指令读取、接收和回报，流程图如图表 49 所示。



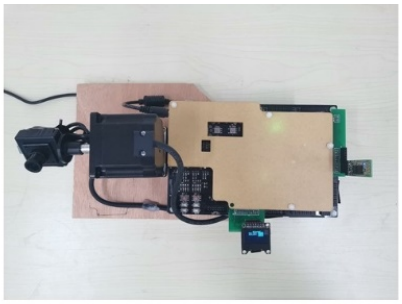
图表 50 蓝牙控制流程图

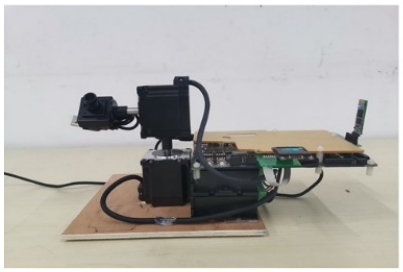
图表 51 蓝牙串口界面（英文）

3.3 物理结构

济小台的物理结构如图表 52 所示。其中包含两个步进电机，下方电机作为主电机，负责水平方向上的维度控制，另外一个电机在竖直方向上与主电机连接，负

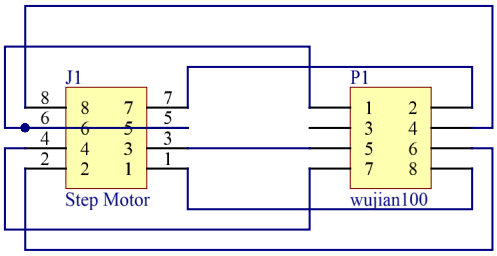
责竖直方向上的维度控制，两者结合可以做到各 360 度的环视功能。两个步进电机和驱动器固定在一块底板上，成为独立的运动部分；FPGA 开发板作为单独的控制部分。驱动器和开发板的连接使用自制 PCB 进行连接，使得连接更加可靠，整个系统运作更加稳定，连接板 PCB 图如图表 53 与图表 54 所示。



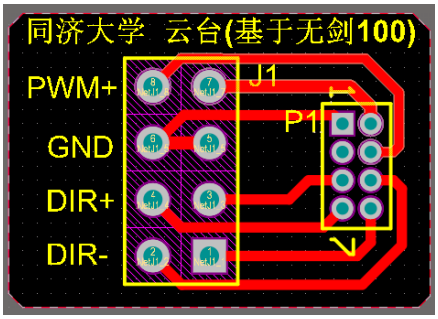


图表 55 设备三视图

驱动器和开发板的连接使用自制 PCB 进行连接，使得连接更加可靠，整个系统运作更加稳定，PCB 图如图表三与图表四所示。



图表 56 连接板原理图



图表 57 连接板 PCB

4 技术创新点

4.1 步进电机在安防云台的应用

目前云台多是采用舵机进行各个自由度的方向控制，舵机一般成本较低，外设简单，结构紧凑体积小，但是由于舵机是恒扭矩，功率较小，使得在速度较低，负

载较大的运动场景中无法较好的应用。将步进电机引入安防云台的设计中，提高了传统安防云台的工作效率、降低安防云台的损修率、提高系统整体的鲁棒性、降低安防云台的功耗，使得其依赖电池供电工作成为可能。步进电机的功率可以是舵机的几倍之多，可以适用于负载需求较大的场合。除此之外，步进电机的精确度在高精度步进驱动器的配合下可以高达10^{-5}°，而且不同于直流电机，其每一执行步的误差不会积累；步进电机的启动和停止响应都极快，不需要复杂的控制算法。

本方案选用步进电机设计云台既可以实现多种负载大小的应用场景中，比如负载较小的监控云台，或视觉定位的底层驱动系统，以及负载较大的机器人头部模块等。与此同时，还能达到极高的精确度和响应速度。

4.2　基于 wujian100 SoC 的类嵌入式操作系统的程序设计

类嵌入式操作系统，即在程序设计中，采用类似于嵌入式操作系统的程序编写。不同于传统的嵌入式程序设计，将大部分程序置于主函数的循环中，而是将主要的功能性程序置于定时器中断函数中。类嵌入式操作系统的设计，极大提高了嵌入式系统的运行效率、准确度，降低了系统资源的使用率以及片上功耗。

4.3　系统资源的极小化运用

在整个设计中仅使用到与步进电机控制相关的 3 个通用输入/输出接口以及 2 个定时器中断与 1 个异步串列传输通讯接口。济小台认同少即是多的设计理念，不以使用资源多、功能复杂冗余为目的，而是精准于使用尽量少的接口、简单的程序实现最核心实用的功能，从而可以极大的节省算力，为其他应用外设可以分配出更充足的开发空间，如可以加入图像采集、视觉处理算法等，进而实现云台追踪。

除此之外，整个云台系统使用的 12V 直流电源，可以脱离高电压、高电流的工业场景，直接使用电池供电。系统实测运行电流不超过 1A，功率低于大部分家用产品，也适用于移动场景。

5 团队介绍

李珈毅，电子科学与技术系 2016 级本科生，研发组长。主要负责产品设计、研发，技术方案拟写，对设备、尤其是针对于 wujian100 SoC 的研究等。

李伟博，电子科学与技术系 2016 级本科生，2020 级博士研究生，技术骨干。主要负责产品研发、技术细节优化等。

翁锦煜，电子科学与技术系 2017 级本科生，技术骨干。主要负责产品设计、技术细节优化等。

6 后续工作

后续研发的重点如下：

- 将济小台通过蓝牙 mesh 组网将其与其他家用智能设备联网，组成家用智能安防网络；
- 利用 wujian100 SoC 以及 FPGA 的其余资源实现运动监控的功能；
- 使用 WLAN 通讯模块将视频实时上传至家中的 NSA 设备[2]。

7 项目心得体会

无剑 100 作为一款开源的基于 RISC-V 指令集的 SoC 平台，承载了很多国内科研人员的重望，也不负众望，可以说是一款非常优秀的嵌入式芯片。其功耗低、占用资源少、轻量、开发简便的特点使得无剑 100 可以作为一款初学者都易上手的 SoC，相比于 Zynq-7000 SoC 的复杂繁琐、编译器的种种非人类设计，无剑 100 甚至可以作为高校的教学用的 FPGA SoC。

在使用的过程中，本着方便日后更多研究者以及同学能够更快速便捷地使用无剑平台，本团队编写了英文的功能使用手册，发布在 jiayi’s Blog 上，共有八个详细的使用教程，包括使用 windows 的比特流生成、CDK 开发平台与 wujian100 SDK 简介、新项目创建、快速开始项目、通用输入/输出接口的使用、同/异步串

[2] *注：由于近期曝光的某度以及某品牌水滴摄像头的监控门事件，为保证用户的隐私安全，济小台没有将视频信号云存储/联网的计划。*

列传输的使用、定时器的使用、中断向量控制器的介绍以及一个问题汇总。

作为一个电子相关专业的本科生团队，深知中国芯片行业目前举步维艰的困难处境：前有高通苹果在集成芯片领域绞杀，英特尔、AMD，后有 ARM 的威胁勒索，中间还夹着可能断供生产的台积电。就连嵌入式芯片，都被 ST 微电子、恩智浦、德州仪器等公司围困，即便是走在最前列的海思半导体，也没能逃过 ARM 的魔掌。

所以中国芯片，尤其是中国的嵌入式微处理器要在 RISC-V 上杀出一条血路。很庆幸能够参与到平头哥半导体冠名的该项赛事，让我们得以了解中国公司目前在 RISC-V 指令集上走出的每一步。这是芯片历史上的很小一步，却是中国芯片发展里程上的重大成就。希望平头哥半导体能够在无剑的基础上流片生产出自主设计研发的 MCU，让仍在大学的我们也能有机会学习到中国自主的嵌入式芯片与架构。