虚谷号通用输入输出控制引脚使用文档

基于 Python 环境的 xugu.py 库

Ver 1.1

文件状态： [ ] 正在修改 [ ] 正在发布	当前版本：	V1.1
	作者：	Adolph
	完成日期：	2019.3.20
	审核：	
	完成日期：	

版本历史

版本号	作者	修改日期	修改说明	审核	备注
V1.1	Adolph	2019.3.20	修改范例		

目录

第一部分、虚谷号 GPIO（通用输入输出）简介

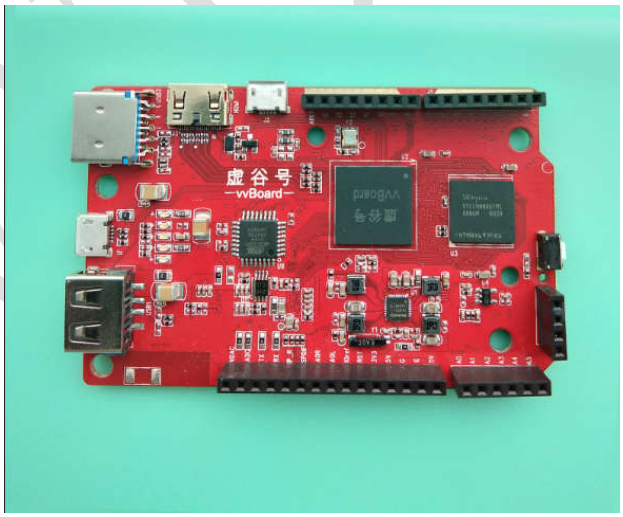
一、 虚谷号简介

虚谷号是一个面向人工智能教学和 Python 编程学习的中国原创开源硬件平台，板内集成高性能 4 核 64 位处理器和通用单片机，内置多功能扩展接口，支持多种常用通信接口，为人工智能和 Python 编程教学提供完整课程资源包。

虚谷号运行完整的 Linux 系统，支持 HDMI 音视频接口，1 个 USB3.0 接口和 2 个 USB2.0 接口，其中 1 个 USB2.0 接口支持 OTG 功能（OTG 是 On-The-Go 的缩写，可用于不同设备，特别是移动设备间的联接和数据交换），自带蓝牙和 Wi-Fi 功能，接上鼠标键盘和显示器，就是一台完整的电脑。

虚谷号板载了高性能 RISC 通用 8 位单片机，用于实时信息采集与控制，拥有 14 个数字输入输出引脚，编号为 D0~D13，其中 D0 和 D1 与串口通信复用，编号为 D3、D5、D7、D9、D10、D11 的 6 个引脚支持 PWM 脉宽调制输出（PWM，既 Pulse Width Modulation，广泛应用于测量、通信、功率控制与变换等领域），在虚谷号上以下划“~”标识，可用于实现模拟输出功能；拥有 6 个数字输入引脚，编号为 A0~A5，也可兼做数字输入输出引脚，这时的编号为 D14~D19。

虚谷号引脚资源和尺寸完全兼容 Arduino（可以看成是一块 Arduino UNO 板），虚谷号通过操作系统内置 Arduino IDE 实现与 Arduino 代码兼容。



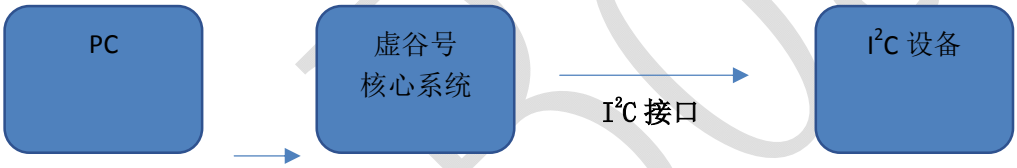
二、 GPIO 功能简介

虚谷号提供了多种方式，实现 GPIO 功能。

方式 1：虚谷号通过串口和板载的 Arduino UNO 连接，可以用任何一款 Arduino 的编程工具，用标准的 Arduino 代码进行编写，控制 Arduino UNO 的所有引脚。

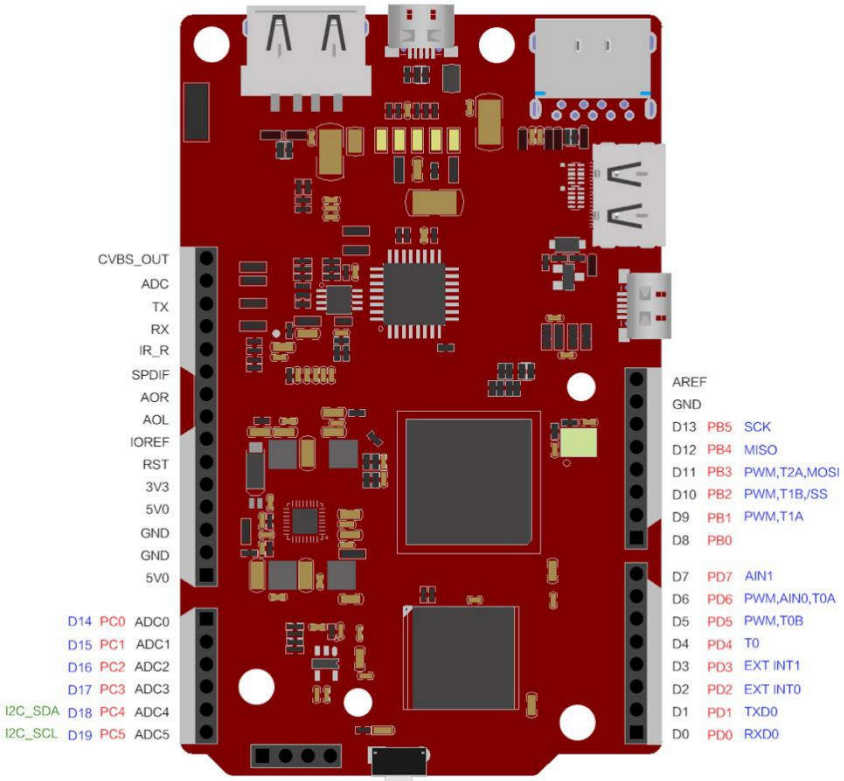
方式 2：虚谷号给 Arduino UNO 烧写标准的 Firmata 协议，通过串口命令进行控制 Arduino 引脚，为降低初学者的开发门槛，虚谷号提供了和 MicroPython 语法完全兼容的“xugu.py”库，供 Python 编程教学时调用。

方式 3：虚谷号的主芯片引出 I^2C 接口，可以通过这一接口来实现对外部设备的控制。



第三种方式中，虚谷号主芯片的 I^2C 接口可以和方式 1、方式 2 结合。相对来说，方式 2 的开发难度最低，能满足常用的大部分需求，只需要有 Python 编程基础即可。

三、 引脚说明图示



第二部分、xugu 库简介

xugu 库中有 Pin、Servo、I2C、SerialMgt、LED、XuguLog 等类。

一、 控制 I/O 引脚：Pin 类

Pin 类用于控制 I/O 引脚，具有设置引脚模式（IN，OUT）的属性和读写电平状态的方法。

1、 构建

Pin(pin_num, pin_model)

pin_num 传入引脚标号，可以直接传入虚谷板上的引脚编号，例如 D3 或者 A5，也可以直接传入 13 或者 19 这样的数字。

pin_model 为引脚模式，Pin.IN 是输入模式，Pin.OUT 是输出模式。

2、 方法

Pin.read_digital ()

返回该 IO 引脚电平值，1 代表高电平，0 代表低电平。该方法在输入模式有效。

Pin.read_analog ()

返回 IO 引脚的模拟值，数据范围在 0 和 1023 之间。该方法在输入模式有效。

Pin.write_digital (value)

给引脚设置电平值。value 指要设置的电平值，1 代表高电平，0 代表低电平。该方法在输出模式有效。

Pin.write_analog (value)

给引脚设置模拟值。value 指要设置的模拟值，数据范围在 0 和 1023 之间。该方法在输出模式有效。

二、 舵机控制：Servo 类

该类用于控制舵机转到指定角度。

1、 构建

Servo(pin_num)

pin_num 引脚标号，可以直接传入虚谷板上的引脚编号，例如 D3 或者 A5，也可以直接传入 13 或者 19 这样的数字。

2、 方法

Servo. write_angle (value)

让舵机转动到指定角度，Value 指角度。每种舵机的最大转动角度不一样，需要参考舵机说明书。

三、 读写 I2C 设备：I2C 类

该类用于读写 I2C 从设备。

注：该类不能用于读写虚谷号主芯片的 I2C 总线。

1、 构建

I2C(time=0)

time 指 I2C 总线连续读写的间隔时间，单位是毫秒（ms），默认值是 0。

2、 方法

I2C .readfrom(address, register, read_byte)

读取 I2C 设备。address 为 I2C 从设备的地址，register 为从设备的寄存器，read_byte 为一次读取的字节数量。

I2C .writeto(address, args)

向 I2C 设备中写入内容。address 为 I2C 从设备的地址，args 是要发送到设备的可变字节数，作为列表传入。

四、 串口对象：SerialMgt 类

该类用于虚谷号和 PC 之间的串口通信。

1、 构建

SerialMgt(port, baudrate)

port 指虚谷号连接 pc 的串口号，baudrate 为串口波特率；

注：当不设置串口号和波特率时，虚谷库默认使用/dev/ttyGS0 串口，波特率为 115200。

2、 方法

SerialMgt.read(bytes)

从串口中读取数据，bytes 为读取的字节数，默认为 100。

SerialMgt.write(data)

向串口中写入数据，data 为写入的数据，类型为 String。

五、 LED 类

该类用于 LED 的简易控制。

1、 构建

LED(pin_num)

Pin_num 为数字引脚编号，范围 0~19。

注：虚谷号已经在第 13 号引脚内置了 LED。

2、 方法

high()

给引脚一个高电位，只有在输入模式有效，当该引脚接入 LED 灯的时候，灯会点亮。

low()

给引脚一个低电位，只有在输入模式有效，当该引脚接入 LED 灯的时候，灯会熄灭。

on()

等价与 high()。

off()

等价与 low()。

六、 XuguLog：日志输出类

该类用于日志输出。程序运行过程中，会将日志信息追加到日志文件中。

1、 构建

XuguLog(filename)

初始化该类的时候，会自动生成一个名为 filename 的日志文件，后缀为.log，生成的文件与运行的 python 程序在同一个目录下。

2、 方法

XuguLog.write(value)

将日志信息写入到日志文件中，value 为要写入的内容，类型为 String。

第三部分、常见范例代码(代码存放在/home/scope/vvBoard/Python/01.example/04.开源硬件路径下)

一、 虚谷闪一闪

代码说明：让虚谷号自带的 LED（接在 13 号引脚，即 D13）闪烁

代码范例 1：使用 LED 类(xugu-blink.py)

```
import time    # 导入 time 模块
from xugu import LED    # 从 xugu 库中导入 LED 类
led = LED(13)    # 初始化 LED 类
while True:    # 用循环实现持续地开灯关灯，到达闪烁的效果
    led.on()    # 点亮连接 13 号引脚的 LED 灯
    time.sleep(1) # 持续 1 秒
    led.off()    # 关闭 LED 灯
    time.sleep(1)    # 持续 1 秒
```

代码范例 2：使用 Pin 类(led_pin.py)

```
import time    # 导入 time 模块
from xugu import Pin    # 从 xugu 库中导入 Pin 类
led = Pin(13, Pin.OUT)    # 初始化 Pin 类
 # 等价的写法：led = Pin("D13", pin.OUT)
while True:    ##用循环实现持续地开灯关灯，到达闪烁的效果
    led.write_digital(1)    # 点亮连接 13 号引脚的 LED 灯
    time.sleep(1) # 持续 1 秒
    led.write_digital(0)    # 关闭 LED 灯
    time.sleep(1)    # 持续 1 秒
```

二、 数字输出

代码说明：将 D10 引脚置于高电平

代码范例 1：输出高电平(pin_out_high.py)

```
from xugu import Pin    # 从 xugu 库中导入 Pin 类
p = Pin(10, Pin.OUT) # 初始化 Pin 类，将 10 号数字引脚设置为输出模式
#等价的写法：  p = Pin("D10", Pin.OUT)
p.write_digital(1)   # 设置 10 号引脚为高电平
```

代码说明：将 D10 引脚置于低电平

代码范例 2：输出低电平(pin_out_low.py)

```
from xugu import Pin    # 从 xugu 库中导入 Pin 类
p = Pin(10, Pin.OUT) # 初始化 Pin 类，将 10 号数字引脚设置为输出模式
# p = Pin("D10", Pin.OUT)
p.write_digital(0)   # 设置 10 号引脚为低电平
```

三、 模拟输出

代码说明：将 D10 引脚设置为模拟输出（采用 PWM 脉宽调制实现），参数为 0-255。

代码范例：模拟输出（analog_write.py）

```
from xugu import Pin    # 从 xugu 库中导入 Pin 类
p = Pin(10, Pin.OUT)   # 初始化 10 号引脚设置为输出模式
p.write_analog(128)   # 给引脚设置模拟值 128
```

注：虚谷号支持的模拟输出（PWM 方式）的引脚共有 6 个： 3、5、6、9、10、11。

四、 数字输入

代码说明：读取 D4 引脚的高低电平状态。

代码范例：数字输入（digital_input_master.py）

```
from xugu import Pin, SerialMgt # 从 xugu 库中导入 Pin、SerialMgt 类
import time # 导入时间模块
p = Pin(4, Pin.IN) # 初始化 4 号引脚,设置为数字输入模式
ser = SerialMgt() #初始化串口
while True:
    value=p.read_digital() # 读取 4 号引脚电平信息,赋值给 value
    print(value) # 将 value 的值打印到终端上
    ser.write(str(value).encode()) # 将 value 的值写入到串口
    time.sleep(0.1) # 持续 100ms
```

五、 模拟输入

代码说明：读取 A0 引脚的电压值

代码范例：模拟输入（analog_input.py）

```
from xugu import Pin, SerialMgt # 从 xugu 库中导入 Pin、SerialMgt 类
p = Pin("A0", Pin.ANALOG) # 初始化 A0 引脚,设置为输入模式
ser = SerialMgt() #初始化串口
value = p.read_analog() #读取 A0 引脚的电压值
ser.write(str(value).encode()) # 将 value 的值写入串口
```

六、 舵机控制

代码说明：控制 D4 引脚上的舵机旋转到 150 角度

代码范例：舵机控制（servo.py）

```
from xugu import Servo #从 xugu 库中导入 Servo 类
servo = Servo(4) # 初始化 4 号引脚,并连接到舵机
servo. write_angle(150) # 将舵机旋转 150°
```

七、 串口通讯

代码说明：读取模拟传感器数值并通过 PC 串口输出（将 Arduino 的串口通信内容转发到 PC 的串

口）

代码范例：虚谷端串口通信（read_analog_20_times.py）

```
# 在虚谷端运行
from xugu import Pin, XuguLog, SerialMgt # 从 xugu 库中导入 Pin、XuguLog、SerialMgt 类
import time # 导入时间模块
test = 20 # 计数
p = Pin("A0", Pin.ANALOG) # 初始化 A0 引脚,设置为输入模式
ser = SerialMgt() #初始化串口
f=XuguLog('read_analog_20_times.log') # 当不指定路径时，log 文件保存在与 digital_input.py 同级路径下
#f=XuguLog('/home/scope/analog_input.log') # 初始化日志对象，当指定路径时 log 文件保存在指定路径下
while test > 0:
    value = p.read_analog() #读取 A0 引脚的电压值
    f. write(str(value)) # 将变量 value 写入到日志文件中
    print(value) # 将 value 的值打印到终端
    ser.write(str(value).encode()) # 将 value 的值写入串口
    test -= 1 # 计数自减 1
    time.sleep(0.5) # 持续 500ms
```

在 PC 端建议通过串口工具打开对应的串口号（COM6）即可查看虚谷板写入的内容

```
# 在 PC 端还可以通过运行以下程序，获得数据
import serial #导入 serial 类
serialPort = "COM6"   # 串口号（假设 PC 和虚谷连接的是 COM6 端口）
baudRate = 115200   # 波特率
ser = serial.Serial(serialPort, baudRate, timeout=0.5) # 初始化串口
while ser.isOpen(): # 判断串口是否打开
    data = '' # 定义 data 变量
    while ser.inWaiting() > 0: # 判断接收到的数据长度
        data = ser.read(100) #读取该串口的值并赋值给 data
    if data != '': # 判断 data 是否为空
        print(data) # 将 data 打印到终端
```

注：虚谷的串口默认波特率是 115200。

八、 读取 I2C 设备

代码说明：读取 I2C 设备

代码范例：

```
from xugu import I2C   # 从 xugu 库中导入 I2C 类
i2c = I2C()   # 初始化 i2c 设备
data = i2c.readfrom(0x42, 0x10, 5)   # 从 0x42 从设备的 0x10 寄存器连续读取 5 个数据
```

九、 写入 I2C 设备

代码说明：写入 I2C 设备

代码范例：

```
from xugu import I2C # 从 xugu 库中导入 I2C 类
i2c = I2C()   # 初始化 i2c 设备
i2c.writeto(0x42, b‘xy’)   # 向 0x42 从设备写入数据‘xy’
```