使用昉·星光 2的GPIO制造蜂鸣声

Python语言版本

应用说明

版本：1.2

日期：2025/08/07

Doc ID: VisionFive2-ANCH-001

法律声明

阅读本文件前的重要法律告知。

版权注释

版权 ©上海赛昉半导体科技有限公司，2025。版权所有。

本文档中的说明均基于“视为正确”提供，可能包含部分错误。内容可能因产品开发而定期更新或修订。上海赛昉半导体科技有限公司（以下简称“赛昉科技”）保留对本协议中的任何内容进行更改的权利，恕不另行通知。

赛昉科技明确否认任何形式的担保、解释和条件，无论是明示的还是默示的，包括但不限于适销性、特定用途适用性和非侵权的担保或条件。

赛昉科技无需承担因应用或使用任何产品或电路而产生的任何责任，并明确表示无需承担任何及所有连带责任，包括但不限于间接、偶然、特殊、惩戒性或由此造成的损害。

本文件中的所有材料受版权保护，为赛昉科技所有。不得以任何方式修改、编辑或断章取义本文件中的说明，本文件或其任何部分仅限用于内部使用或教育培训。使用文件中包含的说明，所产生的风险由您自行承担。赛昉科技授权复制本文件，前提是您保留原始材料中包含的所有版权声明和其他相关声明，并严格遵守此类条款。本版权许可不构成对产品或服务的许可。

联系我们：

地址： 中国（上海）自由贸易试验区盛夏路61弄张润大厦2号电梯楼层5层（实际楼层4层）06室

网站： http://www.starfivetech.com

邮箱：sales@starfivetech.com （销售） support@starfivetech.com（支持）

前言

关于本指南和技术支持信息

关于本手册

本应用说明提供使用昉·星光 2的GPIO Pin，通过Python示例程序发出蜂鸣声的步骤。

修订历史

表 0-1 修订历史

版本	发布说明	修订
1.2	2025/08/07	更新了以下章节： • 运行环境要求 (第 8页) • 准备软件 (第 9页) • 执行演示代码 (第 12页)
1.1	2023/06/08	• 在40-Pin GPIO Header定义 (第 7页)增加注释。 • 在准备软件 (第 9页)中更新安装方式。 • 新增资源下载 (第 15页)和立即购买 (第 16页)章节。
1.0	2022/12/15	首次发布。

注释和注意事项

本指南中可能会出现以下注释和注意事项：

- **提示：**
  建议如何在某个主题或步骤中应用信息。
- **注：**
  解释某个特例或阐释一个重要的点。
- **重要：**
  指出与某个主题或步骤有关的重要信息。
- **警告：**
  表明某个操作或步骤可能会导致数据丢失、安全问题或性能问题。
- **警告：**
  表明某个操作或步骤可能导致物理伤害或硬件损坏。

目录

表格清单 5
插图清单 6
法律声明 2
前言 3
1. 产品简介 7
1.1. 40-Pin GPIO Header定义 7
2. 准备 8
2.1. 运行环境要求 8
2.2. 准备硬件 8
2.2.1. 连接硬件 8
2.3. 准备软件 9
3. 执行演示代码 12
4. 演示源代码 13
5. 资源下载 15
6. 立即购买 16

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

表格清单

表 0-1 修订历史 3
表 2-1 硬件准备 8
表 2-2 将蜂鸣器连接到40-Pin GPIO Header上 8

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

插图清单

图 1-1 40-Pin GPIO Header定义 7
图 2-1 将蜂鸣器连接到40-Pin GPIO Header上 9

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

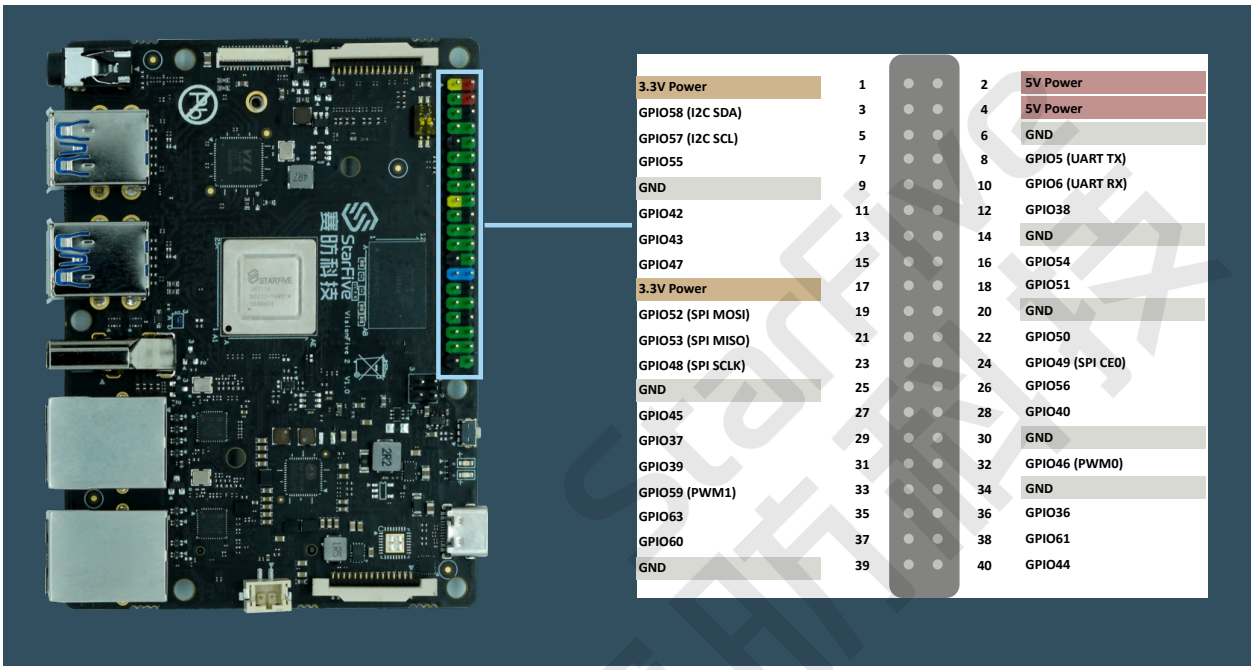
1. 产品简介

本应用说明提供使用昉·星光 2的GPIO Pin，通过Python示例程序发出蜂鸣声的步骤。

1.1. 40-Pin GPIO Header定义

下图显示了40-pin GPIO Header的位置：

图 1-1 40-Pin GPIO Header定义



注：
功能复用pin脚已初始化，不可作为通用GPIO使用。

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

2. 准备

在执行演示程序之前，务必确认已准备好以下项目：

2.1. 运行环境要求

该演示运行环境要求如下：

- Linux内核版本：Linux 6.6
- 操作系统：Debian 13
- 硬件版本：昉·星光 2
- SoC：昉·惊鸿-7110

2.2. 准备硬件

在执行演示程序之前，请务必准备以下硬件：

表 2-1 硬件准备

类型	M/O*	项目	注释
通用	M	昉·星光 2 单板计算机	-
通用	M	• 容量不低于32 GB的Micro-SD卡 • Micro-SD卡读卡器 • 计算机（Windows/Mac OS/Linux） • USB转串口转换器（3.3 V I/O，带线） • 以太网电缆 • 电源适配器（5 V/ 3 A） • USB Type-C数据线	上述项目用于将Debian OS烧录到Micro-SD上。
GPIO演示(Buzzer)	M	一个无源蜂鸣器	-

注：
*: M：必须。O：可选

2.2.1. 连接硬件

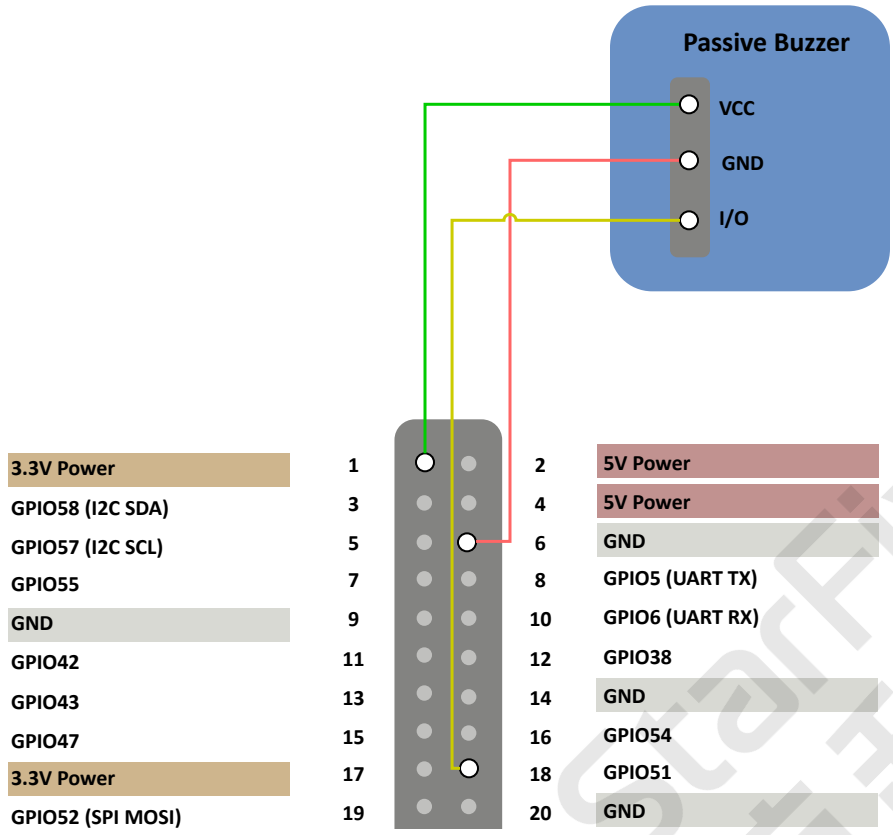
以下表格和图片描述了如何将蜂鸣器连接到40-Pin GPIO Header上：

表 2-2 将蜂鸣器连接到40-Pin GPIO Header上

无源蜂鸣器	40-Pin GPIO Header	
	Pin Number	Pin Name
VCC	1	3.3V 电压
GND	6	GND
I/O	18	GPIO51

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

图 2-1 将蜂鸣器连接到40-Pin GPIO Header上



2.3. 准备软件

确认按照以下步骤进行操作:

注:
该Python应用`VisionFive.gpio`适用于昉·星光单板计算机、昉·星光 2和昉·惊鸿-7110 EVB。

1. 按照《昉·星光 2单板计算机快速参考手册》中的"将*OS*烧录到*Micro-SD*"章节，将Debian OS烧录到Micro-SD卡上。
2. 登录Debian并确保昉·星光 2已联网。有关详细说明，请参阅《昉·星光 2单板计算机快速参考手册》中"通过以太网使用*SSH*"或"使用*USB*转串口转换器"章节。
3. 在Debian上扩展分区，请参见《昉·星光 2单板计算机快速参考手册》中"扩展分区"章节。
4. 在Debian上执行以下命令以安装并创建Python3虚拟环境:

```
sudo apt install python3-venv
python3 -m venv myvenv
```

注:
可自定义 "myvenv" 名称。

5. 在昉·星光 2 Debian上执行`pip`命令，以安装`VisionFive.gpio`包:

注:
由于pypi.org官网尚不支持上传RISC-V平台的`whl`安装包，不能直接执行`python3 -m pip install VisionFive.gpio`命令在线安装，因此请按照以下步骤安装`VisionFive.gpio`包。

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

a. 执行以下命令，在新创建的虚拟环境中安装依赖包：

```
sudo apt install libxml2-dev libxslt-dev
source ./myvenv/bin/activate
python3 -m pip install requests wget bs4
```

b. 执行以下命令，运行安装脚本`Install_VisionFive_gpio.py`：

```
python3 Install_VisionFive_gpio.py
```

安装脚本代码如下：

```
import requests
import wget
import sys
import os
from bs4 import BeautifulSoup

def parse_data(link_addr, class_type, key_str):
    req = requests.get(url=link_addr)
    req.encoding = "utf-8"
    html = req.text
    soup = BeautifulSoup(req.text, features="html.parser")
    package_version = soup.find(class_type, class_=key_str)
    dd = package_version.text.strip()
    data = dd.split()
    return data

def parse_link(link_addr, class_type, key_str):
    version_list = []
    req = requests.get(url=link_addr)
    req.encoding = "utf-8"
    html = req.text
    soup = BeautifulSoup(req.text, features="html.parser")
    search_data = soup.find_all(class_type, class_=key_str)
    for i in range(0, len(search_data)):
        search_data[i] = search_data[i].find("a").get("href)
        version_list.append(search_data[i].split("cp")[-1].split("-")[0])

    python_version = sys.version
    python_version = python_version.split(".")[0] + python_version.split(".")[1]

    for i in range(0, len(search_data)):
        if python_version == version_list[i]:
            return search_data[i]

    return search_data[0]

def get_dl_addr_page():
    link_address = "https://pypi.org/project/VisionFive.gpio/#history"
    key_str = "release version"
    class_key = "p"
    data_get = parse_data(link_address, class_key, key_str)
    latest_version = data_get[0]
    dl_addr_page
 = "https://pypi.org/project/VisionFive.gpio/{}/#files".format(latest_version)
    return dl_addr_page

def get_dl_addr_of_latest_version(link_addr):
    key_str = "card file card"
    class_key = "div"
    addr_get = parse_link(link_addr, class_key, key_str)

    return addr_get

def main():
    dl_addr_p = get_dl_addr_page()
    whl_dl_addr = get_dl_addr_of_latest_version(dl_addr_p)

    whl_name = whl_dl_addr.split("/")[-1]
    whl_name_suffix = os.path.splitext(whl_name)[-1]
```


© 2018-2025 上海赛昉半导体科技有限公司
版权所有

```
    whl_name_prefix = os.path.splitext(whl_name)[0]
    whl_name_prefix_no_platform = whl_name_prefix[0: len(whl_name_prefix) - 3]
    new_platform = "linux_riscv64"

    rename_whl_name = "{}{}{}".format(whl_name_prefix_no_platform, new_platform,
 whl_name_suffix)

    wget.download(whl_dl_addr, out=rename_whl_name)

    os.system("pip install " + rename_whl_name)
    os.system("rm -rf " + rename_whl_name)

if __name__ == '__main__':
    sys.exit(main())
```

c. （可选）退出 Python3 虚拟环境。

```
deactivate
```


© 2018-2025 上海赛昉半导体科技有限公司
版权所有

3. 执行演示代码

执行以下操作，以在昉·星光 2的Debian系统上运行演示代码：

1. 找到测试代码`buzzer.py`所在的目录：

   a. 进入 Python3 虚拟环境：

   ```
   source ./myvenv/bin/activate
   ```

   b. 执行以下命令安装依赖：

   ```
   python3 -m pip install pillow
   ```

   c. 执行以下命令以获取`VisionFive.gpio`所在的目录：

   ```
   python3 -m pip show VisionFive.gpio
   ```

   示例结果：

   ```
   Location: /home/user/myvenv/lib/python3.11/site-packages
   ```

   注：

   实际输出取决于应用的安装方式。

   d. 执行以下命令进入目录，例如上一步所示的 `/home/user/myvenv/lib/python3.11/site-packages`：

   ```
   cd /home/user/myvenv/lib/python3.11/site-packages
   ```

   e. 执行以下命令进入`sample-code`目录：

   ```
   cd ./VisionFive/sample-code/
   ```

2. 在`sample-code`目录下，执行以下命令：

   ```
   sudo python buzzer.py
   ```

   或者，您也可以执行以下命令：

   ```
   sudo python3 buzzer.py
   ```

3. 根据提示输入数值，配置蜂鸣声的音高和持续时间：

   - **Enter Pitch (200 to 20000)**：蜂鸣器的频率（范围：200-20000）。单位：Hz。例如，`400`。

     注：
     如果数值超出了频率范围（200-20000 Hz），系统将返回警告信息，您需要重新输入音高值。

   - **Enter Cycle (seconds)**：蜂鸣器持续鸣叫的时间（秒）。例如，`100`。

   示例：

   ```
   [riscv@fedora-starfive sample-code]$ sudo python3 buzzer.py
   Enter Pitch (200 to 20000): 400
   Enter Cycle (seconds): 100
   ```

   结果：

   蜂鸣器以400 Hz的音高持续鸣叫了100秒。

4. （可选）退出 Python3 虚拟环境。

   ```
   deactivate
   ```


© 2018-2025 上海赛昉半导体科技有限公司
版权所有

4. 演示源代码

本演示中的资源代码仅作为参考。

buzzer.py:

```
'''
Please make sure the buzzer is connected to the correct pins.
The following table describes how to connect the buzzer to the 40-pin header.
----------------------------------------
Passive Buzzer___Pin Number_____Pin Name
    VCC             1         3.3V Power
    GND             6           GND
    I/O             18          GPIO51
----------------------------------------
'''

import VisionFive.gpio as GPIO
import time

buzz_pin = 18
ErrOutOfRange = 0

def setup():
    #Configure the direction of buzz_pin as out.
    GPIO.setup(buzz_pin, GPIO.OUT)
    #Configure the voltage level of buzz_pin as high.
    GPIO.output(buzz_pin, GPIO.HIGH)

def pitch_in_check():
    val_in = input('Enter Pitch (200 to 20000): ')
    val = float(val_in)

    if 200 <= val <= 20000:
        return val
    else:
        print('The input data is out of range (200 to 20,000 Hz). Please re-enter.')
        return ErrOutOfRange

def loop(pitch, cycle):
    delay = 1.0 / pitch
    cycle = int((cycle * pitch)/2)

    #Buzzer beeps.
    while cycle >= 0:
        GPIO.output(buzz_pin, GPIO.LOW)
        time.sleep(delay)
        GPIO.output(buzz_pin, GPIO.HIGH)
        time.sleep(delay)

        cycle = cycle - 1

def destroy():
    GPIO.output(buzz_pin, GPIO.HIGH)
    GPIO.cleanup()

if __name__ == '__main__':
    setup()
    try:
        #Input value of pitch (200 to 20,000 Hz).
        pitch = pitch_in_check()
        while pitch == 0:
            pitch = pitch_in_check()

        #Input value of cycle time (seconds).
        cycle_in = input("Enter Cycle (seconds): ")
        cycle = int(cycle_in)

        #The buzzer beeps with the specified pitch and cycle.
        loop(pitch, cycle)
```


© 2018-2025 上海赛昉半导体科技有限公司
版权所有

```
    finally:
        destroy()
```


© 2018-2025 上海赛昉半导体科技有限公司
版权所有

5. 资源下载

点击本栏找到所有的代码下载资源。

本页包括所有赛昉科技提供的代码下载资源。

- RVspace Wiki
- 应用中心
- 文档中心
- 技术论坛
- 昉·星光 2 GitHub代码仓
- 昉·星光 2 Debian操作系统下载
- 代码下载（赛昉科技官方GitHub页面）
- 所有开源技术文档

© 2018-2025 上海赛昉半导体科技有限公司
版权所有

6. 立即购买

点击本栏获取在线购买链接和配件购买链接。

购买单板计算机

点击以下页面，您可以找到所在地区的经销商，或覆盖全球的销售渠道，以购买昉·星光 2单板计算机。

- 购买昉·星光 2开发板

购买配件

点击以下页面，您可以找到所有昉·星光 2单板计算机已验证适配的配件及其购买链接。

- 购买配件

© 2018-2025 上海赛昉半导体科技有限公司
版权所有