

CCUSR8116 RAID 控制卡 系统盘安装使用手册

版本：1.0

关于安装手册

《CCUSR8116 RAID 控制卡系统盘安装手册》介绍了 CCUSR8116 控制卡 BIOS 环境下创建阵列，并在阵列上安装和使用系统。

本手册默认您已具备关于制作启动盘，常规安装系统的基础知识。

本手册包含以下几个部分内容

“CCUSR8116 RAID 控制卡介绍”概述了 CCUSR8116 RAID 控制卡的特性，以及 RAID 基本概念和控制卡支持的 RAID 级别等相关内容。

“硬件安装”介绍了 CCUSR8116 控制卡硬件设计以及安装步骤。

“CCUSR8116 控制卡 BIOS 功能介绍”介绍了 CCUSR8116 控制卡 BIOS 的基本操作。

“可引导阵列系统安装前准备”介绍了如何制作启动盘、编译驱动、不同系统下的驱动镜像的打包与拷贝。

“可引导阵列系统安装”介绍了在“可引导阵列系统安装前准备”章节基础上，如何将不同的系统安装到可引导阵列。

“可引导阵列系统使用”简单介绍了如何加载使用系统。

“可引导阵列系统驱动更新”介绍了在 RAID 卡驱动更新后如何同步到 initramfs。

目 录

1. CCUSR8116 RAID 控制卡介绍	1
1.1. 产品特性	1
1.2. RAID 基础	1
2. 硬件安装	5
2.1. CCUSR8116 控制卡接口说明	5
2.2. 安装 CCUSR8116 控制卡	5
2.3. 安装超级电容模块（可选）	6
3. CCUSR8116 RAID 控制卡 BIOS 功能介绍	7
3.1. 主板 BIOS 设置	7
3.2. 控制卡信息	8
3.3. 磁盘管理	8
3.4. 初始化磁盘	10
3.5. 创建磁盘阵列	11
3.6. 热备盘池	12
3.7. 重新扫描磁盘	12
4. 可引导阵列系统安装前准备	13
4.1. 下载系统镜像	13
4.2. 制作 U 盘启动盘	14
4.3. 安装系统镜像到普通硬盘	14
4.4. 拷贝驱动源码及打包脚本到系统盘	15
4.4.1. UOS Desktop	16
4.4.2. UOS Server	16
4.4.3. Kylin Desktop	16
4.4.4. Kylin Server	17
4.5. 编译驱动	17
4.6. 打包驱动镜像	17
4.6.1. UOS Desktop	18
4.6.2. UOS Server	19
4.6.3. Kylin Desktop	21
4.6.4. Kylin Server	22
4.7. 拷贝驱动镜像	23

5. 安装系统镜像到可引导阵列	25
5.1. 安装 UOS DESKTOP	25
5.2. 安装 UOS SERVER	26
5.3. 安装 KYLIN DESKTOP	28
5.4. 安装 KYLIN SERVER	31
5.5. 安装 WINDOWS	34
6. 可引导阵列系统使用	38
7. 可引导阵列系统更新	39
7.1. 驱动版本更新	39
7.2. 内核版本更新	41

1. CCUSR8116 RAID 控制卡介绍

CCUSR8116 是一款16通道PCI-Express 3.0 x8转SATA 3.0的RAID控制卡。

CCUSR8116 可应用于企业存储、NAS存储、工作组和Web服务器、工作站的备份和安全系统。

1.1. 产品特性

CCUSR8116 RAID 控制卡的特性如下：

- 16通道PCI-E 3.0 x8转SATA 3.0 RAID控制卡
- 最多可支持16块SATA硬盘
- 支持64位LBA
- 支持原生命令队列
- 支持4Kn硬盘
- 支持RAID 0, 1, 10, 5, 6, 50和JBOD
- 支持在线容量扩展
- 支持在线RAID级别转换
- 每块硬盘可创建多个RAID阵列
- 支持S. M. A. R. T功能，监控硬盘状态
- 支持热插拔
- 支持磁盘扇区修复
- 支持超级电容掉电保护模块
- 基于Web浏览器的RAID管理软件

1.2. RAID 基础

以下 RAID 概念在使用 CCUSR8116 RAID 控制卡时会被提到。

磁盘阵列

“磁盘阵列”这个术语指控制卡上的一组硬盘经过组合后，被操作系统当作一个单独的存储设备单元来使用，也称为逻辑盘。

CCUSR8116 RAID 控制卡使用基于“分区”的磁盘阵列。一个物理硬盘可以在多至4个磁盘阵列中使用。分区将在创建阵列时自动在物理盘的可用空间中划分，无需手动创建。

CCUSR8116 RAID 控制卡支持下列几种磁盘阵列级别。

RAID 0

RAID0 由两个或两个以上硬盘组成，数据从不同的硬盘分段读写。这个数据分段方案不能通过容错来保护数据，但是由于数据平均地分布在各个硬盘上，RAID0 在所有 RAID 级别中的数据读写性能最佳。

RAID 1

RAID 1 磁盘阵列是把一块硬盘上的所有数据始终在另一块硬盘上做完整的备份存储。这种磁盘镜像方案提供数据冗余，确保了当一块硬盘损坏时，数据不会丢失。RAID 1 写入数据的速度和单个磁盘相同。不过由于读数据时是从每块硬盘上平均地读取，所以 RAID 1 读取数据的速度高于单个磁盘。

RAID 1/0

这种双重级别的 RAID 类型使用两个或多个 RAID1 阵列来创建一个 RAID0 阵列。由于磁盘阵列分段，使数据读取和写入的性能都有所提高。要创建一个 RAID1/0 阵列至少需要 4 块硬盘。

RAID 5

RAID5 的数据分布类似于 RAID0，但是它的每条数据段都带有校验码（异或运算），当一个硬盘损坏时就可以通过其他硬盘恢复丢失的数据。这种磁盘阵列方案具有性能优，容错性强，容量大，存储效率高等优点。要创建一个 RAID5 阵列至少需要 3 块硬盘。

RAID6

RAID6 的数据分布类似于 RAID5，但是它的每条数据段都带有两份校验码，当一个或者两个硬盘损坏时可以通过其他硬盘恢复丢失的数据。这种磁盘阵列方案具有性能优，容错性强，容量大，存储效率高等优点。要创建一个 RAID6 阵列至少需要 4 块硬盘。

RAID 5/0

这种双重级别的 RAID 类型使用两个或多个 RAID5 阵列来创建一个 RAID0 阵列。由于磁盘阵列分段，使数据读取和写入的性能都有所提高。要创建一个 RAID5/0 阵列至少需要 6 块硬盘。

RAID 6/0

这种双重级别的 RAID 类型使用两个或多个 RAID6 阵列来创建一个 RAID0 阵列。由于磁盘阵列分段，使数据读取和写入的性能都有所提高。要创建一个 RAID6/0 阵列至少需要 8 块硬盘。

JBOD

JBOD 代表“简单磁盘捆绑”，就是把不同的硬盘空间串连在一起。JBOD 只提供磁盘空间的整合，不会提高系统性能也不支持数据冗余。

磁盘阵列初始化

一个冗余的磁盘阵列 (RAID5, RAID1, RAID10) 需要被初始化以确保完整的性能和可靠性。而没有冗余的磁盘阵列 (RAID0, JBOD) 不需要被初始化。

当您在 RAID 控制卡的 BIOS 配置程序里或使用 RAID 管理软件创建磁盘阵列时，您可以指定一个初始化选项 (跳过初始化，前台初始化，后台初始化)。

前台初始化

前台初始化将磁盘阵列上所有的数据清零，直到初始化完成后这个磁盘阵列才

能被系统使用。

后台初始化

后台初始化允许磁盘阵列立即就能被使用。对于RAID1和RAID10磁盘阵列，初始化的结果是复制数据，等同于两块盘的镜像。对于RAID5，RAID50和RAID6初始化的结果是在磁盘阵列所有的成员盘上创建校验码。

注意:

一个没有经过初始化的RAID1、RAID5、RAID6、RAID10、RAID50仍然可能提供冗余性能并应对子RAID中的一个（RAID6支持两个）成员磁盘损坏。然而在没有经过初始化的这些阵列上进行校验操作时，校验可能会失败。

另外，在正式使用之前，这些阵列都应该进行初始化操作。不建议使用跳过初始化操作，除非您非常明确如此操作的含义和后果。

在线容量扩展 (Online Capacity Expansion)

这个特性允许把任意数量的硬盘添加到一个阵列中，并在扩展过程中继续访问数据。

在线 RAID 级别转换 (Online RAID Level Migration)

这个特性允许把一种级别的磁盘阵列转换成另一种级别。在转换过程中数据可以继续被用户访问。

硬盘初始化 (Disk Initialization)

硬盘必须先经过初始化，才能在磁盘阵列中使用。硬盘的初始化就是把必要的配置信息写入该硬盘，并删除该硬盘上所有已有的数据。这是一个即刻响应的操作。

未初始化的硬盘将显示为Legacy状态，以直通方式提供给系统访问。

如果一个硬盘在初始化后没有将它创建为磁盘阵列或设置为热备盘，在系统重新启动后它将恢复为Legacy模式。

热备盘 (Spare Disk)

热备盘可在一个冗余阵列成员失效时，自动替换失效硬盘并开始数据重建。

缓存策略 (Cache Policy)

CCUSR8116控制卡上内置2GB DDR内存用于缓存数据以提高读写性能。

CCUSR8116控制卡支持如下缓存策略：

无 (None)

不使用缓存。系统的读写请求将直接发送到硬盘。

透写 (Write-through)

系统的写请求将写入缓存，同时写入硬盘。系统的读请求会先查找缓存，如果缓存命中则直接返回缓存数据，不命中时从硬盘读取数据到缓存并

返回系统。使用透写模式时，仅数据读取操作可利用缓存提供的性能优势。

回写（Write-back）

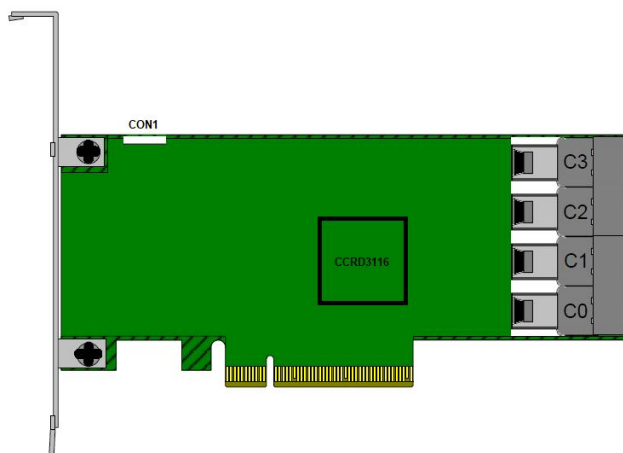
系统的写请求将写入缓存后直接返回成功响应。当控制卡缓存不足时，控制卡将在后台启动缓存刷新操作将数据写入硬盘。系统的读请求会先查找缓存，如果缓存命中则直接返回缓存数据，不命中时从硬盘读取。

回写模式提供了最佳的读写性能，但是存在数据意外丢失的风险，可配合超级电容模块的掉电保护功能使用。

2. 硬件安装

2.1. CCUSR8116 控制卡接口说明

CCUSR8116 RAID 控制卡的接口布局如下图所示：



C0- C3

在 CCUSR8116 RAID 控制卡上共有 4 个 SFF-8643 端口（C0-C3），每个端口可以支持 4 个 SATA 通道，C0 对应 PORT1-PORT4，C1 对应 PORT5-PORT8，C2 对应 PORT9-PORT12，C3 对应 PORT13-PORT16。

CON1

超级电容接口

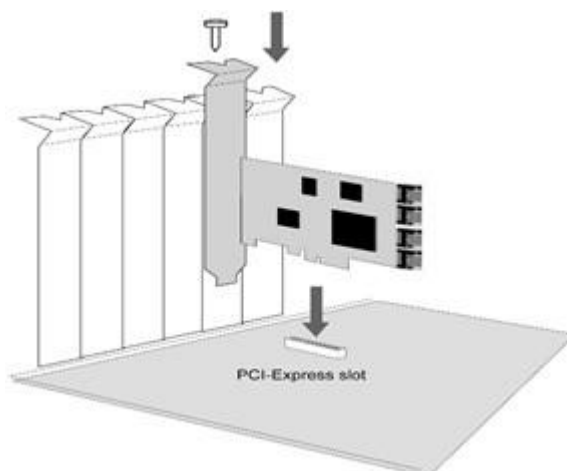
2.2. 安装 CCUSR8116 控制卡

警告

-
- 由于电脑设备中可能带高压电，因此在安装本包装中任何硬件或移动任何电脑设备的保护罩前，请关闭电源开关并拔下电源插头。等放好保护罩后再重新插上电源。
 - 创建或删除磁盘阵列会破坏成员盘上现有的数据。如果硬盘中含有重要数据，请在把硬盘连接到控制卡之前做好备份，把这些数据保存到其他地方。
-

CCUSR8116 控制卡提供标准挡板和短挡板两种支架。根据机箱的设计选用您可能需要使用短挡板支架取代标准支架。

1. 打开机箱盖，找一个空闲的 PCI-Express 插槽。
2. 移去插槽挡盖。
3. 把 CCUSR8116 控制卡插到 PCI-E 槽里，并用螺丝固定在机箱内。



4. 安装好适配卡后，用数据线将硬盘和适配卡连接起来。
5. 如果您的机箱带硬盘热插拔架，可以把数据线连接到机箱背板上，而不必直接连到每块单独的硬盘上。请依据机箱设计选择正确的安装方式。

2.3. 安装超级电容模块（可选）

CCUSR8116 控制卡支持独立的超级电容模块以提供掉电数据保护功能。在意外掉电时，超级电容模块将继续为控制卡供电，直到控制卡将其缓存中的数据写入控制卡上的 EMMC 存储空间。

您可将超级电容模块安装在和控制卡所在插槽相邻的一个空闲 PCIe 插槽，并使用线缆将超级电容模块和控制卡连接。

3. CCUSR8116 RAID 控制卡 BIOS 功能介绍

CCUSR8116 控制卡支持 UEFI BIOS 和 UEFI HII 界面规范，可以在受支持的主板上通过 BIOS 进行 RAID 设置。

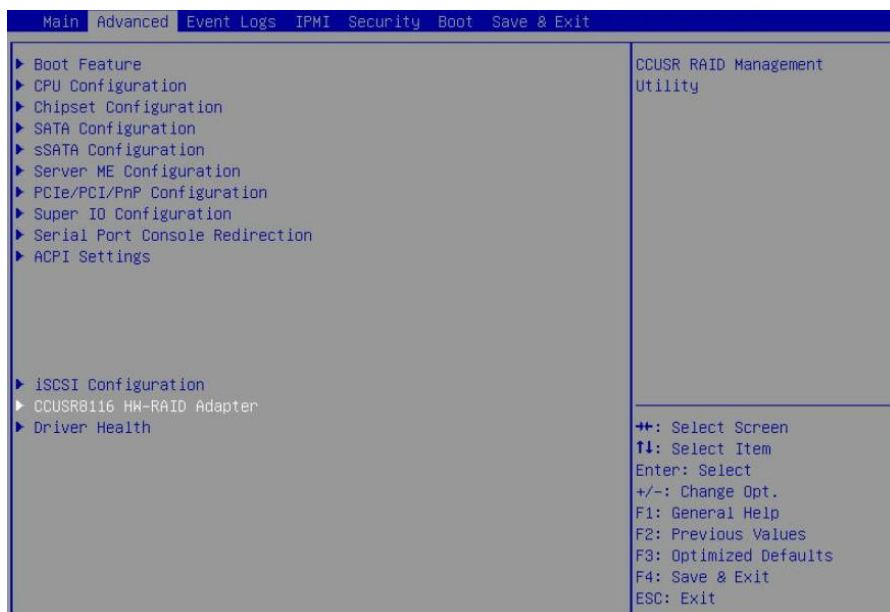
注意

不同主板系统的 BIOS 界面显示内容和操作方式有所不同，本手册的说明仅为示例。请参考主板的使用手册进行相应的设置。

3.1. 主板 BIOS 设置

要使用 UEFI BIOS，需要将主板设置为能够加载 UEFI 模式的 PCI Option ROM。以下以 Supermicro X10DRL-i 主板为例进行说明。

开机后根据提示进入主板 BIOS 界面，在主板 BIOS - Advanced 页面可以看到 CCUSR8116 磁盘阵列控制卡：如图：



选中 CCUSR8116 磁盘阵列控制卡菜单，进入后可以看到控制卡的主菜单如下所示：



主菜单包含如下功能：

- ◆ 控制卡信息
- ◆ 磁盘管理
- ◆ 初始化磁盘
- ◆ 创建阵列
- ◆ 热备盘池管理
- ◆ 重新扫描磁盘

3.2. 控制卡信息

进入“控制卡信息”页面可以看到控制卡的详细信息：

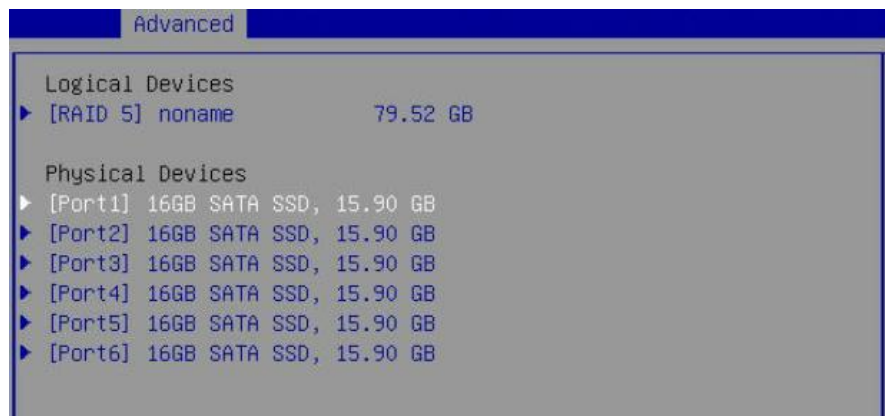
包括控制卡 所在位置、PCI 连接速度、序列号、固件版本、BIOS 版本和超级电容模块信息。

Advanced	
Location	PCI bus 2
PCIe Link	3.0 x8
Serial Number	CCUSR-429496729
Firmware Rev.	v1.0-b1877-g63aeaf69
BIOS Version	v1.0-b1893-gc1e502ed
Supercapacitor Module	
Status	Enabled, Charging
Voltage	11.00V
Temperature	32.6
► Start Discharge Test	

当连接了超级电容模块时，将同时显示了超级电容模块的状态、电压和温度信息。并可通过“开始放电测试”菜单对超级电容模块进行放电测试。有关超级电容模块的相关说明请参见《掉电保护》一章的说明。

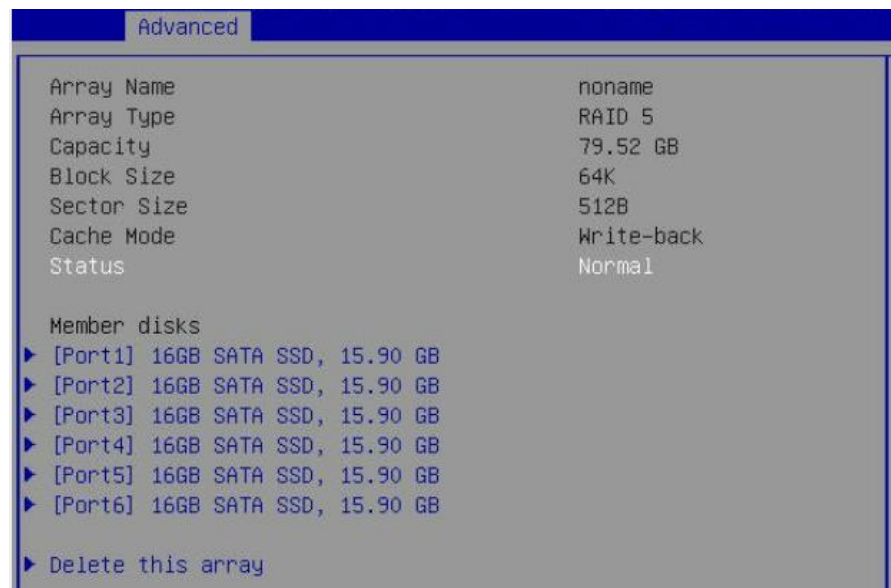
3.3. 磁盘管理

进入“磁盘管理”页面可看到所有的逻辑盘和物理盘。

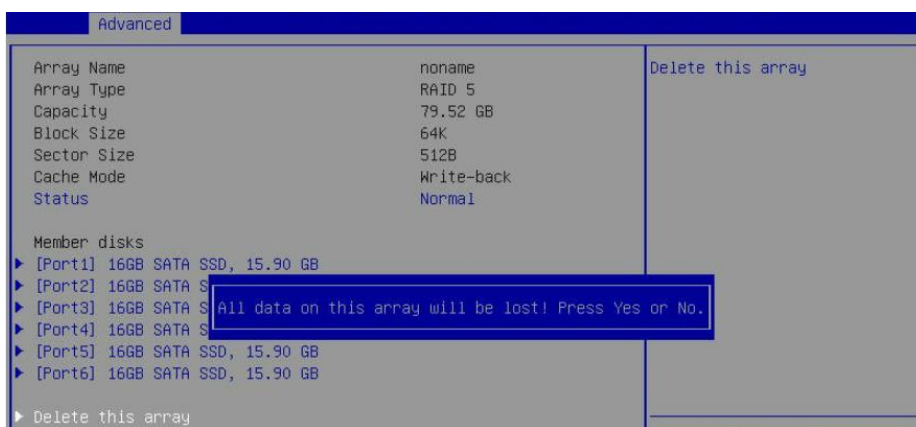


选中某个阵列或磁盘，按回车键，可以查看详细信息。

阵列的详细信息页面如下所示：



在以上界面中选择“删除此阵列”可以删除当前阵列：

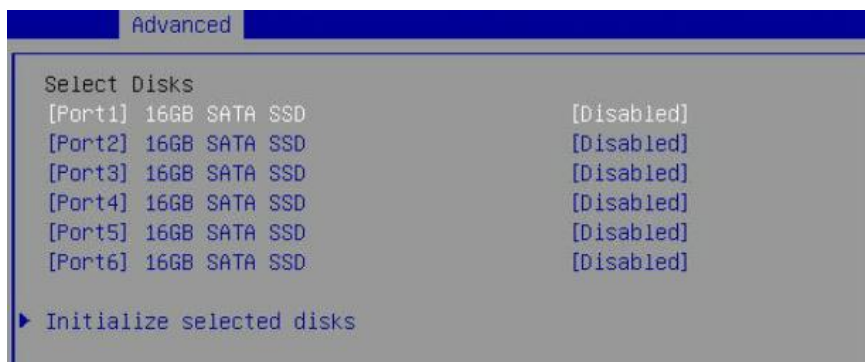


根据提示信息选中“Y”按 Enter 键即可删除阵列。

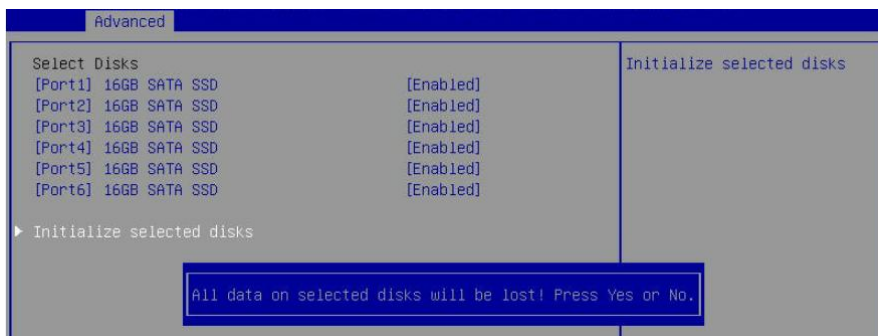
3. 4. 初始化磁盘

选择“初始化磁盘”菜单进入磁盘初始化页面。

此页面将列出所有未初始化的磁盘。



选择需要初始化的硬盘（如上图将 **[Disabled]** 按 Enter 键改为 **[Enabled]** 即表示选中），然后选择“初始化选定的物理盘”进行初始化操作。

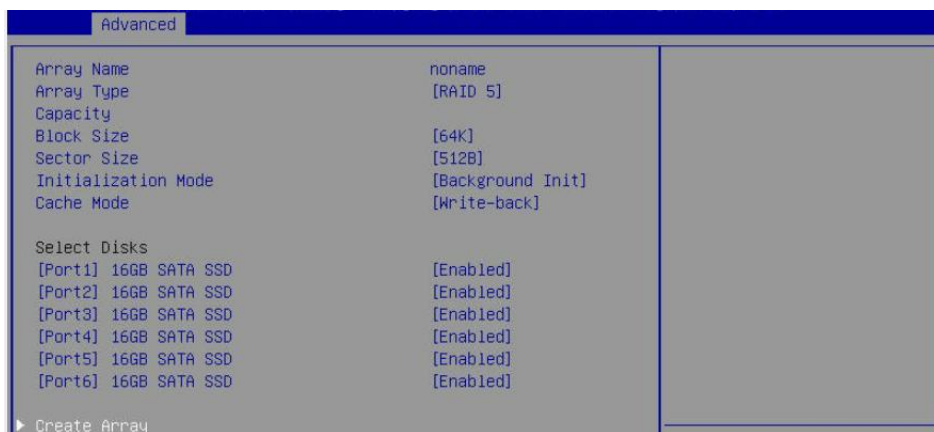


根据提示信息选中“Y”按 Enter 键初始化所选择的硬盘。

3.5. 创建磁盘阵列

选择“创建磁盘阵列”菜单进入创建阵列页面，此页面可以根据需要进行阵列创建选项的设置：

1. 指定阵列名称。
2. 选择 RAID 级别。
3. 指定阵列容量。如不指定，将使用最大允许的容量创建阵列。
4. 阵列类型为 RAID0，RAID10，RAID5，RAID50，RAID6 时，指定条带大小。
5. 指定逻辑扇区大小。默认为 512 字节。
6. 阵列类型为冗余类型时，指定一个初始化选项。
7. 选择阵列缓存模式。RAID0，1，10，JBOD 阵列支持无缓存、透写和回写三种模式；RAID5，6，50 阵列支持透写和回写模式，不支持无缓存模式。
8. 选择该阵列的成员盘。



The screenshot shows a terminal-style interface for configuring a RAID array. The title bar is labeled 'Advanced'. The configuration parameters are as follows:

Parameter	Value
Array Name	noame
Array Type	[RAID 5]
Capacity	
Block Size	[64K]
Sector Size	[512B]
Initialization Mode	[Background Init]
Cache Mode	[Write-back]

Below these settings is a section titled 'Select Disks' with a list of six 16GB SATA SSDs, each with an '[Enabled]' status:

Port	Capacity	Type	Status
[Port1]	16GB	SATA SSD	[Enabled]
[Port2]	16GB	SATA SSD	[Enabled]
[Port3]	16GB	SATA SSD	[Enabled]
[Port4]	16GB	SATA SSD	[Enabled]
[Port5]	16GB	SATA SSD	[Enabled]
[Port6]	16GB	SATA SSD	[Enabled]

At the bottom left, there is a button labeled 'Create Array'.

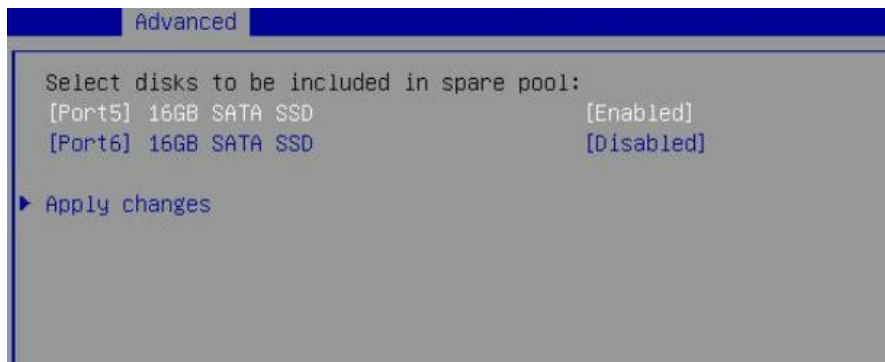
完成阵列参数信息设置后选择“创建阵列”，即可完成阵列创建。

如果指定了初始化选项，控制卡将在后台启动阵列初始化任务。

3.6. 热备盘池

选择“热备盘池”菜单进入备用池管理页面，此页面列出了所有可以作为备用盘使用的硬盘。

硬盘状态显示为 **[Enabled]** 表示设置此硬盘为备用盘。



修改硬盘状态后（如上图把 **[Disabled]** 改为 **[Enabled]** 选择“保存设置”即可。



如果您需要删除热备盘把硬盘状态**[Enabled]** 改为 **[Disabled]**然后再次选中“保存设置”即可。

3.7. 重新扫描磁盘

当控制卡所连接磁盘有热插拔盘动作或磁盘损坏时；选中此菜单可以刷新磁盘状态。

“重新扫描磁盘”后页面自动跳转到“磁盘管理”菜单。



4. 可引导阵列系统安装前准备

- 1.本章节的目的是为了讲述如何下载系统镜像、如何编译驱动、如何将驱动打包到系统镜像启动盘中。
- 2.我们后续将展示 UOS 桌面版、UOS 服务器版、Kylin 桌面版、Kylin 服务器版这四种常用国产操作系统的安装步骤，其它操作系统安装步骤类似。
- 3.因不同国产操作系统之间差异性较大，同一操作系统不同子版本可能安装环境都有所变化，以下的方法目前充当临时的方法（后续我们将有计划接入国产平台的生态，便可以更快捷更方便的安装系统到可引导阵列上），如您按以下步骤进行后仍无法安装您想要的系统，请与我们联系。

4.1. 下载系统镜像

1.UOS 系统镜像下载链接：<https://www.uniontech.com/>

请根据自身实际硬件平台选择。



2.Kylin 系统镜像下载链接：<https://www.kylinos.cn/>

目前 Kylin 官网提供的是试用版下载链接，请根据自身实际硬件平台选择（如您有更好的获取镜像途径也可以自行获取）。



4.2. 制作 U 盘启动盘

将 U 盘格式化成 **fat32** 格式，然后基于 ISO 镜像制作启动盘。

在“**5-相关工具**”>**5-1-启动盘制作工具**”目录下提供了几种常用的制作启动盘的工具，请根据自身喜好选择不同制盘工具。

注：不同的系统 ISO 镜像是有差异性的，如果做出来的启动盘会在安装的时候无法识别到 ISO 介质，请尝试换另一种制盘工具制作。



根据过往经验，我们建议您：

UOS Desktop 启动盘采用 DEEPIN_B 工具制作。

UOS Server 启动盘采用 Fedora Media Writer 工具制作。

Kylin Desktop 启动盘采用 UltraISO 工具制作。

Kylin Server 启动盘采用 Rufus 工具制作。

Windows 启动盘采用 Ventoy 工具制作。

4.3. 安装系统镜像到普通硬盘

注：安装 windows 系统可以跳过此小节（以及跳过后面的 4.4、4.5、4.6 小节），windows 系统制作启动盘后，参考 4.7 小节直接拷贝 RAID 卡驱动到启动盘后即可将 windows 系统安装到阵列上。

1. 安装系统镜像到主板的直连普通硬盘上的过程不在此进行展示，请您自行安装。
2. 安装系统镜像到普通硬盘的目的是：**为了在同一个系统上编译驱动源码生成 ko 文件、然后将生成的 ko 文件驱动打包成启动盘镜像能够识别的格式**
3. 当安装 Kylin Server 系统时，可能会提示软件源出错，导致无法正常安装，如下：



此时，我们要修改启动参数，增加“**inst.repo**”参数，该参数内容应该与“**inst.stage2**”参数完全相同，例如：“**inst.repo=hd:LABEL=KYLIN-SERVE**”。

```
setparams 'Install Kylin Linux Advanced Server V10'

linux /images/pxeboot/vmlinuz inst.stage2=hd:LABEL=KYLIN-SERVE inst.re\
po=hd:LABEL=KYLIN-SERVE_ro console=tty0 video=efifb:off video=VGA-1:640x480-32\
@60me
initrd /images/pxeboot/initrd.img

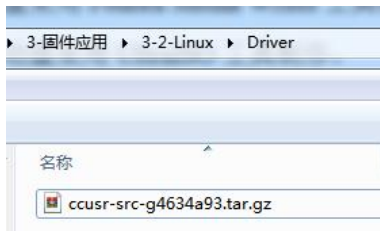
Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists possible
completions.
```

4. 4. 拷贝驱动源码及打包脚本到系统盘

所有 Linux 内核的系统发行版使用的 RAID 控制卡驱动源码是一样的，但是不同系统所使用的打包/加载脚本是不一样的。

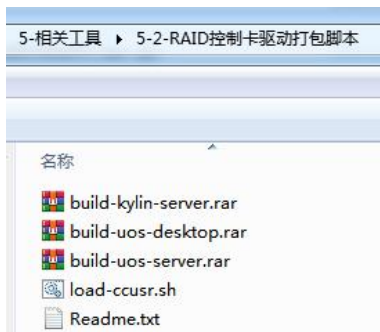
4.4.1. UOS Desktop

1.将 RAID 卡驱动源码（“3-软件环境\3-2-Linux\Driver\ccusr-src-g4634a93.tar.gz”）拷贝到系统盘。



注：驱动源码会不定时动态更新，可能命名不一定为“ccusr-src-g4634a93.tar.gz”，请以最新的为准。

2.将 RAID 卡驱动打包脚本（“5-相关工具\5-2-RAID 控制卡驱动打包脚本\build-uos-desktop.rar”）拷贝到系统盘。



4.4.2. UOS Server

1.将 RAID 卡驱动源码（“3-软件环境\3-2-Linux\Driver\ccusr-src-g4634a93.tar.gz”）拷贝到系统盘。

2.将 RAID 卡驱动打包脚本（“5-相关工具\5-2-RAID 控制卡驱动打包脚本\build-uos-server.rar”）拷贝到系统盘。

4.4.3. Kylin Desktop

1.将 RAID 卡驱动源码（“3-软件环境\3-2-Linux\Driver\ccusr-src-g4634a93.tar.gz”）拷贝到系统盘。

2.将 RAID 卡驱动加载脚本（“5-相关工具\5-2-RAID 控制卡驱动打包脚本\load-ccusr.sh”）拷贝到系统盘。

4.4.4. Kylin Server

- 1.将 RAID 卡驱动源码（“3-软件环境\3-2-Linux\Driver\ccusr-src-g4634a93.tar.gz”）拷贝到系统盘。
- 2.将 RAID 卡驱动打包脚本（“5-相关工具\5-2-RAID 控制卡驱动打包脚本\build-kylin-server.rar”）拷贝到系统盘。

4.5. 编译驱动

说明：不同系统编译驱动的步骤都是一样的（解压后 make），详细步骤如下：

- 1.在主板的直连普通硬盘上安装好系统后，重新断电/上电。
- 2.开机时选择启动设备为系统所在的设备（普通硬盘）。
- 3.进入系统桌面，然后打开 1 个控制台终端。
- 4.将 U 盘启动盘根目录的驱动拷贝到系统目录，然后解压。

PC> cp XXX/ccusr-src-g4634a93.tar.gz .

PC> tar xf ccusr-src-g4634a93.tar.gz

- 5.进入解压后的目录，然后输入 make 进行编译。

PC> cd ccusr-src-g4634a93/

PC> make

```
[ccore@localhost ~]$ cp /media/ccore/U05/ccusr-src-g4634a93.tar.gz .
[ccore@localhost ~]$
[ccore@localhost ~]$ tar xf ccusr-src-g4634a93.tar.gz
[ccore@localhost ~]$ cd ccusr-src-g4634a93/
[ccore@localhost ccusr-src-g4634a93]$ ls
cciop.h  ccusr.c  ccusr.h  Kbuild  Makefile
[ccore@localhost ccusr-src-g4634a93]$ make
make[1]: 进入目录 "/usr/src/kernels/4.19.0-91.82.152.uelc20.aarch64"
CC [M] /home/ccore/ccusr-src-g4634a93/ccusr.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/ccore/ccusr-src-g4634a93/ccusr.mod.o
LD [M] /home/ccore/ccusr-src-g4634a93/ccusr.ko
make[1]: 离开目录 "/usr/src/kernels/4.19.0-91.82.152.uelc20.aarch64"
[ccore@localhost ccusr-src-g4634a93]$
[ccore@localhost ccusr-src-g4634a93]$ ls
cciop.h  ccusr.c  ccusr.h  ccusr.ko  ccusr.mod.c  ccusr.mod.o  ccusr.o  Kbuild  Makefile  modules.order  Module.symvers
[ccore@localhost ccusr-src-g4634a93]$
```

生成的 ccusr.ko 文件就是我们要的驱动文件。

4.6. 打包驱动镜像

不同系统的打包驱动方式不一样，下面我们将对不同的系统打包驱动做出说明。

4.6.1. UOS Desktop

1.对驱动打包文件 build-uos-desktop.rar 进行解压，然后进到解压后的目录，并确保打包脚本有可执行权限。

```
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ ls
build-dd-uos.sh ccusr-uos output vrc9k-uos vrcdrv-uos
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ chmod +x build-dd-uos.sh
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ ls
build-dd-uos.sh ccusr-uos output vrc9k-uos vrcdrv-uos
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
```

2.将之前编译好的 ko 驱动文件放到 output 目录下的指定目录中（不同平台架构以及不同系统版本创建的目录是不一样的，规则如下所示）：

output:

→loongarch64:	-----龙芯指令集架构('uname -m')
→4.19.0-loongson-3-desktop:	-----系统版本('uname -r')
→ccusr.ko	-----对应系统版本的驱动
→x86_64:	-----x86 架构
→4.19.0-6-amd64 :	-----系统版本
→ccusr.ko	-----对应系统版本的驱动
→4.19.0-6-server-amd64:	-----系统版本
→ccusr.ko	-----对应系统版本的驱动
...	

```
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ arch
aarch64
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ uname -r
4.19.0-arm64-desktop
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ mkdir -p output/aarch64/4.19.0-arm64-desktop
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ cp ../../ccusr-src-g4634a93/ccusr.ko output/aarch64/4.19.0-arm64-
desktop/
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
```

PC> mkdir -p output/`uname -m`/`uname -r`

PC> cp ccusr.ko output/`uname -m`/`uname -r`

注: output 中所包含的驱动必须在对应系统的环境下编译，并且架构名、系统内核名必须准确。

3.运行 build-dd-uos.sh，会在同目录下生成 aaa-uos-bbb.squashfs(aaa 为驱动名称，bbb 为日期)。

PC> ./build-dd-uos.sh

```

ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ ./build-dd-uos.sh
output/aarch64/4.19.0-arm64-desktop/ccusr.ko
output/loongarch64/4.19.0-loongson-3-desktop/ccusr.ko
Parallel mksquashfs: Using 64 processors
Creating 4.0 filesystem on ccusr-uos-230907.squashfs, block size 131072.
[=====] 7/7 100%

Exportable Squashfs 4.0 filesystem, gzip compressed, data block size 131072
    compressed data, compressed metadata, compressed fragments, compressed xattrs
    duplicates are removed
Filesystem size 159.52 Kbytes (0.16 Mbytes)
    28.39% of uncompressed filesystem size (561.93 Kbytes)
Inode table size 256 bytes (0.25 Kbytes)
    40.89% of uncompressed inode table size (626 bytes)
Directory table size 285 bytes (0.28 Kbytes)
    56.21% of uncompressed directory table size (507 bytes)
Number of duplicate files found 0
Number of inodes 19
Number of files 4
Number of fragments 1
Number of symbolic links 0
Number of device nodes 0
Number of fifo nodes 0
Number of socket nodes 0
Number of directories 15
Number of ids (unique uids + gids) 1
Number of uids 1
    ccore (1000)
Number of gids 1
    ccore (1000)
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$ ls
build-dd-uos.sh  ccusr-uos  ccusr-uos-230907.squashfs  output  vrc9k-uos  vrcdrv-uos
ccore@ccore-PC:~/Desktop/build-uos-desktop/uos$

```

4.6.2. UOS Server

注：这里的打包驱动方法针对的是 UOS Server 1050 及之后的版本，之前的版本请参考 4.6.1 小节。

1.对驱动打包文件 build-uos-server.rar 进行解压，然后进到解压后的目录，并确保打包脚本有可执行权限。

```

[ccore@localhost ~]$ cd build-uos-server/
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ ls
build-dd-uos-server.sh  output  rhdd3-iso-ccusr  rpm.spec
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ chmod +x build-dd-uos-server.sh
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ ls
build-dd-uos-server.sh  output  rhdd3-iso-ccusr  rpm.spec
[ccore@localhost build-uos-server]$

```

2.将之前编译好的 ko 驱动文件放到 output 目录下的指定目录中（不同平台架构以及不同系统版本创建的目录是不一样的，规则跟 4.6.1 小节一样）：

```

[ccore@localhost build-uos-server]$ arch
aarch64
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ uname -r
4.19.0-91.82.152.uelc20.aarch64
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ mkdir -p output/aarch64/4.19.0-91.82.152.uelc20.aarch64
[ccore@localhost build-uos-server]$
[ccore@localhost build-uos-server]$ cp ../ccusr-src-g4634a93/ccusr.ko output/aarch64/4.19.0-91.82.152.uelc20.aarch64/
[ccore@localhost build-uos-server]$

```

注: output 中所包含的驱动必须在对应系统的环境下编译, 并且架构名、系统内核名必须准确。

3. 安装内核开发包以及依赖包 (需要能上网):

①在终端中输入”uname -r”, 得到内核版本 4.19.0-91.82.132.uelc20.loongarch64, 运行”yum install kernel-devel-4.19.0-91.82.132.uelc20.loongarch64”, 后续安装过程中都只需要按 Y, 直到安装完成。

PC> yum install kernel-dev-XXX

```
[root@localhost build-uos-server]# uname -r
4.19.0-91.82.152.uelc20.aarch64
[root@localhost build-uos-server]#
[root@localhost build-uos-server]# yum install kernel-dev-4.19.0-91.82.152.uelc20.aarch64
UnionTechOS 20 AppStream                                11 MB/s | 8.7 MB  00:00
UnionTechOS 20 BaseOS                                   6.3 MB/s | 2.5 MB 00:00
UnionTechOS 20 PowerTools                               6.1 MB/s | 2.5 MB 00:00
UnionTechOS 20 Plus                                     12 MB/s | 12 MB  00:00
UnionTechOS 20 Extras                                  10 kB/s | 2.0 kB  00:00
UnionTechOS 20 Update                                   4.0 MB/s | 1.6 MB 00:00
未找到匹配的参数: kernel-dev-4.19.0-91.82.152.uelc20.aarch64
错误: 没有任何匹配: kernel-dev-4.19.0-91.82.152.uelc20.aarch64
[root@localhost build-uos-server]#
```

②运行”yum install createrepo”安装 createrepo 包。

PC> yum install createrepo

```
[root@localhost build-uos-server]# yum install createrepo
上次元数据过期检查: 0:01:35 前, 执行于 2023年09月07日 星期四 18时32分59秒。
依赖关系解决。

=====
软件包                                架构          版本                仓库                大小
=====
安装:
createrepo_c                          aarch64       0.20.1-1.uelc20     UnionTechOS-20-AppStream 88 k
安装依赖关系:
createrepo_c-libs                     aarch64       0.20.1-1.uelc20     UnionTechOS-20-AppStream 115 k
drpm                                   aarch64       0.4.1-3.uelc20.3    UnionTechOS-20-AppStream 64 k
=====
事务概要
-----
安装 3 软件包

总下载: 268 k
安装大小: 742 k
确定吗? [y/N]: y
下载软件包:
(1/3): drpm-0.4.1-3.uelc20.3.aarch64.rpm                13 kB/s | 64 kB  00:05
(2/3): createrepo_c-0.20.1-1.uelc20.aarch64.rpm         18 kB/s | 88 kB  00:05
(3/3): createrepo_c-libs-0.20.1-1.uelc20.aarch64.rpm    22 kB/s | 115 kB 00:05
=====
```

③运行”yum install rpm-build”安装 rpmbuild 包 (不同硬件平台可能安装 rpmbuild 的命令不一样, 请根据实际情况安装)。

PC> yum install rpm-build

```
[root@localhost build-uos-server]# yum install rpm-build.aarch64
上次元数据过期检查: 0:07:26 前, 执行于 2023年09月07日 星期四 18时32分59秒。
依赖关系解决。
=====
软件包           架构      版本                仓库                大小
=====
安装:
rpm-build        aarch64    4.15.1-51.uelc20.01 UnionTechOS-20-AppStream 113 k
安装依赖关系:
ctags            aarch64    5.8-24.uelc20.2     UnionTechOS-20-AppStream 162 k
elfutils         aarch64    0.187-4.uelc20.02   UnionTechOS-20-BaseOS   532 k
gc               aarch64    7.6.4-5.uelc20.2    UnionTechOS-20-AppStream 98 k
gdb-headless     aarch64    9.2-7.0.4.uelc20.01 UnionTechOS-20-AppStream 3.5 M
guile            aarch64    5:2.0.14-9.uelc20.5 UnionTechOS-20-AppStream 3.5 M
libatomic_ops    aarch64    7.6.2-5.uelc20.2    UnionTechOS-20-AppStream 37 k
libbabeetrace    aarch64    1.5.8-10.0.1.uelc20.02 UnionTechOS-20-BaseOS   185 k
source-highlight aarch64    3.1.8-18.uelc20.4   UnionTechOS-20-AppStream 436 k
zstd             aarch64    1.5.1-2.0.2.uelc20.01 UnionTechOS-20-AppStream 392 k
=====
事务概要
-----
安装 10 软件包
```

④如果装完以上依赖之后运行 build-dd-uos-server.sh 还提示报错（一般不会再报错），请根据错误提示安装对应的工具包。

4.运行 build-dd-uos-server.sh，会在同目录下生成 **aaa-uos-server-bbb.iso** 的镜像(**aaa** 为驱动名称，**bbb** 为日期)。

PC> ./build-dd-uos-server.sh

```
[root@localhost build-uos-server]# ./build-dd-uos-server.sh
output/loongarch64/4.19.0-91.82.132.uelc20.loongarch64/ccusr.ko
rpmbuild --define "_rpmmdir rhdd3-iso-ccusr/rpms" --define "_arch loongarch64" --define "_rpmfilename loongarch64/ccusr-1.0-2
30907-4.19.0-91.82.132.uelc20.loongarch64.rpm" --define "module_dir /home/ccore/build-uos-server/output/loongarch64/4.19.0-9
1.82.132.uelc20.loongarch64" --define "module_name ccusr" --define "version 1.0" --define "release 230907" --define "kernel_
version 4.19.0-91.82.132.uelc20.loongarch64" --target loongarch64-uos-server-linux --nodeps -bb rpm.spec
构建目标平台: loongarch64-uos-server-linux
为目标loongarch64-uos-server-linux构建
正在执行(%prep): /bin/sh -e /var/tmp/rpm-tmp.yzgFaY
+ umask 022
+ cd /root/rpmbuild/BUILD
+ RPM_EC=0
++ jobs -p
+ exit 0
正在执行(%build): /bin/sh -e /var/tmp/rpm-tmp.QjYrTG
+ umask 022
+ cd /root/rpmbuild/BUILD
+ RPM_EC=0
++ jobs -p
+ exit 0
正在执行(%install): /bin/sh -e /var/tmp/rpm-tmp.9metL1
+ umask 022
+ cd /root/rpmbuild/BUILD
+ '[' /root/rpmbuild/BUILDROOT/ccusr-4.19.0-91.82.132.uelc20.loongarch64-1.0-230907.uelc20.loongarch64 '!=' / ']'
+ rm -rf /root/rpmbuild/BUILDROOT/ccusr-4.19.0-91.82.132.uelc20.loongarch64-1.0-230907.uelc20.loongarch64

[root@localhost build-uos-server]#
[root@localhost build-uos-server]# ls
build-dd-uos-server.sh  ccusr-uos-server-230907.iso  output  rhdd3-iso-ccusr  rpm.spec
[root@localhost build-uos-server]#
```

4.6.3. Kylin Desktop

Kylin Desktop 不需要对 ko 驱动文件进一步打包成镜像，直接配合 load-ccusr.sh 驱动加载脚本一起使用即可。

4.6.4. Kylin Server

1 对驱动打包文件 build-kylin-server.rar 进行解压，然后进到解压后的目录，并确保打包脚本有可执行权限。

```
[ccore@localhost ~]$ cd build-kylin-server/
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ ls
build-dd-kylin-server.sh  rhdd3-iso-ccusr  rhdd3-iso-vrcdrv
output                  rhdd3-iso-vrc9k  rpm.spec
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ chmod +x build-dd-kylin-server.sh
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ ls
build-dd-kylin-server.sh  rhdd3-iso-ccusr  rhdd3-iso-vrcdrv
output                  rhdd3-iso-vrc9k  rpm.spec
[ccore@localhost build-kylin-server]$
```

2.将之前编译好的 ko 驱动文件放到 output 目录下的指定目录中（不同平台架构以及不同系统版本创建的目录是不一样的，规则跟 4.6.1 小节一样）：

```
[ccore@localhost build-kylin-server]$ arch
aarch64
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ uname -r
4.19.90-52.22.v2207.ky10.aarch64
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ mkdir -p output/aarch64/4.19.90-52.22.v2207.ky10.aarch64
[ccore@localhost build-kylin-server]$
[ccore@localhost build-kylin-server]$ cp ../ccusr-src-g4634a93/ccusr.ko output/aarch64/4.19.90-52.22.v2207.ky10.aarch64/
[ccore@localhost build-kylin-server]$
```

注: output 中所包含的驱动必须在对应系统的环境下编译，并且架构名、系统内核名必须准确。

3.安装依赖包（需要能上网）：

①运行”yum install createrepo”安装 createrepo 包。

PC> yum install createrepo

```
[ccore@localhost build-kylin-server]$ sudo yum install createrepo
上次元数据过期检查: 0:12:30 前, 执行于 2023年09月08日 星期五 12时26分25秒。
依赖关系解决。
=====
Package               Architecture      Version           Repository        Size
=====
安装:
createrepo_c          aarch64          0.16.0-3.p01.ky10  ks10-adv-os      133 k
安装依赖关系:
drpm                  aarch64          0.5.0-1.ky10      ks10-adv-os       58 k
=====
事务概要
-----
安装 2 软件包
总下载: 192 k
安装大小: 720 k
确定吗? [y/N]: y
下载软件包:
(1/2): drpm-0.5.0-1.ky10.aarch64.rpm      778 kB/s | 58 kB    00:00
(2/2): createrepo_c-0.16.0-3.p01.ky10.aarch64.rpm 1.3 MB/s | 133 kB  00:00
-----
总计                                     1.8 MB/s | 192 kB  00:00
运行事务检查
```

②运行”yum install rpm-build”安装 rpmbuild 包（不同硬件平台可能安装 rpmbuild 的命令不一样，请根据实际情况安装）。

PC> yum install rpm-build

```
[ccore@localhost build-kylin-server]$ sudo yum install rpm-build
上次元数据过期检查: 0:10:52 前, 执行于 2023年09月08日 星期五 12时26分25秒。
依赖关系解决。

=====
Package                               Architecture Version                               Repository                               Size
-----
安装:
rpm-build                             aarch64     4.15.1-37.p02.se.01.ky10             ks10-adv-updates                         92 k
升级:
python3-rpm                           aarch64     4.15.1-37.p02.se.01.ky10             ks10-adv-updates                         74 k
rpm                                     aarch64     4.15.1-37.p02.se.01.ky10             ks10-adv-updates                        455 k
rpm-libs                              aarch64     4.15.1-37.p02.se.01.ky10             ks10-adv-updates                        325 k
安装依赖关系:
chrpath                               aarch64     0.16-12.ky10                          ks10-adv-os                             26 k
gcc-c++                               aarch64     7.3.0-20220207.45.ky10               ks10-adv-os                             7.3 M
libstdc++-devel                       aarch64     7.3.0-20220207.45.ky10               ks10-adv-os                             1.1 M

事务概要
-----
安装 4 软件包
升级 3 软件包
总下载: 9.3 M
```

③如果装完以上依赖包之后运行 build-dd-kylin-server.sh 还提示报错（一般不会再报错），请根据错误提示安装对应的工具包。

4.运行 build-dd-kylin-server.sh，会在同目录下生成 **aaa-kylin-bbb.iso** 的镜像(**aaa** 为驱动名称，**bbb** 为日期)。

PC> ./build-dd-kylin-server.sh

```
[ccore@localhost build-kylin-server]$ ./build-dd-kylin-server.sh
output/aarch64/4.19.90-52.22.v2207.ky10.aarch64/ccusr.ko
rpm-build --define "_rpmdir rhdd3-iso-ccusr/rpm" --define "_arch aarch64" --define "_rpmfilename aarch64/ccusr-1.6-230908-4.19.90-52.22.v2207.ky10.aarch64.rpm" --define "module_dir /home/ccore/build-kylin-server/output/aarch64/4.19.90-52.22.v2207.ky10.aarch64" --define "module_name ccusr" --define "version 1.6" --define "release 230908" --define "kernel_version 4.19.90-52.22.v2207.ky10.aarch64" --target aarch64-kylin-linux --nodeps -bb rpm.spec
构建目标平台: aarch64-kylin-linux
为目标 aarch64-kylin-linux构建
正在执行 (prep): /bin/sh -e /var/tmp/rpm-tmp.Qyitvd
+ umask 022
+ cd /home/ccore/rpmbuild/BUILD
+ RPM_EC=0
++ jobs -p
+ exit 0
正在执行 (build): /bin/sh -e /var/tmp/rpm-tmp.MPOKtq
+ umask 022
+ cd /home/ccore/rpmbuild/BUILD
+ RPM_EC=0
++ jobs -p
+ exit 0
正在执行 (install): /bin/sh -e /var/tmp/rpm-tmp.QSeWj
+ umask 022
+ cd /home/ccore/rpmbuild/BUILD
```

```
[ccore@localhost build-kylin-server]$ ls
build-dd-kylin-server.sh ccusr-kylin-230908.iso output rhdd3-iso-ccusr rhdd3-iso-vrc9k rhdd3-iso-vrcdrv rpm.spec
[ccore@localhost build-kylin-server]$
```

4.7. 拷贝驱动镜像

1.UOS Desktop:

将 4.6.1 小节打包好的 squashfs 文件（例如 ccusr-uos-230907.squashfs）拷贝到之前制作好的 U 盘启动盘系统的 **live** 目录下，接下来就可以进行在可引导阵列下安装系统了。

PC> cp ccusr-uos-230907.squashfs **XXX**/live/

```
ccore@ccore-PC: ~/Desktop/build-uos-desktop/uos$ cp ccusr-uos-230907.squashfs /media/ccore/uos/ 20/live/
ccore@ccore-PC: ~/Desktop/build-uos-desktop/uos$
ccore@ccore-PC: ~/Desktop/build-uos-desktop/uos$
```

2.UOS Server:

将 4.6.2 小节打包好的 iso 文件（例如 ccusr-uos-server-230907.iso）拷贝到之前制作好的 U 盘启动盘系统的**根目录**下（或者其它 vfat 格式的 U 盘根目录下），接下来就可以进行在可引导阵列下安装系统了。

PC> cp ccusr-uos-server-230907.iso **XXX**/

```
[ccore@localhost build-uos-server]$ cp ccusr-uos-server-230907.iso /media/ccore/UOS/
[ccore@localhost build-uos-server]$
```

3.Kylin Desktop:

①将 4.5 小节编译好的 ko 驱动文件拷贝到之前制作好的 U 盘启动盘系统的根目录下

PC> cp ccusr.ko XXX/

```
kylin@kylin:~/ccusr-src-g4634a93$  
kylin@kylin:~/ccusr-src-g4634a93$ cp ccusr.ko /media/kylin/KYLIN-DESKT/  
kylin@kylin:~/ccusr-src-g4634a93$  
kylin@kylin:~/ccusr-src-g4634a93$
```

②将 load-ccusr.sh 驱动加载脚本拷贝到之前制作好的 U 盘启动盘系统的根目录下

PC> cp load-ccusr.sh XXX/

接下来就可以在可引导阵列下安装系统了。

4.Kylin Server

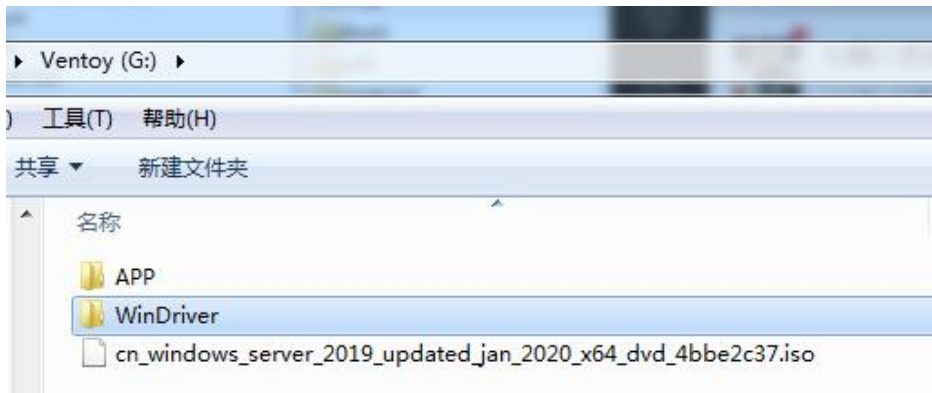
将 4.6.4 小节打包好的 iso 文件（例如 ccusr-kylin-230908.iso）拷贝到之前制作好的 U 盘启动盘系统的根目录下（或者其它 vfat 格式的 U 盘根目录下），接下来就可以进行在可引导阵列下安装系统了。

PC> cp ccusr-kylin-230908.iso XXX/

```
[ccore@localhost build-kylin-server]$ cp ccusr-kylin-230908.iso /run/media/ccore/KYLIN-SERVE/  
[ccore@localhost build-kylin-server]$
```

5.Windows

将 RAID 卡驱动（“3-软件环境\3-4-Windows\WinDriver”）整个目录拷贝到 U 盘启动盘。



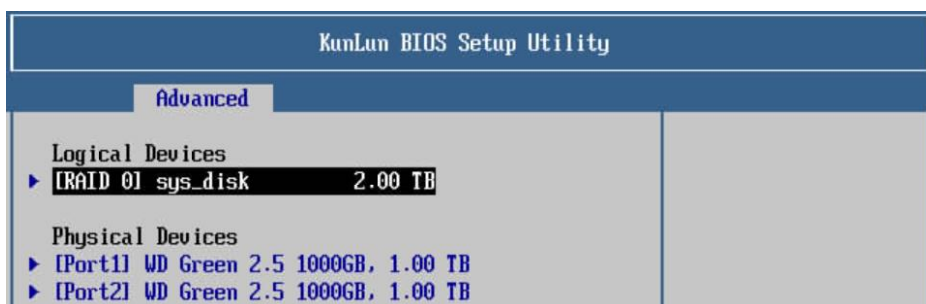
5. 安装系统镜像到可引导阵列

说明：简单来说，要想在 RAID 卡阵列上安装系统并使用，第一，RAID 卡及其上的阵列需要能够在 BIOS 下被识别，然后才能从阵列盘中加载系统。第二，加载的系统上需要包含了 RAID 卡驱动并且能够开机自动加载驱动。

这样从 BIOS 阶段跳到系统阶段的过程阵列盘的数据才能够正确加载，并引导系统正常启动。

1.在 BIOS 下创建磁盘阵列（如您之前在系统上已经创建好了磁盘阵列，并且在 BIOS 下能够被识别到，也是可以用作可引导阵列的）：

不管是装什么系统到可引导阵列，**都必须要先在 BIOS 下创建磁盘阵列**（是指在 RAID 卡上之前并无可用的磁盘阵列的情况下，之前如果在系统下已经创建好了阵列也可以不用重复在 BIOS 下创建）。创建阵列的详细过程请参考 3.5 小节，以创建 RAID0 阵列作为系统盘为示例，创建好的 RAID0 阵列如下：

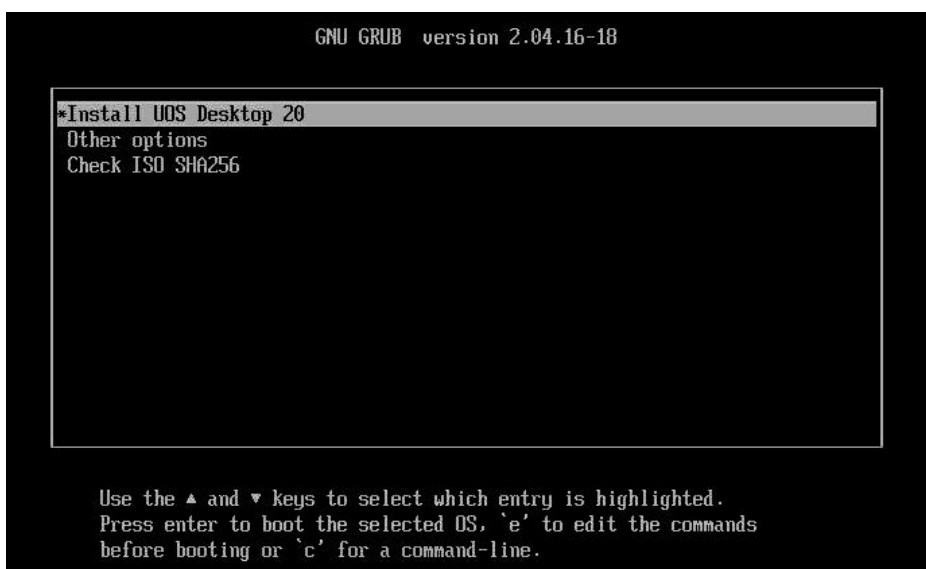


2.然后将之前安装好系统的主板直连普通硬盘拔掉，接下来将 U 盘启动盘的系统镜像装在可引导阵列上。

5.1. 安装 UOS Desktop

说明：本次演示安装的 UOS Desktop 系统版本是：统信 UOS 桌面专业版 1060

1.开机时选择启动设备为系统安装介质所在的设备（U 盘启动盘），系统启动后显示启动菜单。



此时请选择“Install UOS Desktop 20”选项，按回车键继续。

2.此时发现，选择盘符时可以识别到可引导阵列，选中可引导阵列作为系统盘。

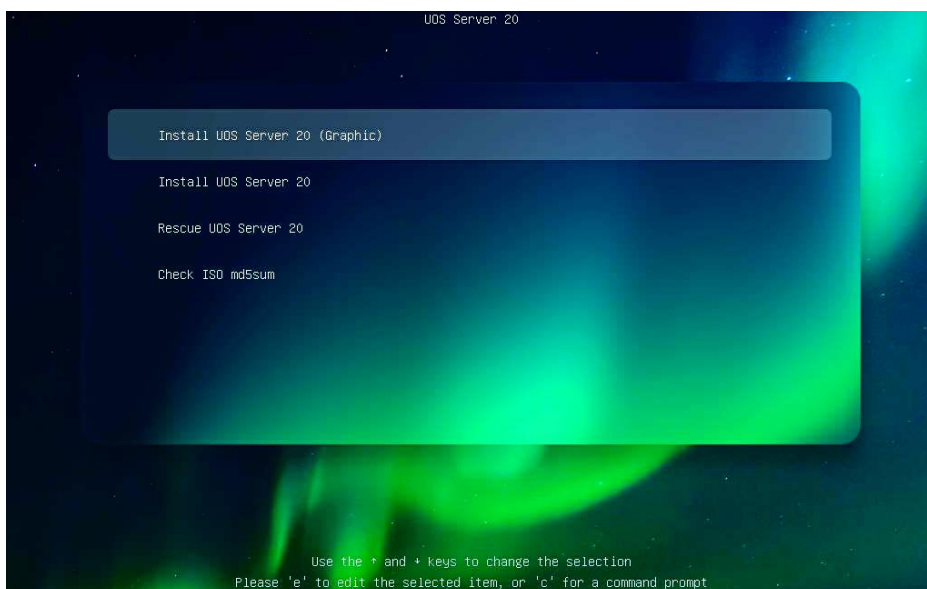


3.后续的安装步骤，请参考操作系统的安装手册进行操作。

5.2. 安装 UOS Server

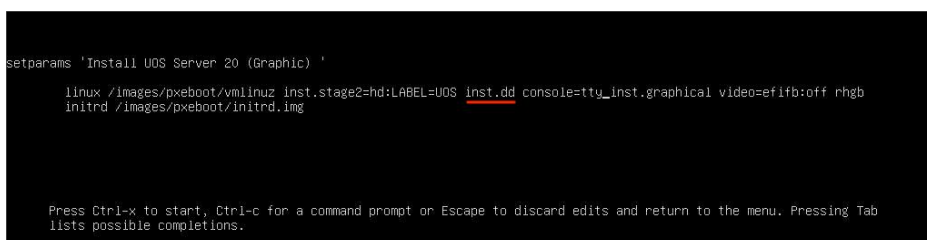
说明：本次演示安装的 UOS Server 系统版本是：统信服务器操作系统 V20（1060）

1.开机时选择启动设备为系统安装介质所在的设备（U 盘启动盘），系统启动后显示启动菜单。



2.修改启动参数:

在安装系统时需要加载 RAID 控制卡的驱动程序，安装程序才能识别到磁盘阵列。为此，在安装的启动界面，键盘输入”e”进入启动参数配置界面，然后添加”inst.dd”启动参数。然后按”ctrl + x”启动驱动安装。



3.选择驱动程序:

①安装系统启动后将进入“Driver disk device selection”提示界面。此界面会列出所有的存储卷供选择，请输入驱动盘所在的卷编号。如下图示例中，编号 4 为驱动光盘，则输入“4”后按回车键。

②安装程序将在指定的卷上搜索 ISO 镜像或驱动文件。当采用将驱动盘的 ISO 镜像复制到 U 盘上进行安装时，安装程序将列出 ISO 镜像文件进行进一步选择确认。选择完成后将列出驱动文件供选择，此时输入“1”并回车，确认加载驱动包：

③此时按 c 键回车继续安装程序。安装程序将加载驱动程序并开始正常安装流程。

```

[ OK ] Reached target System Initialization.
[ OK ] Reached target Basic System.
[ OK ] Created slice system-driverxx2updates.slice.
Starting Driver Update Disk UI on tty1...
DD: starting interactive mode

(Page 1 of 1) Driver disk device selection
/DEVICE TYPE LABEL UUID
1) /run/ins squashfs
2) loop1 ext4 1961c005-3ef5-4513-adfc-d33cab562004
3) dm-1 ext4 1961c005-3ef5-4513-adfc-d33cab562004
4) sdb4 vfat UOS B4FE-5315
5) loop0 squashfs
6) dm-0 ext4 1961c005-3ef5-4513-adfc-d33cab562004
7) sda2 xfs 7bafed65-2904-4cd2-9a90-1e85b828ab0b
8) sda3 LVM2_mem 2PgHh5-Yfzp-grSY-tQRX-Hr5u-9Dni-dkdn
9) sda1 vfat 79B2-56B3
# to select, 'r'-refresh, or 'c'-continue: 4
DD: Examining /dev/sdb4
mount: /media/DD-1: WARNING: source write-protected, mounted read-only.

(Page 1 of 1) Choose driver disk ISO file
1) /media/DD-1/ccusr-uos-server-231108.iso
# to select, or 'c'-continue: 1
DD: Examining /media/DD-1/ccusr-uos-server-231108.iso

(Page 1 of 1) Select drivers to install
1) [ ] /media/DD-2/rpms/aarch64/ccusr-1.0-231108-4.19.0-91.82.152.uelc20.aarch64.rpm
# to toggle selection, or 'c'-continue: 1

(Page 1 of 1) Select drivers to install
1) [x] /media/DD-2/rpms/aarch64/ccusr-1.0-231108-4.19.0-91.82.152.uelc20.aarch64.rpm
# to toggle selection, or 'c'-continue: c
DD: Extracting: ccusr-4.19.0-91.82.152.uelc20.aarch64

(Page 1 of 1) Driver disk device selection
/DEVICE TYPE LABEL UUID
1) /run/ins squashfs
2) loop1 ext4 1961c005-3ef5-4513-adfc-d33cab562004
3) dm-1 ext4 1961c005-3ef5-4513-adfc-d33cab562004
4) sdb4 vfat UOS B4FE-5315
5) loop0 squashfs
6) dm-0 ext4 1961c005-3ef5-4513-adfc-d33cab562004
7) sda2 xfs 7bafed65-2904-4cd2-9a90-1e85b828ab0b
8) sda3 LVM2_mem 2PgHh5-Yfzp-grSY-tQRX-Hr5u-9Dni-dkdn
9) sda1 vfat 79B2-56B3
# to select, 'r'-refresh, or 'c'-continue:

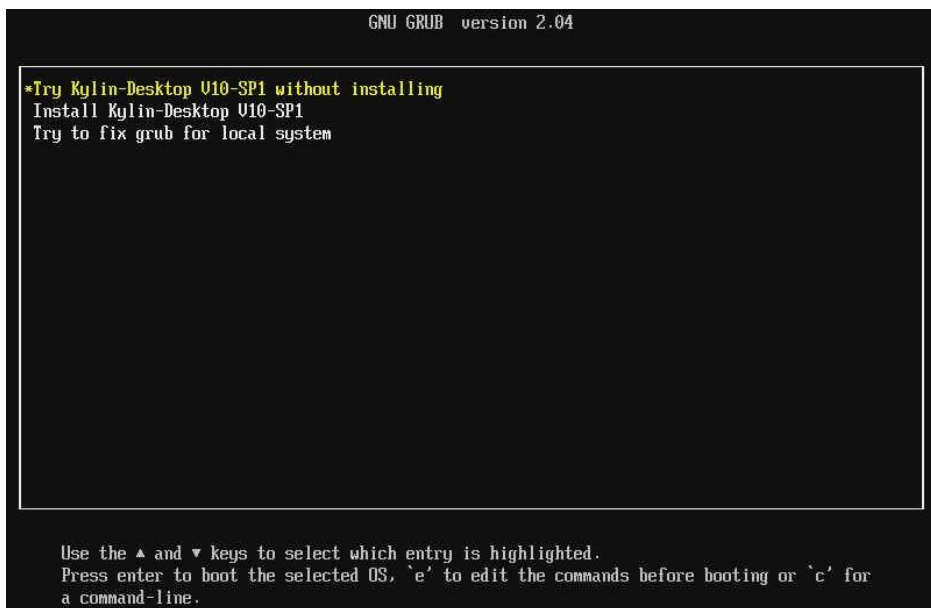
```

4.后续的安装步骤，请参考操作系统的安装手册进行操作（请选择 RAID 卡上的磁盘阵列作为系统盘）。

5.3. 安装 Kylin Desktop

说明：本次演示安装的 Kylin Desktop 系统版本是：银河麒麟桌面操作系统 V10

1.开机时选择启动设备为系统安装介质所在的设备（U 盘启动盘），系统启动后显示启动菜单。



此时请选择“Try Kylin-Desktop V10-SP1 without installing”选项，按回车键继续。

2.系统启动完成后进入系统桌面，此时点击鼠标右键选择“打开终端”打开一个终端窗口，然后运行加载 U 盘启动盘根目录的加载脚本 load-ccusr.sh（请确保 load-ccusr.sh 跟 ccusr.ko 在同一级目录下）：

```
PC> sudo /cdrom/load-ccusr.sh
```

```
kylin@kylin:~$ sudo /cdrom/load-ccusr.sh
Driver loaded successfully.
kylin@kylin:~$
kylin@kylin:~$ lsmod | grep "ccusr"
ccusr                20480  0
kylin@kylin:~$
kylin@kylin:~$ modinfo ccusr
filename:            /lib/modules/5.4.18-85-generic/kernel/drivers/scsi/ccusr.ko
version:             1.0 b1935-g4634a93
license:             GPL
description:         CCUSR HW-RAID Adapter Driver
author:              VolansComputer
srcversion:          617FCFEEC4D9B191EE52F88
alias:               pci:v0000FACEd00006316sv*sd*bc*sc*i*
alias:               pci:v00009000d00008116sv*sd*bc*sc*i*
alias:               pci:v00009000d00008108sv*sd*bc*sc*i*
depends:
name:                ccusr
vermagic:            5.4.18-85-generic SMP mod_unload modversions aarch64
parm:                use_msi:int
kylin@kylin:~$
```

显示驱动成功加载后可关闭终端窗口，也可以通过 lsmod, modinfo 来辅助确认驱动是否成功被加载。

3.双击桌面上的安装程序图标开始安装系统。



4.后续的安装步骤，请参考操作系统的安装手册进行操作（请选择 RAID 卡上的磁盘阵列作为系统盘）。

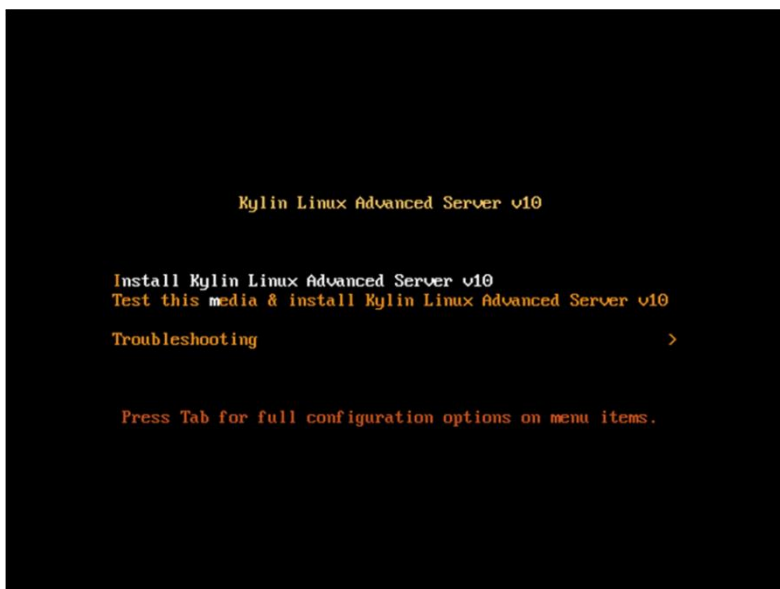


5.4. 安装 Kylin Server

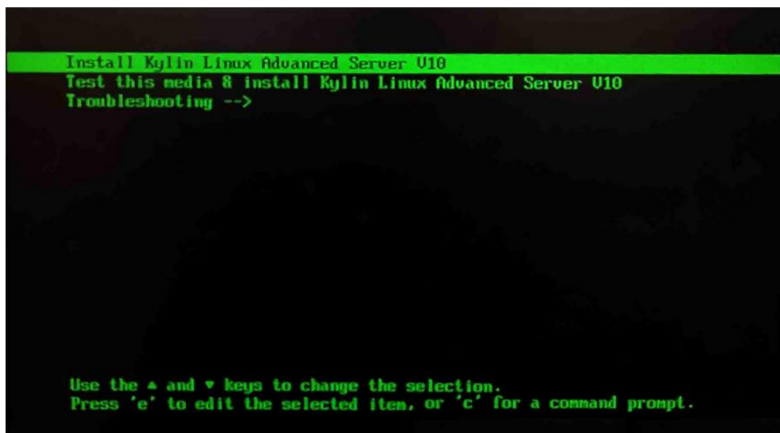
说明：本次演示安装的 Kylin Server 系统版本是：银河麒麟高级服务器操作系统 V10

1.开机时选择启动设备为系统安装介质所在的设备（U 盘启动盘），系统启动后显示启动菜单。

①x86_64 平台启动界面如下：



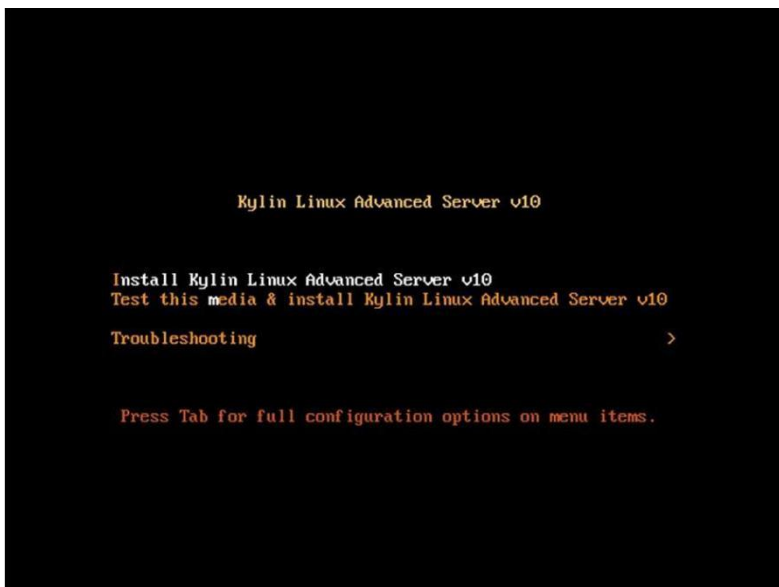
②arm64/mips64 平台启动界面如下：



2.修改启动参数：

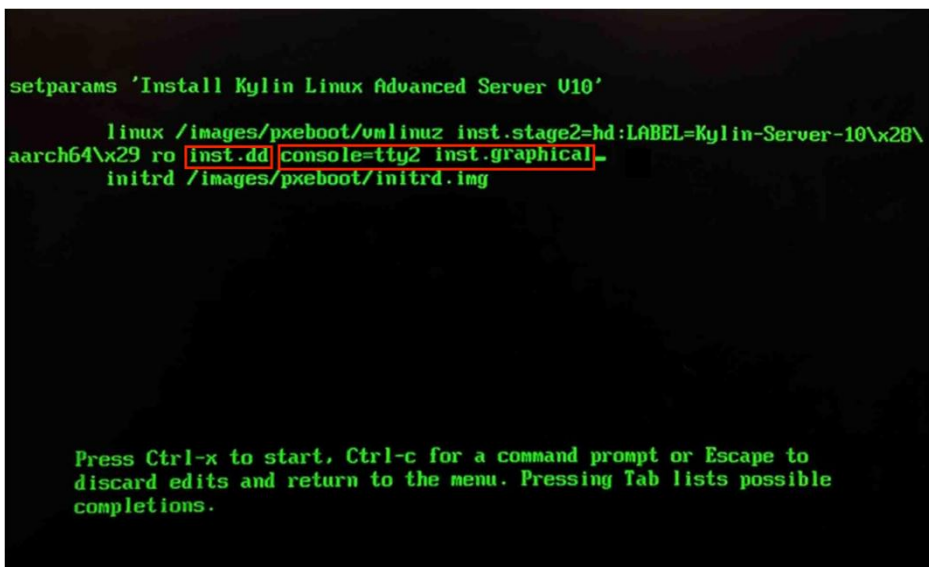
在安装系统时需要加载 RAID 控制卡的驱动程序，安装程序才能识别到磁盘阵列。为此，需要在开始安装之前添加"inst.dd"启动参数。根据具体平台或安装版本的不同，指定安装参数的方法有所不同，请根据屏幕提示操作。

①x86_64 平台上，选择第一项安装菜单后，可以按 Tab 键添加启动参数，回车键启动安装。如下图所示：



②在 arm64、mips64 平台上，选择第一项安装菜单后，按 e 键修改启动参数，按 ctrl + x 启动安装。

arm64 版本在部分平台上添加“inst.dd”参数后可能无法显示界面，需要再额外添加“console=tty0”参数；而在华为鲲鹏平台上需要再额外添加“console=tty2 inst.graphical”参数。如下图所示：



3.选择驱动程序:

①安装系统启动后将进入“Driver disk device selection”提示界面。此界面会列出所有的存储卷供选择，请输入驱动盘所在的卷编号。如下图示例中，编号2为驱动光盘，则输入“2”后按回车键。

```
[ OK ] Reached target Paths.
[ OK ] Reached target Basic System.
[ OK ] Started Device-Mapper Multipath Device Controller.
      Starting Open-iSCSI...
[ OK ] Started Open-iSCSI.
      Starting dracut initqueue hook...
      Mounting Configuration File System...
[ OK ] Mounted Configuration File System.
[ 46.006909] dracut-initqueue[948]: Warning: Can't get kickstart from /dev/sr1
:/ks.cfg
[ 46.259208] dracut-initqueue[948]: Warning: Can't get kickstart from /dev/sr1
:/ks.cfg
[ OK ] Created slice system-driver\x2dupdates.slice.
      Starting Driver Update Disk UI on tty1...
DD: starting interactive mode

(Page 1 of 1) Driver disk device selection
# to select, 'r'-refresh, or 'c'-continue: _
```

#	DEVICE	TYPE	LABEL	UUID
1)	sr0	iso9660	Kylin-Server-10\x28x	2020-04-29-16-13-36-00
2)	sr1	iso9660	OEMDRV	2021-01-04-11-43-34-00
3)	sda1	xfs		2b3fdc6f-f5e4-4e5f-96e1-44054be7f0fd
4)	sda2	swap		1ea31b85-0f92-4803-a895-1ba05ffdaa7a
5)	sda3	xfs		0b49b1a4-5e78-48cf-81a7-27830cc36375
6)	sda5	xfs		94addf06-af78-4b55-8950-f06aa42a74a8

②安装程序将在指定的卷上搜索 ISO 镜像或驱动文件。当采用将驱动盘的 ISO 镜像复制到 U 盘上进行安装时，安装程序将列出 ISO 镜像文件进行进一步选择确认。选择完成后将列出驱动文件供选择，示例如下：

```
[ OK ] Mounted Configuration File System.
[ 46.006909] dracut-initqueue[948]: Warning: Can't get kickstart from /dev/sr1
:/ks.cfg
[ 46.259208] dracut-initqueue[948]: Warning: Can't get kickstart from /dev/sr1
:/ks.cfg
[ OK ] Created slice system-driver\x2dupdates.slice.
      Starting Driver Update Disk UI on tty1...
DD: starting interactive mode

(Page 1 of 1) Driver disk device selection
# to select, 'r'-refresh, or 'c'-continue: 2
DD: Examining /dev/sr1
mount: /dev/sr1 is write-protected, mounting read-only

(Page 1 of 1) Select drivers to install
# to toggle selection, or 'c'-continue: _
```

#	FILE
1)	/media/DD-1/rpms/x86_64/vrcdrv-1.0-201021-4.19.90-vhulk2001.1.0.0026.ns7
2)	.15.x86_64.rpm

③此时输入“1”并回车，确认加载驱动包：

```

[ OK ] Created slice system-driver\x2dupdates.slice.
Starting Driver Update Disk UI on tty1...
DD: starting interactive mode

(Page 1 of 1) Driver disk device selection
  /DEVICE  TYPE      LABEL                      UUID
  1) sr0    iso9660    Kylin-Server-10\x28x      2020-04-29-16-13-36-00
  2) sr1    iso9660    OEMDRV                     2021-01-04-11-43-34-00
  3) sda1    xfs          2b3fdc6f-f5e4-4e5f-96e1-44054be7f0fd
  4) sda2    swap         1ea31b85-0f92-4803-a895-1ba05ffdaa7a
  5) sda3    xfs          0b49b1a4-5e78-48cf-81a7-27830cc36375
  6) sda5    xfs          94addf06-af78-4b55-8950-f06aa42a74a8
# to select, 'r'-refresh, or 'c'-continue: 2
DD: Examining /dev/sr1
mount: /dev/sr1 is write-protected, mounting read-only

(Page 1 of 1) Select drivers to install
  1) [ ] /media/DD-1/rpms/x86_64/vrcdrv-1.0-201021-4.19.90-vhulk2001.1.0.0026.ns7
     .15.x86_64.rpm
# to toggle selection, or 'c'-continue: 1
(Page 1 of 1) Select drivers to install
  1) [x] /media/DD-1/rpms/x86_64/vrcdrv-1.0-201021-4.19.90-vhulk2001.1.0.0026.ns7
     .15.x86_64.rpm
# to toggle selection, or 'c'-continue: _

```

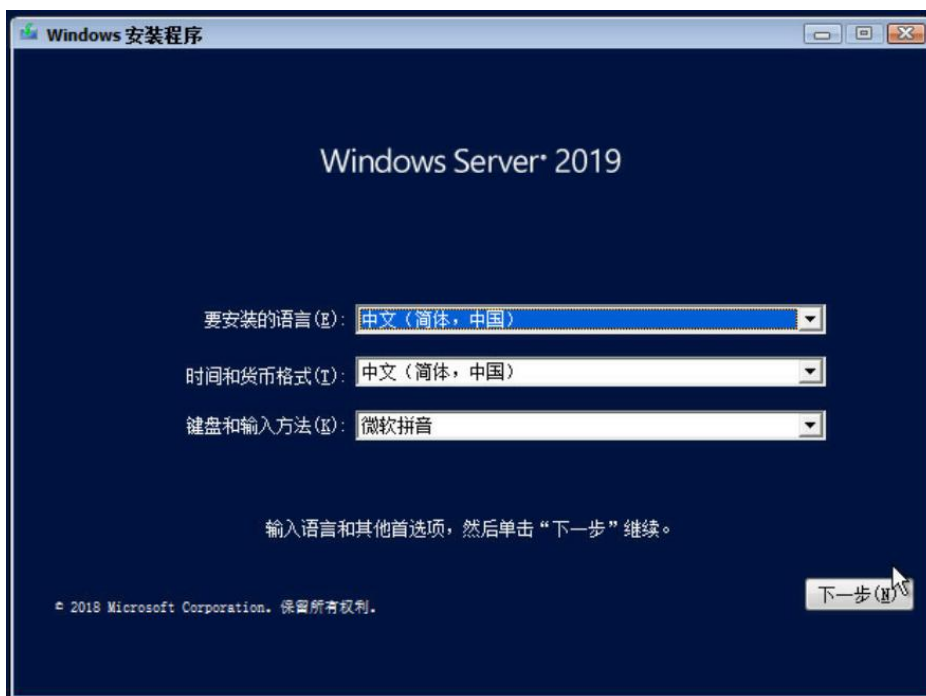
④此时按 c 键回车继续安装程序。安装程序将加载驱动程序并开始正常安装流程。

4. 后续的安装步骤，请参考操作系统的安装手册进行操作（请选择 RAID 卡上的磁盘阵列作为系统盘）。

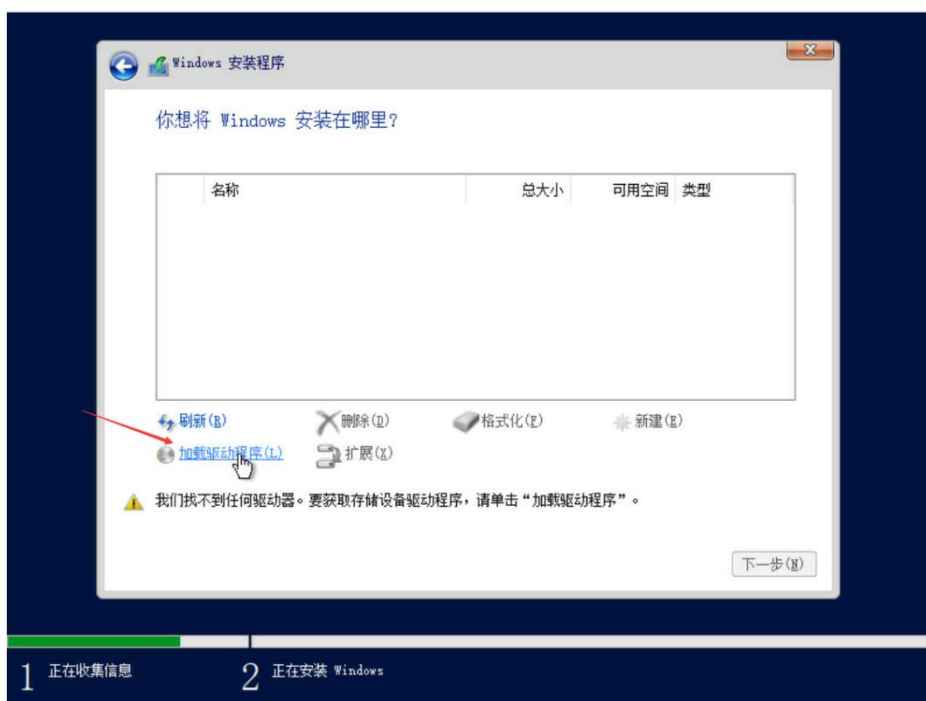
5.5. 安装 Windows

说明：本次演示安装的 Windows 系统版本是：Windows server 2019

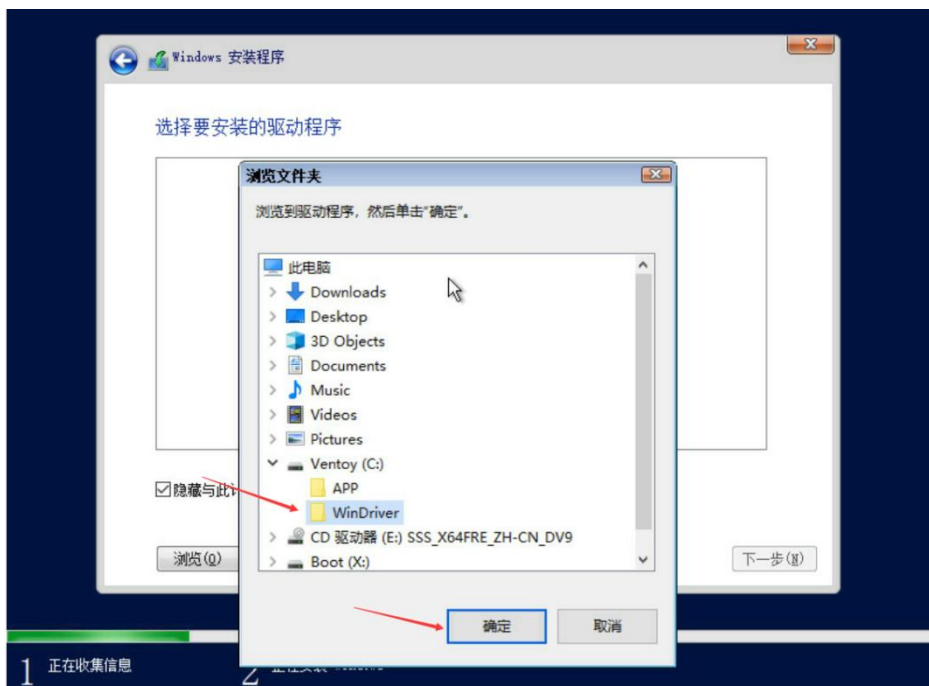
1. 开机时选择启动设备为系统安装介质所在的设备（U 盘启动盘），系统启动后显示启动菜单。



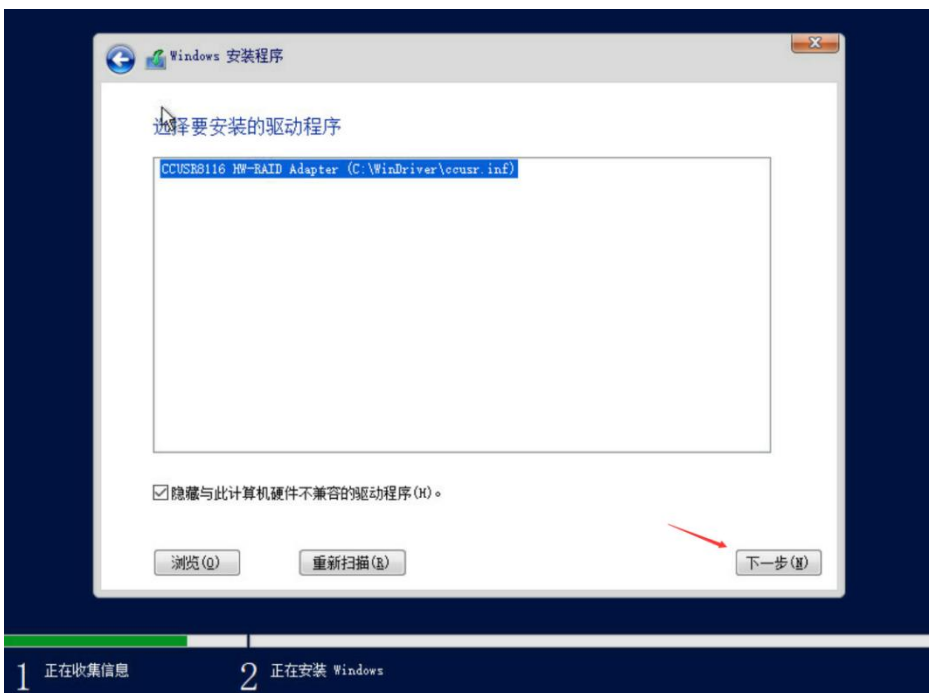
2.当安装程序提示指定 Windows 安装位置时, 选择"加载驱动程序".



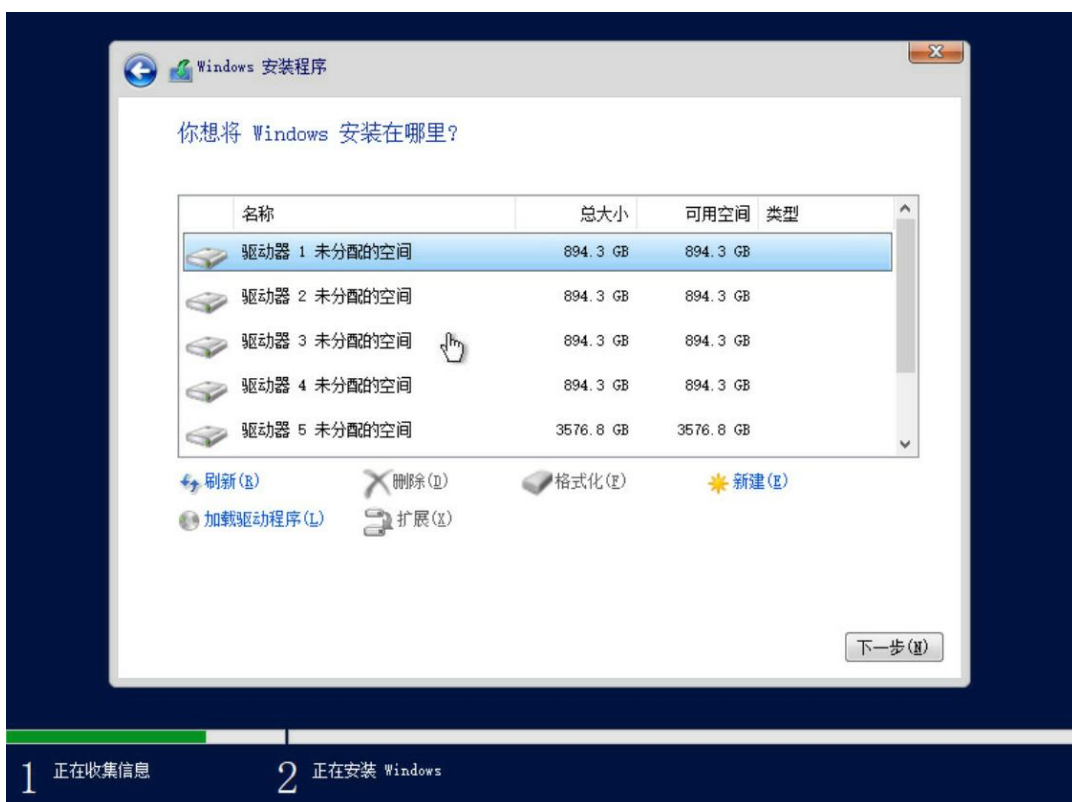
3.找到驱动程序所在目录, 点击“确定”。



4.Windows 会提示将要加载的驱动程序，此时单击"下一步"。



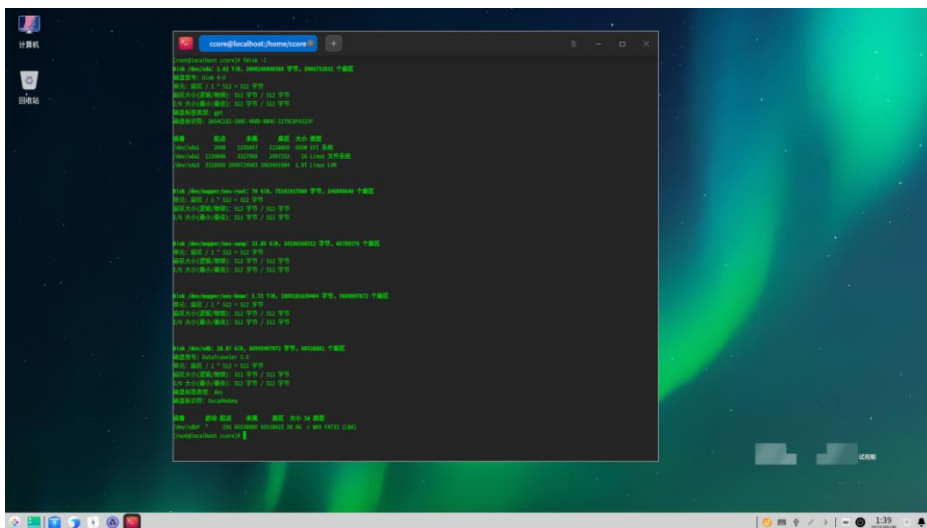
5.驱动加载成功后，可以看到前面所创建的阵列作为驱动器列出。



6.接下来按照正常安装 Windows 系统的步骤完成安装即可。

6. 可引导阵列系统使用

- 1.装好系统到可引导阵列后，按相关提示进行重启。
- 2.重启后进入 BIOS，将 BIOS 启动选项设置为可引导阵列优先，保存并退出。
- 3.等待进入系统界面（第一次加载系统可能会慢一点，请耐心等待一会）。



- 4.进入系统后，我们可以看到我们在 BIOS 下创建的可引导阵列就是系统盘了。
- 5.然后可以尝试进行多次断电/重启/关机测试验证稳定性。
- 6.若要对可引导阵列查询状态信息或其它功能测试，请参考《[2-1-CCUSR8116 RAID 控制卡用户手册](#)》。

7. 可引导阵列系统更新

7.1. 驱动版本更新

当系统安装在磁盘阵列上时，后续要更新 RAID 卡驱动程序时，需要同步更新 initramfs 以使用新版本驱动程序。不同系统更新方式不太一样，请按照如下步骤进行操作：

1.UOS Desktop:

①将新版本的驱动程序模块复制到/lib/modules/\${kernel-ver}/kernel/drivers/scsi 目录。

PC> cp ccusr.ko /lib/modules/\${kernel-ver}/kernel/drivers/scsi

②更新驱动依赖关系

PC> depmod -a

③ 重新生成 initramfs:

PC> update-initramfs -u -k \${kernel-ver}

其中\${kernel-ver}需替换为当前实际使用的内核版本号。

```
ccore@ccore-PC:~/ccusr-src-g4634a93$ uname -r
4.19.0-amd64-desktop
ccore@ccore-PC:~/ccusr-src-g4634a93$
ccore@ccore-PC:~/ccusr-src-g4634a93$ sudo cp ccusr.ko /lib/modules/4.19.0-amd64-desktop/kernel/drivers/scsi/
ccore@ccore-PC:~/ccusr-src-g4634a93$
ccore@ccore-PC:~/ccusr-src-g4634a93$ sudo update-initramfs -u -k 4.19.0-amd64-desktop
update-initramfs: Generating /boot/initrd.img-4.19.0-amd64-desktop
cryptsetup: WARNING: The initramfs image may not contain cryptsetup binaries
nor crypto modules. If that's on purpose, you may want to uninstall the
'cryptsetup-initramfs' package in order to disable the cryptsetup initramfs
integration and avoid this warning.
setupcon is missing. Please install the 'console-setup' package.
W: plymouth: The plugin label.so is missing, the selected theme might not work as expected.
W: plymouth: You might want to install the plymouth-themes package to fix this.
I: The initramfs will attempt to resume from /dev/sda7
I: (UUID=a68da10b-98b3-480c-b70e-0f22dd155fc5)
I: Set the RESUME variable to override this.
live-boot: core filesystems devices utils udev blockdev dns.
Adding IMA binaries
/etc/initramfs-tools/scripts/init-bottom/deepin-iso-verify: 1: /etc/initramfs-tools/scripts/init-bottom/deepin
-iso-verify: Syntax error: ")" unexpected
ccore@ccore-PC:~/ccusr-src-g4634a93$
```

2.UOS Server:

①将新版本的驱动程序模块复制到/lib/modules/\${kernel-ver}/kernel/drivers/scsi 目录。

PC> cp ccusr.ko /lib/modules/\${kernel-ver}/kernel/drivers/scsi

②更新驱动依赖关系

PC> depmod -a

③ 重新生成 initramfs:

PC> dracut /boot/initramfs-\${kernel-ver}.img \${kernel-ver} --force

```
[root@localhost ccore]# uname -r
4.19.0-91.82.152.uelc20.aarch64
[root@localhost ccore]#
[root@localhost ccore]# cp ccusr.ko /lib/modules/4.19.0-91.82.152.uelc20.aarch64/kernel/drivers/scsi/
[root@localhost ccore]#
[root@localhost ccore]# dracut /boot/initramfs-$(uname -r).img $(uname -r) --force
[root@localhost ccore]#
```

其中\${kernel-ver}需替换为当前实际使用的内核版本号。

3.Kylin Desktop:

①将新版本的驱动程序模块复制到/lib/modules/\${kernel-ver}/kernel/drivers/scsi 目录。

PC> cp ccusr.ko /lib/modules/\${kernel-ver}/kernel/drivers/scsi

②更新驱动依赖关系

PC> depmod -a

③ 重新生成 initramfs:

PC> update-initramfs -u -k \${kernel-ver}

其中\${kernel-ver}需替换为当前实际使用的内核版本号。

```
root@ccore-pc:/home/ccore# uname -r
5.4.18-85-generic
root@ccore-pc:/home/ccore# cp ccusr.ko /lib/modules/5.4.18-85-generic/kernel/drivers/scsi/
root@ccore-pc:/home/ccore#
root@ccore-pc:/home/ccore# update-initramfs -u -k 5.4.18-85-generic
update-initramfs: Generating /boot/initrd.img-5.4.18-85-generic
I: The initramfs will attempt to resume from /dev/sda6
I: (UUID=cba0da3-b9fd-4e32-a186-1daa2bd0dd03)
I: Set the RESUME variable to override this.
root@ccore-pc:/home/ccore#
```

4.Kylin Server:

①将新版本的驱动程序模块复制到/lib/modules/\${kernel-ver}/kernel/drivers/scsi 目录。

PC> cp ccusr.ko /lib/modules/\${kernel-ver}/kernel/drivers/scsi

②更新驱动依赖关系

PC> depmod -a

③重新生成 initramfs:

PC> mkinitrd -f --preload=ccusr /boot/initramfs-\${kernel-ver}.img \${kernel-ver}

其中\${kernel-ver}需替换为当前实际使用的内核版本号。

```
[root@localhost ccore]# uname -r
4.19.90-52.22.v2207.ky10.aarch64
[root@localhost ccore]#
[root@localhost ccore]# cp ccusr.ko /lib/modules/4.19.90-52.22.v2207.ky10.aarch64/kernel/drivers/scsi/
[root@localhost ccore]#
[root@localhost ccore]# mkinitrd -f --preload=ccusr /boot/initramfs-4.19.90-52.22.v2207.ky10.aarch64.img
4.19.90-52.22.v2207.ky10.aarch64
Creating: target[kernel]dracut args[basicmodules]
[root@localhost ccore]#
```

7.2. 内核版本更新

在需要进行内核版本升级的场景下，对应的 RAID 卡驱动也要同步编译更新。不然因为 RAID 卡驱动 ko 文件跟内核版本不匹配，导致无法正常加载 RAID 卡驱动，进而无法配置/使用 RAID 卡下的相关磁盘阵列。

以 Kylin Server 版本为例（其它系统版本类似），对于内核版本升级的驱动更新步骤如下：

①查看当前系统有两个内核版本，当前是 A 版本，想升级到 B 版本：

```
[root@localhost ccusr-src-2117-g7366641a]# uname -r
4.19.90-52.22.v2207.ky10.aarch64
[root@localhost ccusr-src-2117-g7366641a]#
[root@localhost ccusr-src-2117-g7366641a]# ls /lib/modules/
4.19.90-52.22.v2207.ky10.aarch64 4.19.90-52.33.v2207.ky10.aarch64
[root@localhost ccusr-src-2117-g7366641a]#
```

②修改驱动 Makefile 文件，手动指定为要升级的内核版本：

```
ifneq ($(KERNELRELEASE),)
include Kbuild
else

default: modules;
.PHONY: force
Makefile force;

#KERNELDIR ?= /lib/modules/$(shell uname -r)/build
KERNELDIR ?= /lib/modules/4.19.90-52.33.v2207.ky10.aarch64/build

%: force
	@$(MAKE) -C $(KERNELDIR) M=$$PWD $@

endif

"Makefile" 15L, 267C
```

③编译驱动、拷贝驱动以及更新驱动依赖关系：

PC> make

PC> cp ccusr.ko /lib/modules/\${kernel-ver}/kernel/drivers/scsi

PC> depmod -av \${kernel-ver}

```
[root@localhost ccusr-src-2117-g7366641a]# make
make[1]: 进入目录 /usr/src/kernels/4.19.90-52.33.v2207.ky10.aarch64
CC [M] /root/ccusr-src-2117-g7366641a/ccusr.o
Building modules, stage 2.
MODPOST 1 modules
CC /root/ccusr-src-2117-g7366641a/ccusr.mod.o
LD [M] /root/ccusr-src-2117-g7366641a/ccusr.ko
make[1]: 离开目录 /usr/src/kernels/4.19.90-52.33.v2207.ky10.aarch64
[root@localhost ccusr-src-2117-g7366641a]# ls
ccusr.o ccusr.h ccusr.ko ccusr.mod.o ccusr.mod.o ccusr.o Kbuild Makefile modules.order Module.symvers
[root@localhost ccusr-src-2117-g7366641a]#
[root@localhost ccusr-src-2117-g7366641a]# cp ccusr.ko /lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/drivers/scsi/
[root@localhost ccusr-src-2117-g7366641a]#
[root@localhost ccusr-src-2117-g7366641a]# depmod -av 4.19.90-52.33.v2207.ky10.aarch64
/lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/arch/arm64/crypto/sha2-ce.ko.xz needs "sha256_block_data_order": /lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/arch/arm64/crypto/sha256-arm64.ko.xz
/lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/arch/arm64/crypto/sha512-ce.ko.xz needs "sha512_block_data_order": /lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/arch/arm64/crypto/sha512-arm64.ko.xz
/lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/arch/arm64/crypto/sha3-ce.ko.xz needs "crypto_sha_init": /lib/modules/4.19.90-52.33.v2207.ky10.aarch64/kernel/crypto/sha3_generic.ko.xz
```

④重新生成 initramfs:

PC> mkinitrd -f --preload=ccusr /boot/initramfs-**\${kernel-ver}**.img **\${kernel-ver}**

```
[root@localhost ccusr-src-2117-g7366641a]# mkinitrd -f --preload=ccusr /boot/initramfs-4.19.90-52.03.v2207.ky10.aarch64.img 4.19.90-52.03.v2207.ky10.aarch64
Creating: target[kernel]dracut args[basicmodules]
[root@localhost ccusr-src-2117-g7366641a]#
```