Programa de Ingeniería de Sistemas
Laboratorio de Sistemas Operativos
Proyecto Tercer Parcial-2-2025 Grupo A

Fecha: 12 de noviembre de 2025

Planteamiento del problema

Deberá desarrollar un programa en C que tome como entrada, por línea de comandos, la ruta de un dispositivo tipo disco, e imprima su tabla de particiones. Se deberá mostrar la siguiente información:

- Esquema de particionado (MBR, GPT)
- Para cada partición:
  - Tipo de partición
  - Sector inicial
  - Sector final
  - Tamaño en bytes

Entrada del programa

El programa deberá recibir, por línea de comandos, la ruta del dispositivo (tipo disco duro), del cual se desea imprimir la tabla de particiones. **Es importante tener en cuenta que el programa no deberá modificar (escribir) ninguna información en el dispositivo especificado por el usuario, ya que puede causar la pérdida total de la información de ese disco**.
En GNU/Linux, se puede determinar los discos conectados usando el comando **lsblk** (Con permisos de superusuario):

```
root@debian:/home/ingesis# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0   40G  0 disk
├─sda1   8:1    0   39G  0 part /
├─sda2   8:2    0    1K  0 part
└─sda5   8:5    0  975M  0 part [SWAP]
sdb      8:16   0   40G  0 disk
├─sdb1   8:17   0 18,6G  0 part /mnt/datos1
└─sdb2   8:18   0 21,3G  0 part /mnt/datos_windows
sr0     11:0    1 1024M  0 rom
root@debian:/home/ingesis#
```

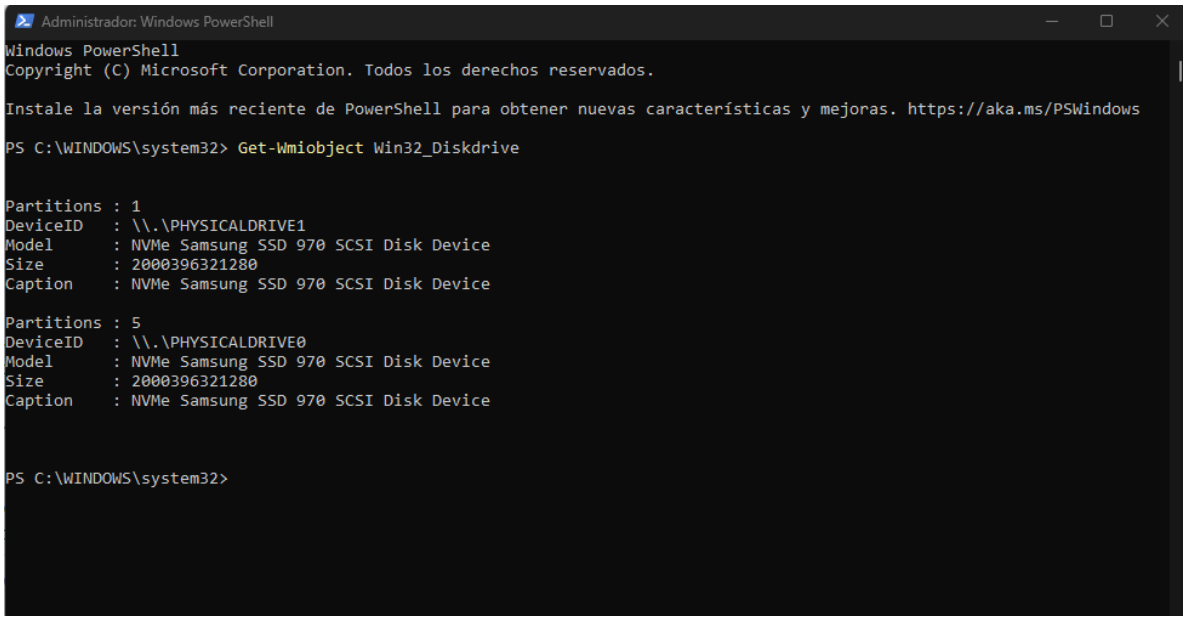
En este caso, el sistema tiene conectados dos discos, llamados **/dev/sda** y **/dev/sdb**.

Se deberá ejecutar el programa con permisos de superusuario, iniciando un shell como root (**sudo su**), o mediante el comando **sudo**. Luego se ejecutará el programa como se indica a continuación:

```
#Listar particiones de un disco SATA
sudo ./listpart /dev/sda
#Listar particiones de un disco NVME
sudo ./listpart /dev/nvme0n1
```

Para sistemas operativos Windows, primero se deberá determinar los discos existentes en el sistema, usando un PowerShell con permisos de administrador y ejecutando el comando:

```
Get-Wmiobject Win32_Diskdrive
```



Se deberá ejecutar el programa en una terminal de Windows (cmd) con permisos de administrador:

```
listpart.exe \\.\PHYSICALDRIVE0
```

Salida del programa

El programa deberá imprimir la información del dispositivo especificado: Su tipo de inicialización (MBR, GPT), seguido por la tabla de particiones. En la tabla de particiones se deberá imprimir el bloque (sector) de inicio y fin, y en el caso de discos GPT, opcionalmente se debe imprimir el tipo (el GUID o una descripción), y el nombre de la partición, si se encuentra definido.

```
Disk initialized as GPT.
MBR Partition Table
Start LBA    End LBA      Type
------------ ------------ ------------------------------------
           1   3907029167 GPT Protective MBR
------------ ------------ ------------------------------------
GPT header
Revision: 0x10000
First usable lba: 34
Last usable lba: 3907029134
Disk GUID: 0421bb1e-e8d3-4831-abc7-cf8750ee9bf5
Partition entry LBA: 2
Number of partition entries: 128
Size of a partition entry: 128
Total of partition table entries sectors: 32
Size of a partition descriptor: 128
Start LBA    End LBA      Size         Type                                 Partition name
------------ ------------ ------------ ------------------------------------ ------------------------------------
        2048       534527    272629760 EFI System partition                 EFI system partition
      534528       567295     16777216 Microsoft Reserved Partition (MSR)   Microsoft reserved partition
      567296   2614239231 1338200031232 Basic data partition                  Basic data partition
  2614239232   2616010751    907018240 Windows Recovery Environment
  2616010752   3500224511 452717445120 Linux filesystem data
  3500224512   3516225535   8192524288 Swap partition
------------ ------------ ------------ ------------------------------------ ------------------------------------
```

Para otro disco, la información que se muestra es la siguiente:

```
Disk initialized as GPT.
MBR Partition Table
Start LBA    End LBA      Type
------------ ------------ ------------------------------------
           1   4294967295 GPT Protective MBR
------------ ------------ ------------------------------------
GPT header
Revision: 0x10000
First usable lba: 34
Last usable lba: 3907029134
Disk GUID: 78a977c6-2cac-4f14-930d-417a5dffa364
Partition entry LBA: 2
Number of partition entries: 128
Size of a partition entry: 128
Total of partition table entries sectors: 32
Size of a partition descriptor: 128
Start LBA    End LBA      Size         Type                                 Partition name
------------ ------------ ------------ ------------------------------------ ------------------------------------
          34        32767     16759808 Microsoft Reserved Partition (MSR)   Microsoft reserved partition
       32768   3516325887 1800342077440 Basic data partition                  Basic data partition
------------ ------------ ------------ ------------------------------------ ------------------------------------
```

Lógica del programa

El programa deberá implementar una lógica similar a la que se presenta a continuación. Esta lógica se puede modularizar en subrutinas.

```
leer el primer sector del dispositivo (512 bytes)
// Sin importar el tipo de inicialización, todo disco
// contiene una tabla de particiones tipo MBR en el
// primer sector (512 bytes) del disco.
// La tabla de particiones se encuentra en un desplazamiento
// de 446 bytes desde el inicio del sector de 512 bytes
es_gpt = verdadero
num_particiones_mbr = 0
ultimo_tipo = 0
para cada part:mbr_partition_descriptor en la tabla de particiones:
  ultimo_tipo = part->type
  si part->type == 0
  cancelar //break
  fin si
  // Hay una partición definida
  num_particiones_mbr = num_particiones_mbr  + 1
  // Si hay mas de una particion definida, no es GPT
  si num_particiones_mbr > 1
    es_gpt = falso
  fin si
  // Imprimir información de la partición:
  fin_lba = part->start_lba + part->lba_sector_count
  size = // Calcular el tamaño en bytes: (fin - inicio) * 512
  // Obtener la cadena del tipo correspondiente: str_tipo
  imprimir inicio_lba, fin_lba, size, str_tipo
fin para

//Es un disco MBR, terminar.
si num_particiones_mbr != 1 || ultimo_tipo != MBR_TYPE_GPT || !es_gpt
 terminar
fin si
// Leer imprimir el encabezado MBR
leer header: el segundo sector del disco (sector 1)

// Imprimir la información almacenada en el encabezado
imprimir revisión
imprimir primer sector usable LBA
imprimir último sector usable LBA
// Convertir el GUID a un string: guid_str
imprimir guid_str
imprimir número de particiones de la tabla
imprimir el tamaño de una entrada de la tabla (partition_entry_lba)

// Calcular la cantidad de sectores a leer
```

```
// Cada sector de 512 bytes almacena
// 4 descriptores de partición (campo partition_entry_size)
sectores = (header->partition_entry_size * partition_entres) / 512

//Leer tantos sectores como entradas se encuentren definidas
// La lectura se debe terminar cuando se encuentre
// la primera entrada nula (tipo = 0)
para sector: header->partition_entry_lba hasta
    header->partition_entry_lba + sectores
    // Leer un sector
    leer sect
    // Cada sector contiene cuatro descriptores de particion
    para i = 0 hasta 4
      desc = sect[i]
      si desc->type == 0
        cancelar // break
      fin si
      size = // Calcular el tamaño en bytes: (fin - inicio) * 512
      // Obtener la cadena del tipo correspondiente: str_type
      imprimir desc->start_lba, desc->end_lba, desc->size, str_type
    fin para
fin para
```

¿Qué se debe entregar?

Basados en el código base (archivo **listpart_codigo_base.zip**), ubicar dicho código en una carpeta denominada **lsoA-proy03-g0X**, (donde X corresponde al número de grupo asignado en este curso), crear la siguiente estructura de carpetas:

```
lsoA-proy03-g0x/
└── src
```

En la subcarpeta src/, deben incluir los archivos .c, .h y el archivo Makefile. Antes de enviar los archivos fuente de la solución, limpiar la carpeta src/ mediante el comando "**make clean**". Comprimir la carpeta y subir el archivo comprimido (en formato .rar o .zip) mediante el enlace que se defina a la plataforma Univirtual.

Fecha de entrega: 10 de diciembre de 2025, hasta las 9:00 am.

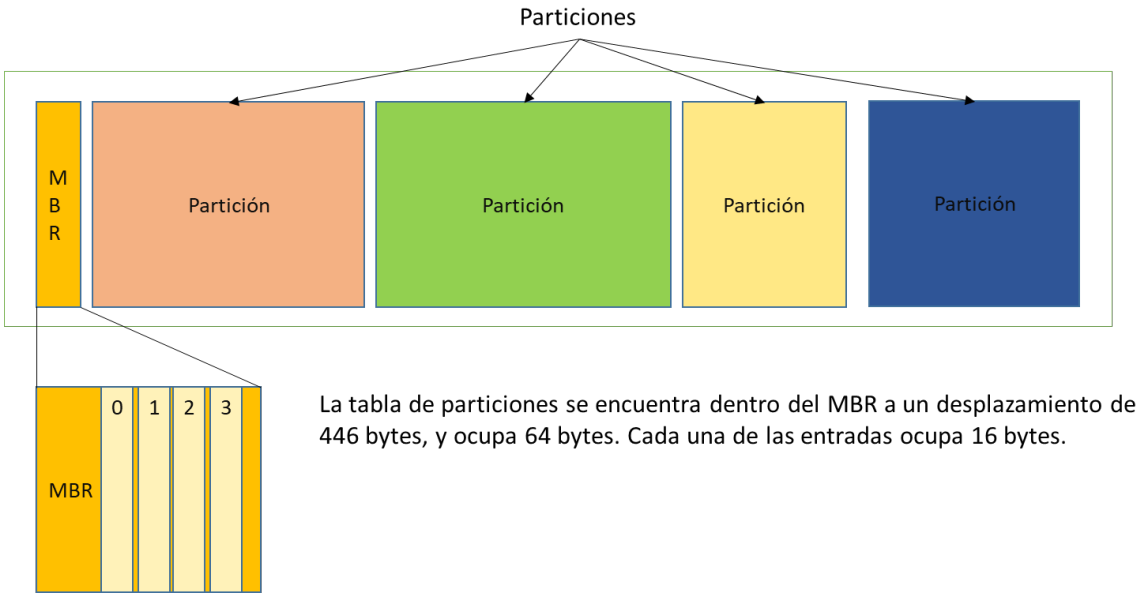
Esquemas de particionado de discos duros

Los discos duros son elementos fundamentales de cualquier sistema computacional. Se usan para almacenar el sistema operativo, los programas y los archivos creados por los usuarios. Antes de usar un disco duro, se debe inicializar y luego se crean una o más particiones que almacenarán los archivos y directorios. Cada partición puede tener un sistema de archivos diferente. Por ejemplo, para sistemas Windows se usa **NTFS** en particiones de disco y **FAT** para medios extraíbles. Por su parte, GNU/Linux usa alguna variante del sistema de archivos EXT2, generalmente EXT4.

Existen dos tipos de esquemas de particionado: MBR (Master Boot Record) y GPT (GUID Partition Table). El primero se usaba en sistemas con BIOS, y el segundo se utiliza en la actualidad en sistemas con UEFI. Cuando se conecta un nuevo disco duro al computador, se realiza el proceso de ***inicializar*** el disco, es decir, definir cuál de los dos esquemas de particionado se usará.

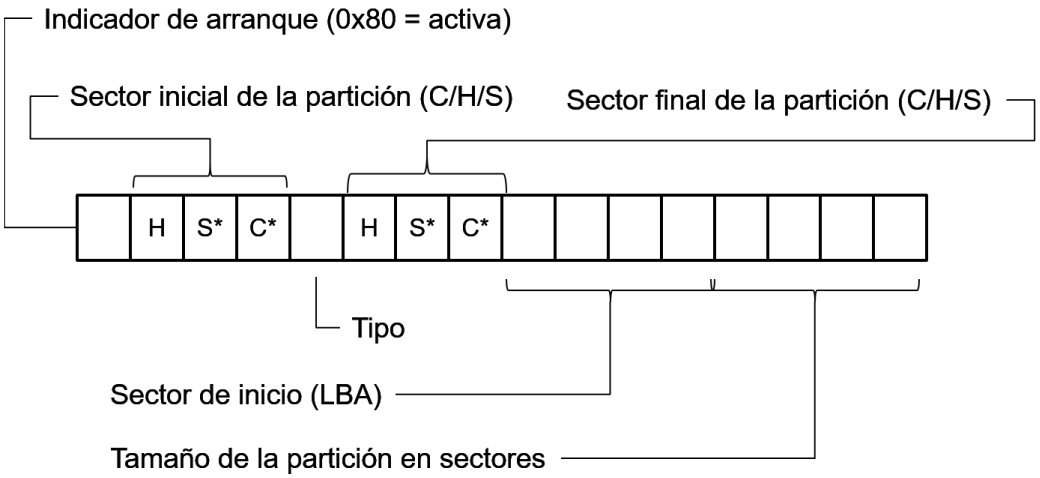
Esquema de particionado MBR

En sistemas con BIOS, la definición de las particiones de disco duro se encuentra en el primer sector del disco (de 512 bytes), a un desplazamiento de 446 bytes. En esta estructura de datos se pueden definir hasta cuatro particiones primarias, o tres primarias y una extendida, dentro de la cual se puede tener particiones lógicas.

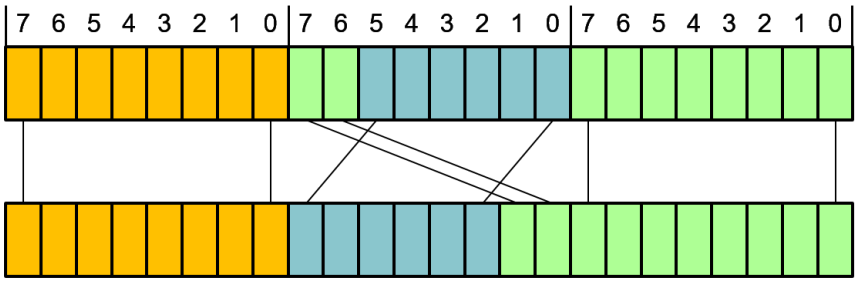


La tabla de particiones se encuentra dentro del MBR a un desplazamiento de 446 bytes, y ocupa 64 bytes. Cada una de las entradas ocupa 16 bytes.

Cada entrada de 16 bytes describe el tipo, el sector C/H/S y/o LBA de inicio y fin de cada sector, como se describe en la siguiente gráfica.



Por limitaciones en la cantidad de bytes que se puede usar, dos bits del campo “S”(sector) se toman para el campo “C” (Cilindro) como se muestra en la siguiente figura.

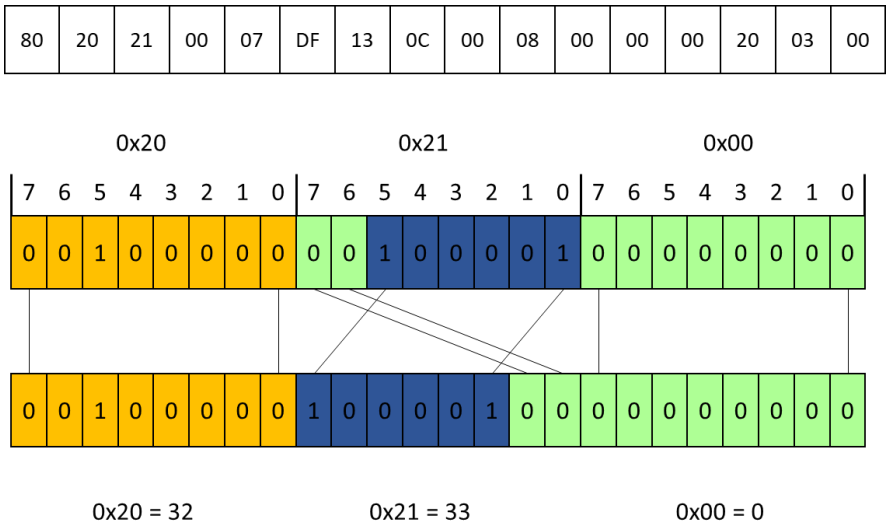


H (8 bits) S (6 bits) C (10 bits)

0-255 1-63 0-1023

Primer sector C/H/S: 0/0/1
Último sector C/H/S: 1023/255/63

A manera de ejemplo, se muestra una gráfica en la cual se explica cómo se decodificará una entrada de la tabla de particiones.



C/H/S = 0/32/33

Debido a las limitaciones impuestas por la BIOS, en la especificación ATA-5 se definió que, si una partición comenzaba o terminaba más allá del cilindro 1024, se debía almacenar los valores FE FF FF en los bytes correspondientes a C/H/S, esto significaba que se debía usar LBA y los valores C/H/S se consideraban obsoletos.

El valor LBA correspondiente a CHS (c, h, s) se puede obtener mediante la siguiente fórmula:

LBA = (c x H x SPT) + (h x SPT) + (s - 1)

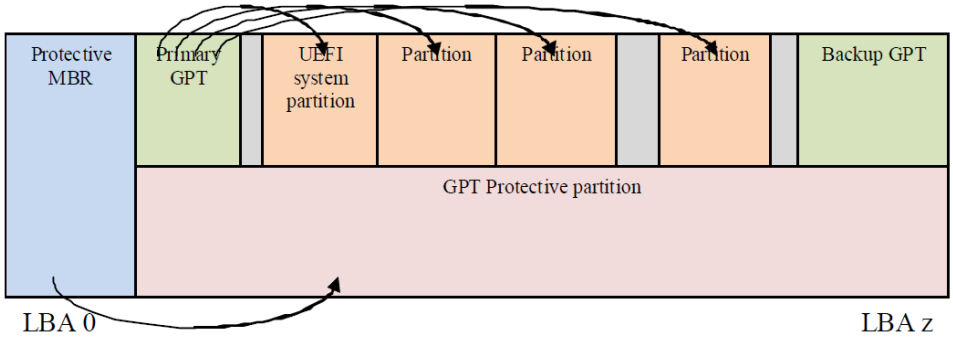
Para la mayoría de los discos actuales, H se toma como 16 y SPT (Sectors Per Track) como 63, por lo cual se puede usar la siguiente fórmula:

LBA = (c x 16 x 63) + (h x 63) + (s - 1)

Esquema de particionado GPT

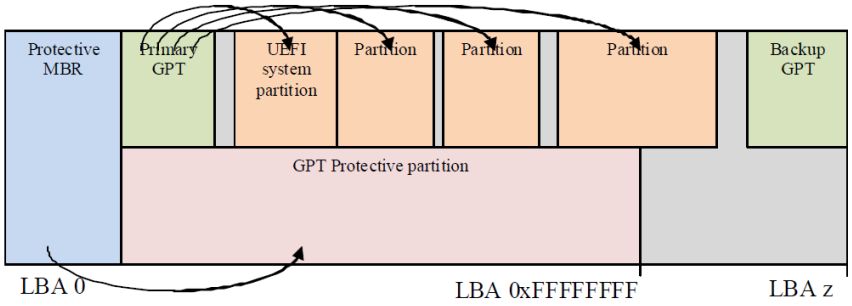
El esquema GPT forma parte de la Especificación UEFI, propuesta inicialmente por Intel para superar las limitaciones impuestas por el hardware y la BIOS en computadores personales. Para mantener compatibilidad

con sistemas anteriores (legacy), el primer sector del disco se denomina MBR de Protección (Protective MBR), tiene el formato MBR y sólo define una partición con tipo 0xEE (Protective GPT). Justo después de este sector, se encuentra la Tabla de Particiones GPT.



Tomado de UEFI Specification Version 2.10.

Cuando un disco tiene una cantidad de sectores mayor a 0xFFFFFFFF (el mayor número que se puede almacenar en 32 bits, tiene el siguiente esquema de particionado:

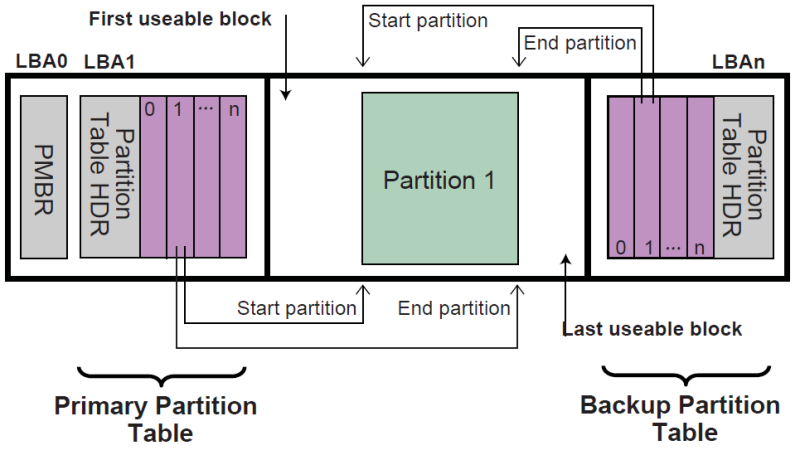


Tomado de UEFI Specification Version 2.10

Como se puede observar, la partición definida en la tabla MBR contiene el límite máximo en el campo LBA (0xFFFFFFFF).

En ambos casos, existe una copia de la GPT al final del disco, en una ubicación definida dentro de la GPT primaria.

La tabla de particiones GPT tiene el siguiente formato:



Tomado de UEFI Specification 2.10.

Después del primer sector del disco (el MBR de protección en el sector LBA 0) se encuentra la tabla de particiones. Esta consiste en un encabezado (Partition Table HDR – PTHDR en el sector LBA 1) que siempre ocupa como máximo un bloque (sector) en el dispositivo, seguido de **n** descriptores de partición. Cada uno de estos descriptores contienen la información de las particiones en el disco, que se encuentran a partir del primer sector usable (**First useable block**).

Como se define en la especificación UEFI, si el tamaño del bloque (sector) es de 512 bytes (por defecto), el primer bloque LBA usable debe ser mayor o igual a 34: 1 bloque para el MBR de protección, 1 bloque para el Encabezado de la Tabla de Particiones (PTHDR) y 32 bloques de 512 bytes de para el arreglo de descriptores de partición. ***Para este proyecto se puede asumir que el tamaño de bloque lógico es de 512 bytes, que corresponde al valor de los discos actuales***.

Para discos con tamaño de bloque de 4096 bytes (4 KB), el primer bloque LBA usable debe ser mayor o igual que 6: 1 bloque de 4KB para el MBR de protección (del cual solo se usan los primeros 512 bytes), 1 bloque para el Encabezado de la Tabla de Particiones (PTHDR) y 4 bloques para el arreglo de descriptores de partición.

Cada descriptor de partición ocupa exactamente 128 bytes, y contiene la siguiente información:

- GUID del Tipo de Partición (Identificador de la partición) 16 bytes: Define el tipo de la partición. La especificación UEFI define un GUID para cada tipo de partición, por ejemplo:
    - 00000000-0000-0000-0000-000000000000: Descriptor sin usar

    - C12A7328-F81F-11D2-BA4B-00A0C93EC93B: Partición del sistema UEFI (donde se almacena el cargador de arranque)
- GUID único de partición (16 bytes): GUID generado dinámicamente cuando se crea la partición. Cada partición debe tener un GUID diferente a las demás particiones definidas.
- Inicio LBA de la partición (8 bytes): Bloque de inicio de la partición
- Fin LBA de la partición (8 bytes): Bloque en el cual finaliza la partición
- Atributos (8 bytes): Atributos de la partición. Cada bit establecido en 1 define un atributo.
- Nombre de la partición (72 bytes): Cadena de caracteres terminada en NULL con el nombre dado a la partición.
- Reservado (SizeOfPartitionEntry - 128): Espaciador para que todos los descriptores tengan el mismo tamaño.

El valor de SizeOfPartitionEntry se encuentra en el Encabezado de la Tabla de Particiones (PTHDR) y define el tamaño de todos los descriptores. ***Para este proyecto, este campo no se debe incluir dentro del descriptor, ya que por defecto el tamaño de una entrada es 128 bytes y este campo está reservado para uso futuro***.

La documentación completa de la Tabla GPT puede ser consultado en el siguiente link:

https://uefi.org/specs/UEFI/2.10/05_GUID_Partition_Table_Format.html#id5

Referencias

Esquema de particiones MBR: https://thestarman.pcministry.com/asm/mbr/PartTables.htm

Esquema de particiones GPT: https://uefi.org/specs/UEFI/2.10/05_GUID_Partition_Table_Format.html#id5

Listado de tipos de particiones GPT: https://en.wikipedia.org/wiki/GUID_Partition_Table#cite_note-tn2166-12