O HPS espera um SDCARD formatado com o esquema a seguir para que o possa carregar o boot loader e depois o kernel :

Esse roteiro não trata a fundo a criação das partições no SDCARD, que podem ser ~facilmente~ geradas com o fsdisk do Linux. Aqui iremos usar uma imagem padrão que já possui a formatação e atualizar o SDCARD com o nosso uboot / kernel e filesystem.

Imagem padrão

Utilizaremos uma imagem (.iso) já gerado com as especificações e que já possui todo o sistema necessário para executar o linux no HPS (incluindo boot loader, kernel e filesystem):

O conteúdo da imagem e como a mesma foi criada é detalhado no página (acesse para fazer o download da iso):

- SDcard img base.

Para usar, basta fazer o download e salvar no SDCard. Note que no comando DD deve-se substituir o `SeuDevice` pelo caminho que o seu Linux atribuiu ao dispositivo. Para saber basta verificar o dmesg:

```
$ dmesg | tail
 4789.207972] mmc0: new ultra high speed SDR50 SDHC card at
address aaaa
[ 4789.211680] mmcblk0: mmc0:aaaa SL16G 14.8 GiB
[ 4789.215857]  mmcblk0: p1 p2 p3
[ 4988.443942]  mmcblk0: p1 p2 p3
```

- Estamos interessados no: `mmcblk0` .

Agora vamos salvar a .iso no SDcard.

⚠️ Cuidado, se errar o dispositivo (of=/dev/mmcblk0) pode acontecer muitas coisas ruins

```
$ sudo dd bs=4M if=SDCardEmptyCycloneV of=/dev/mmcblk0 conv=fsync
status=progress
$ sync
```

O sync é necessário para que o kernel faça um flush do cashe escrevendo realmente no SDCard todos os dados que foram endereçados a ele. Essa etapa pode ser um pouco demorada.

Agora basta montar no seu linux o SDCard recém escrito e devemos ter duas partições visíveis:

- 819,2 MiB: FAT32
  - Script de configuração do uboot; Kernel comprimido; Device Tree Blob file
  - u-boot.scr; zImage; socfpga.dtb
- 1,0 GiB:
  - Filesystem (/)

E outra partição que não é visível (contém o preloader e o uboot), para visualizar :

```
$ sudo fdisk -l /dev/mmcblk0
...
Device         Boot   Start     End Sectors   Size Id Type
/dev/mmcblk0p1      2121728 3799448 1677721 819,2M  b W95 FAT32
/dev/mmcblk0p2        14336 2111488 2097153     1G 83 Linux
/dev/mmcblk0p3         2048    4096    2049     1M a2 unknown
...
```

Note que a partição 3 (mmcblk0p3) é do tipo *unknown* (a2) e possui 1M de espaço. É nela que temos salvo o **preloader** e o **uboot**.

Próximos passo

⚠️ Se é a primeira vez nesse tutorial siga direto para o Tutorial HPS BlinkLED ⚠️

.

Atualizando os arquivos

Para atualizar o SDCARD com a versão recém gerada siga os passo a seguir.

Kernel

Para atualizar o kernel basta montar a partição 1 (819,2 M). No meu caso o path é : */run/media/corsi/B0DA-B234/* e copiar o arquivo zImage para ela.

> Você irá ter que editar para o caminho que a partição foi montada no seu linux.

```
$ cp ~/work/HPS-Linux/zImage /run/media/corsi/B0DA-B234/
$ sync
```

Quando fizer isso, irá reparar que a versão do kernel do Linux é a que foi gerada na etapa de compilação do kernel.

FileSystem

Para insierirmos nosso fileSystem no SDCARD, primeiramente deve montar a partição. no meu caso : */run/media/corsi/9cb79fd9-69b8-43e3-bcfe-fa4582579e2c/*. Uma vez montada, devemos primeiramente excluir os arquivos ali salvo (apagar o fileSystem antigo) e então extrair o que foi gerado pelo buildrrot.

```
# Limpando fs antigo
$ sudo rm -r /run/media/corsi/9cb79fd9-69b8-43e3-bcfe-fa4582579e2c/
$ sync
# Extraindo novo fs
$ sudo tar xvf rootfs.tar -C /run/media/corsi/9cb79fd9-69b8-43e3-
bcfe-fa4582579e2c/
$ sync
```

uboot script e device tree

```
$ cp ~/work/HPS-Linux/u-boot.scr /run/media/corsi/B0DA-B234/
$ cp ~/work/HPS-Linux/socfpga.dtb /run/media/corsi/B0DA-B234/
$ sync
```