FreeBSD en Solid State Apparaten

John Kozubik

john@kozubik.com

Dit artikel behandelt het gebruik van solid state disk-apparaten in FreeBSD voor het maken van embedded systemen.

Embedded systemen hebben het voordeel van verhoogde stabiliteit wegens het ontbreken van bewegende delen (harde schijven). Er moet echter rekening worden gehouden met de over het algemeen weinig beschikbare schijfruimte in het systeem en de duurzaamheid van het opslagmedium.

Specifieke onderwerpen die aan bod komen omvatten de typen en attributen van solid state-media die geschikt zijn om in FreeBSD als schijf te gebruiken, kernelopties die interessant zijn in zo'n omgeving, de mechanismen van `rc.diskless` die de initialisatie van zulke systemen automatiseren en de noodzaak voor alleen-lezen bestandssystemen, en het van voor af aan bouwen van bestandssystemen. Het artikel zal afsluiten met wat algemene strategiën voor kleine en alleen-lezen FreeBSD-omgevingen.

Vertaald door René Ladan.

Inhoudsopgave

1. Solid State Disk-apparaten

Het bereik van dit artikel zal beperkt zijn tot solid state disk-apparaten die gemaakt zijn met flash-geheugen. Flash-geheugen is een solid state-geheugen (geen bewegende onderdelen) dat niet-vluchtig is (het geheugen blijft gegevens behouden zelf nadat alle stroombronnen zijn ontkoppeld). Flash-geheugen kan enorme fysieke schokken weerstaan en is redelijk snel (de oplossingen met flash-geheugens die in dit artikel worden behandeld zijn iets langzamer dan een EIDE-harde schijf voor schrijfbewerkingen, en veel sneller voor leesbewerkingen). Een heel belangrijk aspect van flash-geheugen, waarvan de ramnificaties later in dit artikel besproken zullen worden, is dat elke sector een beperkte herschrijfcapaciteit heeft. Een sector flash-geheugen kan maar een bepaald aantal keren beschreven, gewist, en herschreven worden voordat de sector permanent onbruikbaar wordt. Hoewel veel flash-geheugenproducten automatisch slechte blokken in kaart brengen, en hoewel sommigen zelfs schrijfoperaties gelijkmatig over de eenheid distribueren, blijft het een feit dat er een limiet bestaat aan de hoeveelheid waarmee het apparaat kan worden beschreven. Concurrerende apparaten hebben tussen de 1.000.000 en 10.000.000 schrijfbewerkingen per sector in hun specificaties staan. Dit getal varieert vanwege de omgevingstemperatuur.

In het bijzonder worden ATA-compatibele compact-flash-eenheden besproken, welke vrij populair zijn als opslagmedium voor digitale camera's. Bijzonder interessant is het feit dat de pinnen ervan precies met die van de IDE-bus overeenkomen en dat ze compatibel zijn met de ATA-commandoverzameling. Daarom kunnen deze apparaten direct aan een IDE-bus in een computer gekoppeld worden met een zeer eenvoudige en goedkope adapter. Eenmaal op deze wijze geïmplementeerd zien besturingssystemen zoals FreeBSD het apparaat als een normale harde schijf (doch klein).

Er bestaan nog andere solid state disk-oplossingen, maar hun kosten, zeldzaamheid, en relatieve gebruiksongemak plaatst ze buiten het bereik van dit artikel.

2. Kernelopties

Enkele kernelopties zijn specifiek interessant voor degenen die een embedded FreeBSD-systeem creëren.

Ten eerste zullen alle embedded FreeBSD-systemen die flash-geheugen als systeemschijf gebruiken geïntereseerd zijn in geheugenschijven en geheugenbestandssystemen. Vanwege het beperkt aantal keren dat het flash-geheugen kan worden beschreven, is het het waarschijnlijkst dat de schijf en de bestandssystemen op de schijf als alleen-lezen worden aangekoppeld. In deze omgeving zullen bestandssystemen zoals `/tmp` en `/var` als geheugenbestandssystemen worden aangekoppeld zodat het systeem logs kan creëren en tellers en tijdelijke bestanden kan bijwerken. Geheugenbestandssystemen zijn een kritiek station naar een succesvolle implementatie van solid state FreeBSD.

De volgende regels dienen in uw kernelinstellingenbestand te staan:

```
options         MFS             # Geheugenbestandssysteem
options         MD_ROOT         # md-apparaat bruikbaar als een potentieel root-apparaat
pseudo-device   md              # geheugenschijf
```

3. `rc.diskless` en alleen-lezen bestandssystemen

De post-boot-initialisatie van een embedded FreeBSD-systeem wordt beheerd door `/etc/rc.diskless2` (`/etc/rc.diskless1` is voor BOOTP-schijfloos opstarten). Dit initialisatiescript wordt aangeroepen door de volgende regel in `/etc/rc.conf` te plaatsen:

```
diskless_mount=/etc/rc.diskless2
```

`rc.diskless2` koppelt `/var` als een geheugenbestandssysteem aan, maakt een instelbare lijst van mappen in `/var` aan met het commando mkdir(1), verandert de modus van sommige van deze mappen, en pakt een lijst van apparaatingangen uit naar een schrijfbare (weer als een geheugenbestandssysteem) partitie `/dev`. Tijdens het uitvoeren van `/etc/rc.diskless2` is er nog een `rc.conf`-variabele in het spel - `varsize`. Het bestand `/etc/rc.diskless2` maakt een partitie `/var` aan gebaseerd op de waarde van deze variabele in `rc.conf`:

```
varsize=8192
```

Onthoud dat deze waarde in sectoren is. De creatie van de partitie `/dev` door `/etc/rc.diskless2`, wordt echter geregeerd door een harde waarde van 4096 sectoren. Het is triviaal om deze waarde in het bestand `/etc/rc.diskless2` zelf te wijzigen, alhoewel er niet meer ruimte voor `/dev` dan dat nodig zou zijn.

Het is belangrijk om te herinneren dat het script `/etc/rc.diskless2` aanneemt dat de conventionele partitie `/tmp` reeds door een symbolische koppeling naar `/var/tmp` is vervangen. Omdat `tmp` een van de mappen is die in `/var` door het script `/etc/rc.diskless2` wordt aangemaakt, en omdat `/var` een geheugenbestandssysteem is (dat als lezen-schrijven is aangekoppeld), zal `/tmp` nu ook een lees-schrijf map zijn.

Het feit dat `/var` en `/dev` lees-schrijf bestandssystemen zijn is een belangrijk verschil, aangezien de partitie `/` (en alle andere partities die op uw flash-medium kunnen staan) als alleen-lezen aangekoppeld dienen te worden. In Paragraaf 1 hebben we de beperkingen van flash-geheugen uiteen gelegd - in bijzonder de beperkte herschrijfcapaciteit. Het belang van het niet als lezen-schrijven aankoppelen van flash-media en het belang van het niet gebruiken van een wisselbestand kunnen niet genoeg benadrukt worden. Een wisselbestand op een druk systeem kan binnen een jaar een flash-medium opmaken. Het uitgebreid loggen of aanmaken en vernietigen van tijdelijke bestanden kan hetzelfde doen. Daarom dient u, naast het verwijderen van de regels `swap` en `/proc` uit het bestand `/etc/fstab`, dient u ook de Options van elk bestandssysteem als volgt op `ro` te zetten:

```
# Device                Mountpoint      FStype  Options         Dump    Pass#
/dev/ad0s1a             /               ufs     ro              1       1
```

Op een gemiddeld systeem zullen enkele applicaties het onmiddellijk niet meer doen als gevolg van deze verandering. Ports bijvoorbeeld zullen niet installeren vanuit de portsboom omdat `/var/db/port.mkversion` niet bestaat. cron zal niet correct draaien vanwege ontbrekende crontabellen in het `/var` dat door `/etc/rc.diskless2` is aangemaakt, en syslog en DHCP zullen problemen ondervinden als gevolg van het alleen-lezen bestandssysteem en ontbrekende items in het `/var` dat `/etc/rc.diskless2` heeft aangemaakt. Dit zijn slechts tijdelijke problemen, en worden tezamen met oplossingen voor het uitvoeren van andere veelgebruikte softwarepakketten behandeld in Paragraaf 5.

Een belangrijk ding om te onthouden is dat een bestandssysteem dat met `/etc/fstab` als alleen-lezen was aangekoppeld ten alle tijden lezen-schrijven kan worden gemaakt door dit commando te geven:

```
# /sbin/mount -uw partitie
```

en kan op alleen-lezen worden teruggezet met het commando:

```
# /sbin/mount -ur partitie
```

4. Een bestandssysteem uit het niets opbouwen

Omdat ATA-compatibele compact-flash-kaarten door FreeBSD als normale IDE harde schijven worden gezien, kunt u theoretisch FreeBSD vanaf het netwerk installeren door de floppies kern en mfsroot of een CD te gebruiken.

Zelfs een kleine installatie van FreeBSD die normale installatieprocedures gebruikt kan echter een systeem produceren met een omvang van meer dan 200 MB. Omdat de meeste mensen kleinere flash-geheugenapparaten zullen gebruiken (128 MB wordt als redelijk groot gezien - 32 of zelfs 16 MB is gebruikelijk) is een installatie dat de normale mechanismen gebruikt niet mogelijk er is simpelweg niet genoeg schijfruimte voor zelfs de kleinste van de conventionele installaties.

De gemakkelijkste manier om over deze beperking heen te komen is om FreeBSD met conventionele middelen naar een normale harde schijf te installeren. Kleedt, nadat de installatie voltooid is, het besturingssysteem uit tot een grootte die op uw flash-medium past, en pak vervolgens het gehele bestandssysteem in. De volgende stappen leiden u door het proces heen van een normaal flash-geheugen voorbereiden op uw ingepakte bestandssysteem. Omdat er geen normale installatie wordt uitgevoerd, moeten bewerkingen zoals partitioneren, labelen, het aanmaken van bestandssystemen, etc. met de hand uitgevoerd worden. Naast de floppies kern en mfsroot heeft u ook de floppy fixit nodig.

1. Het flash-media-apparaat partitioneren

   Kies nadat er met de floppies kern en mfsroot is opgestart `custom` uit het installatiemenu. Kies `partition` in het aangepaste installatiemenu. In het partitiemenu dient u alle bestaande partities te wissen met de toets **d**. Maak nadat alle bestaande partities gewist zijn een partitie aan met de toets **c** en accepteer de standaardwaarde voor de grootte van de partitie. Zorg ervoor dat het type van de partitie op `165` is ingesteld wanneer daar naar wordt gevraagd. Schrijf nu deze partitietabel naar schijf door op de toets **w** te drukken (dit is een verborgen optie op dit scherm). Wanneer u een ATA-compatibele flash-kaart gebruikt, dient u de opstartbeheerder van FreeBSD te gebruiken. Druk nu op de toets **q** om het partitiemenu te verlaten. Het menu van de opstartbeheerder wordt nog een keer aan u getoond - herhaal de keuze die u eerder heeft gemaakt.
2. De bestandssystemen op uw flash-geheugenapparaat aanmaken

Verlaat het aangepaste installatiemenu, en kies van het hoofdinstallatiemenu de optie `fixit`. Geef na het binnengaan van de fixit-omgeving het volgende commando:

```
# disklabel -e /dev/ad0c
```

Op dit punt bent u de tekstverwerker vi binnengegaan onder toezien van het commando disklabel. Vervolgens dient u een regel met `a:` aan het einde van het bestand toe te voegen. Deze regel dient er als volgt uit te zien:

```
a:       123456  0       4.2BSD  0       0
```

Hierbij is *123456* een getal dat exact gelijk is aan het getal in de bestaande regel met `c:` voor de grootte. In feite dupliceert u de bestaande regel met `c:` als een regel met `a:`, met daarbij `4.2BSD` als type van het bestandssysteem. Sla het bestand op en verlaat de tekstverwerker.

```
# disklabel -B -r /dev/ad0c
# newfs /dev/ad0a
```

3. Uw bestandssysteem op het flash-medium plaatsen

   Koppel het nieuw voorbereide flash-medium aan:

   ```
   # mount /dev/ad0a /flash
   ```

   Activeer deze machine in het netwerk zodat we ons tar-bestand kunnen overzenden en het op het bestandssysteem van het flash-medium kunnen uitpakken. Een manier om dit te doen is:

   ```
   # ifconfig xl0 192.168.0.10 netmask 255.255.255.0
   # route add default 192.168.0.1
   ```

   Nu de machine op het netwerk is, kan het tar-bestand worden overgezonden. U kunt nu tegen een dilemma aanlopen - als bijvoorbeeld uw flash-geheugen 128 MB groot is, en uw tar-bestand groter is dan 64 MB, kan uw tar-bestand niet op het zelfde moment op het flash-medium staan als dan wanneer u het uitpakt - u zult schijfruimte tekort komen. Een oplossing voor dit probleem is, wanneer u FTP gebruikt, om het bestand uitpakt terwijl u het over FTP verzendt. Als u de overdracht op deze manier aanpakt, zult u nooit het tar-bestand en de inhoud ervan op hetzelfde moment op uw schijf hebben:

   ```
   ftp> get tar-bestand.tar "| tar xvf -"
   ```

   Als uw tar-bestand met gzip is ingepakt, kunt u dit ook voor elkaar krijgen:

   ```
   ftp> get tar-bestand.tar "| zcat | tar xvf -"
   ```

   Nadat de inhoud van uw ge-tar-de bestandssysteem op het bestandssysteem van uw flash-geheugen staan, kunt u het flash-geheugen afkoppelen en opnieuw opstarten:

   ```
   # cd /
   # umount /flash
   # exit
   ```

   Aangenomen dat u uw bestandssysteem correct heeft geconfigureerd toen het gebouwd werd op de normale harde schijf (met uw bestandssystemen als alleen-lezen aangekoppeld en met de nodige opties in de kernel gecompileerd) zou u nu succesvol uw embedded FreeBSD-systeem moeten kunnen opstarten.

5. Systeemstragiën voor kleine en alleen-lezen omgevingen.

In Paragraaf 3 werd erop gewezen dat het bestandssysteem `/var` zoals geconstrueerd door `/etc/rc.diskless2` en de aanwezigheid van een hoofdbestandssysteem dat alleen gelezen kan worden problemen veroorzaakt met veel

alledaagse softwarepakketten die door FreeBSD gebruikt worden. In dit artikel zullen suggesties voor het succesvol draaien van cron, syslog, ports-installaties en de webserver Apache worden gegeven.

5.1. cron

In `/etc/rc.diskless2` staat een variabele genaamd `var_dirs`. Deze variabele bestaat uit een met spaties afgebakende lijst van mappen die binnen `/var` aangemaakt zullen worden nadat het als een geheugenbestandssysteem is aangekoppeld. `cron` en `cron/tabs` staan niet in deze lijst, en zonder deze mappen zal cron klagen. Door `cron`, `cron/tabs`, en misschien zelfs `at` en `at/jobs` als elementen van deze variabele toe te voegen, wordt het makkelijker om de daemons cron(8) en at(1) te draaien.

Dit lost echter nog steeds niet het probleem van het behouden van cron-tabellen na het opnieuw opstarten op. Wanneer het systeem opnieuw opstart, zal het bestandssysteem `/var` dat in het geheugen staat verdwijnen en zullen alle cron-tabellen die er in stonden ook verdwijnen. Daarom is een oplossing hiervoor het aanmaken van cron-tabellen voor de gebruikers die ze nodig hebben, uw bestandssysteem `/` als lezen-schrijven aan te koppelen en die cron-tabellen naar een veilige plaats zoals `/etc/tabs` te kopiëren en een regel aan het einde van `/etc/rc.diskless2` toe te voegen die deze cron-tabellen naar `/var/cron/tabs` kopieert nadat die map is aangemaakt tijdens de syseeminitialisatie. U moet misschien ook een regel toevoegen die de modi en toestemmingen van de mappen die u aanmaakt en de bestanden die u met `etc/rc.diskless2` kopieert verandert.

5.2. syslog

`syslog.conf` specificeert de plaats van bepaalde logbestanden die in `/var/log` bestaan. Deze bestanden worden niet door `/etc/rc.diskless2` tijdens de systeeminitialisatie aangemaakt. Daarom dient u ergens na de sectie die de mappen in `/var` aanmaakt in `/etc/rc.diskless2` iets als het volgende toevoegen:

```
# touch /var/log/security /var/log/maillog /var/log/cron /var/log/messages
# chmod 0644 /var/log/*
```

U moet ook de logmap toevoegen aan de lijst van mappen die `/etc/rc.diskless2` aanmaakt.

5.3. ports-installatie

Voordat de veranderingen die nodig zijn om succesvol de portsboom te gebruiken besproken worden, is een herinnering ten aanzien van de alleen-lezen-natuur van uw bestandssystemen op het flash-medium op zijn plaats. Aangezien ze alleen-lezen zijn, dient u ze tijdelijk als lezen-schrijven aan te koppelen waarbij de koppelsyntaxis zoals getoond in Paragraaf 3 wordt gebruikt. U dient deze bestandssystemen altijd als alleen-lezen te herkoppelen als u klaar bent met enig onderhoud - onnodige schrijfacties naar het flash-medium kunnen de levensduur ervan aanzienlijk verkorten.

Om het mogelijk te maken om een portsmap binnen te gaan en succesvol `make install` uit te voeren, moeten we een pakketmap op een bestandssysteem aanmaken dat niet geheugengebaseerd is en dat onze pakketten tussen herstarts bijhoudt. Omdat het toch nodig is om uw bestandssystemen als lezen-schrijven te koppelen voor het installeren van een pakket, is het zinnig om aan te nemen dat een gebied op het flash-medium ook gebruikt kan worden om pakketinformatie naar te schrijven.

Maak als eerste een map aan voor de pakketdatabase. Dit is normaliter `/var/db/pkg`, maar we kunnen het daar niet plaatsen aangezien het telkens als het systeem wordt opgestart zal verdwijnen.

```
# mkdir /etc/pkg
```

Voeg nu een regel aan `/etc/rc.diskless2` toe die de map `/etc/pkg` aan `/var/db/pkg` koppelt. Een voorbeeld:

```
# ln -s /etc/pkg /var/db/pkg
```

Nu zal telkens wanneer u uw bestandssystemen als lezen-schrijven aankoppelt en een pakket installeert, `make install` werken, en zal de pakketinformatie succesvol naar `/etc/pkg` worden geschreven (omdat het bestandssysteem op dat moment als lezen-schrijven is aangekoppeld) wat altijd als `/var/db/pkg` beschikbaar is voor het besturingssysteem.

5.4. Apache Web Server

Apache houdt pid-bestanden en logs in *`apache_install`*`/logs`. Aangezien deze map ongetwijfeld op een bestandssysteem staat dat alleen gelezen kan worden, zal dit niet werken. Het is nodig om een nieuwe map aan de lijst van mappen die in `/var` aangemaakt moeten worden toe te voegen in `/etc/rc.diskless2` en om *`apache_install`*`/logs` aan `/var/log/apache` te koppelen. Het is ook nodig om de toestemmingen en eigenaarschappen van deze nieuwe map in te stellen.

Voeg eerst de map `log/apache` toe aan de lijst van mappen die in `/etc/rc.diskless2` aangemaakt moeten worden.

Voeg ten tweede deze commando's toe aan `/etc/rc.diskless2` na de sectie die mappen aanmaakt:

```
# chmod 0774 /var/log/apache
# chown nobody:nobody /var/log/apache
```

Verwijder als laatste de bestaande map *`apache_install`*`/logs` en vervang het door een koppeling:

```
# rm -rf (apache_installatie)/logs
# ln -s /var/log/apache (apache_installatie)/logs
```