|  |  |
| --- | --- |
| 5 랩5 lba가 기존에 사용하던 lba와 동일한가요? ppt를 보면 zone id와 fcg id를 lba로 나타내는 것 같은데 input1.txt를 예로 들면 0번 lba는 zone id=0, fcg id=0, 그리고 lba가 3일 경우 zone id=1, fcg=1인건가요?  위의 말은 살짝 말이 안되는거 같은데 lba가 0이라면 0번 존일 테니까 L2P를  0번 존에 해당하는 값이 zone id+fcg id로 바꿀 수 있게 테이블을 만드는 건가요?  그리고 max open zone의 경우 fcg마다 존재하는게 맞나요?  zone 할당의 경우 큐 방식으로 한다고 말씀해주셨는데 fcg가 여러개 일 경우 어떤  순서로 해야 하나요? | 기본적으로는 host에 제공되는 같은 LBA 인터페이스인데 zone에 따른 제약이 생긴 점이 다르고 내부적으로 LBA와 NAND간의 맵핑은 완전히 다릅니다.  input1.txt의 경우 zone 하나의 크기가 64섹터이며 하나의 zone은 하나의 FCG에 맵핑됩니다. 따라서 lba 0부터 63까지는 같은 zone이므로 FCG도 같습니다. LBA 0-7은 FBG의 첫 bank의 첫 page에 적히면 됩니다.  max open zone은 FCG마다가 아니라 전체로 관리 해주시면 됩니다.  fifo 방식 관리는 FCG마다 관리되는 부분이며 초기화 할때 각 FCG에서 큐에 input1 예제 기준으로 8개의 블록을 가지므로 FBG를 0부터 7까지 enqueue 하고 차례대로 dequeue 해서 사용하면 됩니다. zone reset 등으로 할당되었던 FBG이 erase 될 때 큐에 enqueue되면 되구요 |
| 6 랩5에서 lba는 그럼 섹터의 주소를 나타내는게 아닌건가요?  input1.txt를 예로 말씀드리면 dzone이 2라서 fcg가 2개가 되는데 처음 0 lba를  받게 되면 낸드 상에서 새로운 존을 할당해줘야 할 것 같습니다. 근데 이 때 fcg0에서  할당할지 fcg1에서 할당할지 어떻게 정하는 건가요? | 섹터의 주소를 나타내는게 맞으며 NAND 맵핑만 기존 페이지맵핑 FTL과 다르다고 생각하시면 됩니다. zone 번호 (상위 부분)가 결정되면 나머지 부분인 page, block, bank는 static하게 결정됩니다.  FCG는 ZNS+ 논문의 방식대로 맵핑을 합니다. Lab5슬라이드의 striping 부분을 보시면  zone 번호의 하위 비트를 보고 판단할 수 있습니다.  해당 예제는 fcg가 두개이므로 짝수 zone은 fcg 0, 홀수 zone은 fcg 1에 가게 됩니다. |
| 7 **그럼 fcg가 4개일 때 0번 존(lba가 0인 존)은 fcg 0에서 골라야 하고 1번 존은 fcg 1, 2번 존은 fcg 2, 3번 존은 fcg 3, 4번 존은 fcg0이 맞나요?**  6번 답변에서 홀수 zone이 fcg 1인거죠? | **네 맞습니다**  아 반대로 쓴듯 하네요.. **general 하게 하면 zone 번호를 FCG 수로 나눈 나머지가 FCG 번호입니다.** |
| 8 zns\_tl\_open의 경우 기존 zone과 새로 적히는 zone에 대한 매핑이 2개가 될텐데 이럴 경우 l2p 말고도 다른 정보 저장소가 있어야 하나요? | 네 L2P (zone to fbg)를 임시로 하나 더 관리하셔야 합니다. |
| 9 zns\_tl\_open의 경우에서 valid arr의 수는 무작위인가요? | 고정 크기입니다. zone size (sector단위) 입니다. |
| 10 마지막으로 tl\_open은 처음 tl\_open이었다가 full로 바뀌고 source zone은 full에서 empty로 가는게 맞나요? | 네 (valid가 아닌)마지막 섹터가 쓰일 때 source zone은 갖고있던 FBG를 반납하고 empty로 dest zone은 full 상태가 됩니다. |
| 11 버퍼의 경우 오픈 zone 개수만큼만 할당해야 하나요? 아니면 모든 zone만큼 할당해도 되나요?  open zone만큼 리소스를 할당한다면 버퍼에서 하나의 엔트리를 더 만들어서 몇번째 존에 대한 버퍼인지도 저장해줘야 하는 것이 맞나요? | 실제 SSD라면 open zone 수만큼만 리소스를 사용하겠지만 본 과제에서는 어떻게 구현하셔도 무방할 듯 합니다.  네 각 open zone에 할당된 버퍼가 어떤 것인지 맵핑 정보의 관리가 필요할듯합니다. |
| nand\_write에서 spare 공간은 아무것도 안적어줘도 되나요? | 네 아무값이나 채우셔도 됩니다. LBA를 적어두면 디버깅하기 편하실 수도 있어요 |
| 섹터(8개)만큼 다 차지 않아서 버퍼에 임시적으로 쓸 경우에도 wp를 올려주어야 한다고 생각하는데 버퍼 자체에서도 오버라이트가 안되는 것이 맞나요? | 네 현재 NVMe ZNS 스펙상 host에서 한번 write한 섹터들의 overwrite를 허용하지 않으며 본 과제에서도 마찬가지로 버퍼에 쓰더라도 overwrite는 불가능합니다. |
| 과제 ppt 33페이지에 있는대로 zns\_tl\_open을 할 때 write(48,3) 이렇게 들어오면 -1을 리턴해야 하나요? 그리고 혹시나 write(49,5)처럼 할 경우에는 52번째 섹터에서 기존 낸드에 있던 것과 새로 들어온 데이터 충돌이 일어날 것 같은데 이러한 경우 어떻게 해주어야 하나요? | 네 write(48, 3)이나 write(49, 5)처럼 기존 데이터와 충돌나는 경우에는 -1리턴해주시면 됩니다. |
| ppt 32페이지에 각 뱅크마다 MAX\_OPEN\_ZONE을 갖는다고 했는데 5번 질문에서 전체로 관리해달라고 하셨습니다. 혹시 어떤 게 맞을까요?  tl\_open의 source zone 상태를 full로 출력해야 하나요 아니면 tlopen으로  출력해야 하나요?   tl open zone을 선택할 때 같은 fcg에 있는 zone을 선택하면 되나요? | 그만큼의 NAND fbg을 OP로 보유한다는 뜻이고 관리는 전체로 합니다.  tl\_open은 source dest와 logical하게 같은 zone입니다. tl\_open된 zone은 ZONE\_TLOPEN 상태로 출력하시면 됩니다  네 동일한 zone 넘버 유지하기 때문에 fcg도 같아야 합니다. |
| ppt 29페이지에서는 인자가 szone인데 코드에서는 lba로 되어있습니다.  어떤 것이 맞나요? | 아 ppt가 잘못되었습니다. lba로 해주세요 desc를 가져올 첫번째 zone의 lba입니다. 두 번째 인자는 헤더파일에 nsect라 되어있는데 c파일처럼 nzone이 맞고 zone의 수입니다. |
| zns\_izc에서 sector는 page를 말하는건가요, 아니면 page내부 sector를 말하는건가요? 만약 page 내부 sector라면, 일반 write처럼 하나의 page가 가득차지 않으면 buffer에 올려놨다가 나중에 nand\_write해줘야 하나요? | sector는 NAND page보다 작은 host에 제공하는 LBA 단위 입니다. 네 일반 write와 동일하게 page보다 작은 write의 경우 버퍼에 잠시 저장해둡니다. |
| reset함수에서 해당 lba가 맵핑된 zone을 empty상태로 만들기만 하는건지, 아니면  해당 lba와의 맵핑을 끊는 것 까지 해야하는지 헷갈립니다. | reset하면 해당 zone은 fbg를 반납하므로 맵핑이 끊어지게 됩니다. |
| zns\_izc에서 sector를 카피해오는데 그냥 해당 zone에서 몇 번째 섹터인지를 나타내는  건가요? 31페이지 예시를 보면 zone3에서 0 섹터를 가져오는데 zone 3의 lba는  0이 될 수 없다고 생각해서요 | zone 내에서 오프셋입니다 |
| output1.txt에서 29, 33번째 줄을 보면 256에 대해 한번은 실패하고 한번은 성공하는데 둘 다 실패해야 되지 않나요? | write(112, 16) 하면서 zone 1번이 full로 전환되면서 추가적인 open이 가능해집니다. |
| output2.txt에서 마지막에 lba가 0인 존에 대한 설명을 출력하는데 slba가 0 아닌가요? output2.txt에서는 64로 나와서 질문드립니다 | 0이 맞습니다 수정해서 올리겠습니다 |
| 예를 들어 한 zone이 64개 sector로 구성된다고 가정하겠습니다. tl\_open된 상태에서, 유효한 마지막 sector가 50번이고, 이 50번 sector가 새로 open한 zone에 복사가 완료된 경우, 해당 zone의 상태는 tl\_open인가요 open인가요? | tl\_open된 zone의 상태는 full 되기 전까지는 tl\_open 입니다. |
| 21번 질문에서 0이 맞다는 말씀이시죠? 그리고 22번 질문에서 zone state가 open인 것들만 max open zone에 유효한 것들인가요? zone이 full이 되면 사용 가능한 zone이 하나 늘어나는 건가요? | 네 0이 맞아요 ㅠㅠ TL open을 포함한 open zone의 수가 최대 MAX\_OPEN일 때까지만 open이 가능하고 full이 되면 open count가 줄어듭니다 |
| tl\_open할 때도 사용한 zone 수를 줄여야 하나요? | tl open도 opne과 마찬가지로 FULL되기 전까지 하나의 open 리소스 count를 차지합니다 |
| input1.txt에서 FBG 1개는 2개의 block으로 이루어져 있는게 맞나요? | 네 두 개의 bank를 묶어서 fbg로 사용하므로 블록도 두 개가 됩니다 |
| tl\_open에서 sector별로 copyback을 진행할 때에 즉시 nandwrite를 수행해야 하나요? 아니면 메모리상에 따로 버퍼를 두어서 정확한 page가 완성될 때마다 nand\_write하면 되나요? 전자의 경우는 bulk erase 특성상 불가능할 것 같고 후자의 경우는 어차피 메모리상에 tl에 대한 data를 가지고 있는 버퍼를 두기에 굳이 nand를 이용해야 한다는 비효율성이 생각됩니다. | nand 특성상 page를 한 번 쓰고 나면 수정이 불가능 하기 때문에 일반 write와 비슷하게 write 버퍼를 할당하고 copyback이 필요한 섹터들은 nand로 부터, 아닌 경우는 입력으로 들어온 data를 버퍼에 쓰고 nand 페이지마다 한번씩 nand\_write 해주시면 됩니다. |
| **ppt에서 26page에서 저 파란 점선 박스는 무엇을 의미하는 건가요? 저 예시에서는 dzone이 4라서 하나의 zone에 4개의 block이 있고 FCG별로 bank가 4개 있는 것 까지는 이해했습니다. 또다른 질문으로 FBG에 대한 개수도 해당 예시에서 알려주실 수 있으신가요? FBG개념이 잘 이해가 되지 못했습니다...** | **파란점선이 flash block group(FBG) 입니다.  logical한 zone 하나를 NAND에 맵핑하는 방식은 여러가지가 가능하겠지만 zone하나에 write할 때 NAND parallelism을 최대한 활용하려면 여러 bank의 블록들을 묶어서 zone에 할당해야하고 이를 fbg라고 합니다.  한 FCG내에서 FBG의 수는 블록의 수와 동일합니다. 6페이지 그림의 경우 dzone이 4이고 bank가 8개이므로 2개의 FCG가 나오고 하나의 FCG내에는 4개의 FBG가 나옵니다.** |
| 26번 추가질문입니다. 그러면 메모리상에 zone과 크기가 똑같은 size의 array를 두어서 copyback을 진행하고 유저가 sparse한 sector에 data를 write할 때마다 page가 완성되면 TL zone에 실제 nand\_wirte를 진행하는 것으로 이해하는 것이 맞나요? 근데 위와 같이 하면 tl\_open이 page단위가 아니라 sector단위로 이루어지기에 zone을 nand라는 저장소에 copyback하지 못하고 (모든 page당 sector가 일부만 copyback 되었다면 모든 page에 대해 write 수행이 불가) 전체 zone의 부분적인 sector data를 메모리의 array에 담게 되는 다소 이상한 상황인 것 같아서요.. 만약 copyback이 page bitmap단위로 이루어지면 실제 nand\_write는 page단위로 이루어지므로 위와 같이 전체 zone에 대한 copyback data를 메모리에 담을 필요없이 tl zone에 page마다 copy back을 한뒤에 일반적인 zone처럼 sector를 위한 page크기의 버퍼만 두어서 bitmap을 통해 관리할 수 있지 않을까 하는 생각이 들었습니다. | zone 전체를 메모리에 올려두어도 되겠지만 메모리 효율성을 생각하면 좋은 방법은 아닙니다 valid map만 저장해두면 1 페이지 크기의 write buffer로도 threaded logging이 가능합니다. (과제에서는 4B섹터 vs 1B로 valid array를 사용하여 차이가 크지 않지만 실제라면 512B 정도 섹터에 valid를 비트맵으로 1bit씩 저장하면 많은 차이가 납니다)  예를 들어 섹터 5-7이 valid 데이터로 copyback 되어야 하는 부분이면 write(0, 4)이 들어왔을 때, nand\_read로 첫 페이지를 읽어 버퍼에 쓰고 0-4섹터를 버퍼에 쓰면 기존의 데이터 5-7을 포함한 한 페이지가 완성되며 이 때 nand\_write를 하면 됩니다.  물론 시뮬레이터 과제인만큼 어떻게 구현하셔도 상관은 없습니다. |
| 28번 추가 질문입니다. 조교님이 답변해주신 NAND\_READ의 경우 저 READ가 TL\_OPEN의 COPYBACK의 target 존인것으로 이해하는데 target zone이 user로 인해서 reset 명령이 오는 경우 target zone과 tl zone 모두 empty로 해주면 되나요? 저는 target zone만 empty되서 target zone에서 더이상 read가 불가능해져서 메모리에 모든 zone의 data를 tl을 위해 담고있어야 된다고 생각했습니다.. | TL open의 경우 기존 zone과 새로운 zone 이 동일한 zone입니다. 내부적으로 새로운 fbg를 할당하고 기존 fbg에서 copyback 해오는 것이고 기존 fbg에 대한 erase는 tl open된 zone의 끝까지 쓰여서 full 상태가 될 때 erase해주면 됩니다 |
| 같은 zone에 대해서 tl open이 여러번 발생할 수 있나요? 예를 들어서 기존의 fbg에 쓴뒤, 새로운 fbg에 tlopen한 data를 다쓰고, full이 된 뒤에 reset없이 또 다시 같은 zone을 tlopen할 수 있나요? | 네 tl open으로 다 쓴 full zone이나 일반 open으로 쓴 full zone이나 동일한 상태이기 때문에 다시 tl open으로 여는 것도 가능합니다. |
| nbank보다 dzone이 더 클수도 있나요? 예를들어 nbank가 4이고 dzone이 8인 경우면 skeleton에서 NUMFCG가 0이되는데 이러한 상태에서의 ZNS test는 무의미한 것이죠? | dzone은 nbank 보다 작고 nbank의 약수로만 가능합니다. 이외의 경우는 없다고 생각하셔도 됩니다. |
| zone의 read나 write의 범주는 하나의 zone의 범위에만 들어가나요?  예를들어 zone\_size가 64sector일 경우 znc\_write(63,2)는 입력으로 들어오지 않나요? | 네 read, write의 lba 범위는 하나의 zone으로만 제한하겠습니다. 말씀하신 예시 같은 경우는 없습니다 |
| 새로올려주신 pdf에서 22페이지 zone state가 이해가 잘 되지 않습니다 Full 상태에서 TL Open을 하면 TLOpen 상태가 된다고 적혀있는데 TL Open을 하면 src 존은 Full->Full, dest 존은 Empty->TLOpen이 맞을까요? 33(+32)페이지에서 61번 섹터를 쓰고나면 dest 존은 TLOpen->Full, src존은 erase 전에는 계속 Full 상태를 유지하면 될까요? | tl open은 source dest가 따로 없습니다 하나의 zone에 대해 일어나는 것이고 해당 zone은 물리적인 NAND fbg만 두 개를 일시적으로 할당한 상태입니다. 33페이지 예시에서 61번 섹터를 쓰고나면 zone은 full 상태가 되는 것이고 src fbg는 nand\_erase 해주시면 됩니다. |
| 아직 write되지 않은 영역을 0xff로 채우기 위해서, 이전에는 특정 페이지가 write됐던  곳인지에 대한 정보를 들고 있었고, NBANK\*NBLK\*NPAGE 크기의 공간이  필요했습니다. 이번 과제를 할 때에는 메모리를 신경쓰면서 해보려고 하는데,  어떤 식으로 페이지가 write됐던 곳인지를 알 수 있을까요?  spare 공간을 이용하려고 생각해봤으나 그러기 위해서는 init에서 nand의 모든 페이지의 spare 공간을nand\_write로 -1로 만들어줘야 한다고 생각했는데 맞을까요? | 구현하시다보면 내부적으로 각 zone마다 어디까지 쓰였는지 writer pointer를 관리하실텐데 read요청에 해당하는 lba가 해당 zone에서 wp 이전인지 이후인지를 보면 알 수 있을듯 합니다.  zone마다 wp 하나만 관리하면 되기 떄문에 기존 page mapping FTL에서 사용한 L2P 공간대비하면 훨씬 적게 메모리를 사용할 수 있습니다. |
| 명시적으로 ppt에 fail에 대해 언급(return -1)되지 않은 경우에 대해서는 테스트하지 않는다고 생각해도 될까요? 예를 들어 전달받은 대로 write 동작을 수행했을 때 처음엔 정상작동하다가 중간부터 문제가 생기는 그런경우가 있을지 궁금합니다 | 슬라이드에 제시되지 않은 경우는 없다고 생각하셔도 됩니다. |
| EMPTY한 zone에 write를 시도하는데 첫 sector를 건너뛰고 write를 보내면 write는 실패할거같은데 OPEN으로 바꾸긴 해야하나요? | 실패하는 경우 open상태로 바꾸지 않습니다. |
| zns\_izc 함수 내부에서 zns\_write와 zns\_read를 호출하여 구현해도 되나요? 비슷한 방식으로 새로 짜는게 더 최적화가 될 것 같긴 하지만 zns\_write와 zns\_read를 호출하여 구현해도 host dram을 거치지 않고 SSD내부에서 block을 copy하게 되니 host-device traffic을 줄이려는 본래 목적은 충족시키는 것 같아 질문드립니다 | 네 강의 슬라이드에 제시된 기능을 만족시키기만 하면 편하신대로 구현하셔도 됩니다. |
| zns\_izc 함수에서 copy\_len이 0이거나 음수일 수 있나요? copy\_len이 0일 수 있으면 open, reset은 해야하나요? zns\_write나 zns\_read에서 nsect도 마찬가지로 궁금합니다 | 0인 경우는 없다고 간주하셔도 됩니다. |
| zone이 valid한 sector로 시작한다면 zns\_tl\_open만 하고 zns\_write를 아직 안했을 때 copy를 해야하나요? | copy를 바로 하는지는 구현상 자유롭게 하셔도 될듯 합니다. read 했을 때 해당 데이터가 잘 읽히기만 하면 괜찮습니다 |