Making ECC Work For Flash Part II

Optimizing Flash ECC and RAID

Steven Hetzler, IBM Fellow

Outline

- A discussion of UBER (definitions)
- The JEDEC specs and what they mean
- UBER/NRRE and RAID
- Failure targets
- DNR ECC
- New RAID codes

UBER: Uncorrectable Bit Errors

The "other" component of reliability

- UBER is when there are more bit errors than the ECC can correct
  - For example, if the sector ECC can correct 50 bits, but there are more than 50 bits in error
- One component of non-recoverable read errors (NRRE)
  - 2 outcomes of an NRRE event:
    - The ECC detects the error count is too large, and declares the sector lost
    - The ECC blissfully applies the correction and produces an incorrect value (miscorrection)
      - We tend to add CRC to detect these events and turn them into NRRE events
- I'll use NRRE going forward in this analysis

Non-Recoverable Read Errors

NRRE events

- NRRE events contribute to data loss
  - Impact depends on the system architecture
  - Loss is at least a sector worth of bits

- NRRE is specified as an interval: e.g. < 1 in 10^{14} bits
- Or as a rate: e.g. <=10^{-14} per bit

- 10^{14} bits seems really large
  - But there are 0.08 x 10^{14} bits in a terabyte!

Alternate NRRE Specifications 1

Some alternate approaches

- Express as rate per TB transferred
  - Nice for computing from data moved
  - NRRE/TB = error_interval/8 x 10^{12}

- Express as sector failure probability per operation (sector read)
  - More accurate, since we lose a sector on an NRRE event, not a bit
  - psfail = sector_bits/error_interval

Alternate NRRE Specifications 2

Some typical specifications (assume 1kB sectors)

	Consumer HDD	Enterprise SSD
Typical NRRE Spec (b)	1e14	1e17
NRRE/TB	8%	8e-5
psfail	8.2e-11	8.2e-14

Observation: the new metrics are more informative

Which may explain why the vendors haven't adopted them

NRRE Specs and Data Loss

Speed Kills

- SSDs need much tighter NRRE specs than HDDs
- SSD industry has set specs based on HDDs
  – Unfortunately, industry hasn't quite noticed the need for improvement
- We can estimate data loss rates from specs and workload
  – Workload will be small block random IO (why we use SSDs)

NRRE Specs and Data Loss

Simple to estimate

- psfail = sector_bits/error_interval
- Sector_Ops/Y = 3,600*8,760*IOPS*sectors_per_IO
- Mean Y/Sector Loss = 1/(Sector_Ops/Y * psfail)
- Can add duty cycle effects, but these are small
  - R/W typically 70/30
  - Active duty cycle ~80% enterprise, ~20% consumer

NRRE Specs and Data Loss

SSDs running at spec are at high risk of data loss

	Con HDD	Con SSD	Ent HDD	Ent SSD
IOPS (4kB)	100	10,000	350	150,000
Sector Ops/Y	1.3e10	1.3e12	4.4e10	1.9e13
NRRE Interval (bits)	1e15	1e16	1e16	1e17
psfail	8.8e-12	8.8e-13	8.2e-14	1.6e-15
Mean Y/Sector Loss	10	1	28	0.6
MTTDL (Hours)	85k	8.5k	242k	5.5k
Scaled NRRE Interval		1e17		5e18

- Both consumer and enterprise SSD NRRE specs are too loose
  - Duty cycle effects impacts consumer more than enterprise
  - Will see the effects more at high PE cycle counts
- We need tighter specs!
  - JEDEC specs (JESD218) are 1e15 and 1e16

Shorter than the MTBF!

NRRE and RAID

Protecting against device loss and sector loss

- When we use the term RAID we refer to an erasure correcting code that protects against unit loss
  - Can be hardware or software based
- NRRE impacts reliability during rebuilds
  - If there is no parity left, a sector loss becomes a data loss event
  - Occurs when rebuilding a first failure in RAID 5
  - Occurs when rebuilding a second failure in RAID 6
  - Usually higher NRRE probability than a further unit failure during rebuild
    - Rebuild windows are short
    - Declustering parity doesn't help sector loss

DNR ECC (“Do Not Resuscitate”)

Allow the NRRE (sector loss rate) at the SSD to be much greater, and let the RAID layer reconstruct the data

DNR ECC

We let sectors fail at a higher rate with DNR ECC

- Failure (at the flash layer) is acceptable in RAID with larger limits than solo devices permit

- System can be optimized by adjusting the correction at each level

- No need to try so hard at the flash layer
  - DNR – we deliberately set a higher failure rate target at the component level
  - Improves flash efficiency, simplifies encode/decode
    - Need to correct fewer errors
    - Makes the components more testable

Failure Targets

How to create data loss targets for a system

- Failure events should be expressed per unit time
  - This is how the customer experiences events
    - Not per byte, or per IO
- Program based targets
  - Look at the behavior of an entire field population
    - Helps for modeling warranty costs
    - Also helps with program financial targets
- Inputs
  - Install base
    - Unit ships per year, field lifetime, program lifetime
  - Usage characteristics
    - Total data operations, total data transferred
  - Failure tolerance
    - Depends on the failure type
    - Is it a warranty event, loss of availability, loss of data or customer near-death experience?

Modeling Failures

Precision is highly over rated here

- We need only compute first order terms!
- Why?
- Our assumptions are errors are independent of each other and of time
  - These are rarely true
    - (Well, essentially not at all with NAND…)
- The biggest deviations will be these assumptions
- So first order is good enough
  - Still a good idea to verify which terms are second order
- Thus, we can compute from binomials
  - Easy to do in a spreadsheet too!

System Data Loss Targets

Program Design	Value	Notes
Field lifetime Y	5	Typical
Mean field units	1,000,000	Assume a successful program
Units/Array	10	RAID span
IO size (kB)	4	Assume transaction processing
Total field IOs	3.15e19	Assume 50,000 IOPS/unit
Arrays/field	100,000	

Program Loss Targets	Value	Notes
Data Loss Events/program	1	For the entire program
Target Prob data loss/array/Y	2e-6	Assume a successful program

SSD Device	Value	SSD Device	Value	SSD Device	Value
IOPS	50,000	AFR	0.5%	ECC ovh bits	924
Capacity (TB)	1	NRRE	1e16	Total bits/sect	9,212
Sector kB	1	ECC Corr bits	66	Data Eff.	89%

Cumulative Binomial

Useful for estimating failures

- cumbinomial(fails,trials,errorrate)
  - Fails is the number of failures
  - Trials is the total number of events (ops, bits, etc.)
  - errorrate is the failure rate per trial (e.g. ber)

- This is the cumulative binomial distribution
  - In Excel, use the Binom.Dist function as:

    1-Binom.Dist(fails,trials,errorrate,TRUE)

    - Beware sometimes this runs out of precision when it shouldn't

RAID Down 1 Unit

Probability a RAID array has lost 1 unit in a year

- P1fail is the probability there is one failure in an array
- Probability an array is down 1 unit:
  - P1fail/Y = 1-binomial(0,arraysize,AFR) = 4.9% here
  - Not surprising:
    - 0.5% AFR * 10 units = 5%
    - 0.5% AFR = 1.75MH MTBF

Computing Sector Failure Targets

The required psfail which meets the system target

- psfail1f is the sector failure rate with 1 unit failure
  - We have a 1TB SSD and1kB sectors here

- psfail1f needed to meet array data loss target with 1 unit failure (RAID 5)
  - psfail1f = TgtDataLoss/Y / (sectorsread*P1fail/Y)
  - Sectorsread = (1TB/1kB)*(10 - 1)
  - psfail1f = 4.55e-15
  - (NRRE1f = 4.87E-19 is the equivalent NRRE to psfail1f)

Device Sector Fail and NRRE

Our device is out of spec for the system

- Recall our consumer grade SSD had NRRE 1e16
- Which has psfail = 8.8e-13
  but we need 4.55e-15!
- So, this device doesn't work here as specified
  - (No surprise, it's a consumer device)
  - But the enterprise drive at 1e-17 won't work either
- To continue, we will increase the ECC bits
  - Alternative is to limit the ber
  - Shouldn't change the answer much (either way it's a change to the SSD)

Computing NRRE Targets

Get the raw bit error rate from the NRRE

- We can compute the raw ber from the psfail spec and ECC
  1. Assume BCH 66 code on 1kB
     - Corrects 66 bit errors out of 1,024 data bytes
     - Requires 924 check bits
     - sectorbits = databits + checkbits + metadata ~ 9,212
  2. psfail = (1 – cumbinomial(66,sectorbits,ber))/sectorbits
  3. Invert by iteration to solve for ber
  4. Here: ber = 2.65e-3
  5. To meet system target need @ ber 2.65e-3 need 75 bits
     - 9,338 sectorbits

  - Hint: you can use Goal Seek in Excel to quickly iterate to find the ber

RAID 5 vs. RAID 6

Parity Groups

- Parity group (pgroup)
  - A collection of sectors that form an independent ECC set
  - In RAID 5 and 6 it's one sector from each unit
- RAID 6 has 2 parities per group
  - Can correct 1 sector/group after 1 unit failure
- Parity groups/array = 1e9 here
  - =TB/unit * units/array *1e9 / (sectors/group * sector_kB)
- Our arrays here chosen as 10 sectors/pgroup

A0	B0	C0	D0	E0	F0	G0	H0	I0	P0	RAID 5
A0	B0	C0	D0	E0	F0	G0	H0	P0	Q0	RAID 6

RAID 5 vs. RAID 6

Parity group failure on rebuild

- Sector data efficiency 0.9 RAID5, 0.8 RAID 6
  - (data sectors per group)/(total sectors per group)
- sparity is sector parities per group available after 1 unit failure
  - 0 for RAID 5, 1 for RAID 6
- Prob that a group fails to rebuild:
  pgroupfail =
  1-cumbinomial(1+sparity,sectors/pgroup-1,psfail)

RAID 5 vs. RAID 6

Rebuild failure and results

- Prob rebuild failure/array (multiple parity groups/array) prebuildfail = pgroupfail * pgroup/array
- Absolute probability of array failure/year parrayfail = prebuildfail * P1fail/Y
- Then, adjust the ECC correction bits to compute psfail until the parrayfail <= data_loss-target
  - Since this is an integer, Goal Seek in Excel doesn't do as well

DNR Results for RAID 5 vs RAID 6

RAID Type	RAID 5	RAID 6
sparity/pgroup	0	1
sectors/pgroup	10	10
pgroup/array	1e9	1e9
RAID data efficiency	0.90	0.80

Failure computations		
parrayfail	2.00e-6	2.00e-6
psfail	4.55e-15	3.37e-8
ECC corr bits	75	55
Sector efficiency	0.88	0.90

Net data efficiency	0.79	0.72

Not the answer the judges were looking for!

DNR Results for RAID 5 vs RAID 6

Sometimes you overpay for RAID 6 protection

- RAID 6 DNR doesn’t increase the efficiency
- RAID 6 has 1 sector parity per parity group
  - These double as second unit failure protection
- What we need is a more efficient class of RAID
- What about parities designed for sector loss?
- fpof – first point of failure
  - The minimum number of losses that cause a RAID failure

PMDS Codes

Optimized for both device and sector protection

- New RAID codes designed for this very problem
  - (I know, I was there at the time)
  - Parity group is now multiple sectors from each device (columns)

A0	B0	C0	D0	E0	P0
A1	B1	C1	D1	E1	P1
A2	B2	C2	D2	E2	P2
A3	B3	C3	D3	E3	P3
A4	B4	C4	q_a	q_b	P4

P0 is row 0 parity (Example with 6 units)

P1 is row 1 parity

P2 is row 2 parity

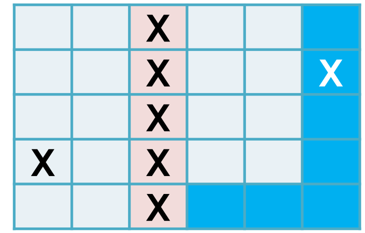
P3 is row 3 parity

P4 is row 4 parity, q_a, q_b group parities

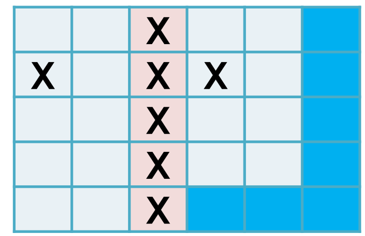
- Unit loss protection via row parities P
- Floating sector loss protection via group parities q
  - The q can be placed anywhere in the parity group
  - They are invoked only *after* more than 1 sector in a row is lost

RAID 5++

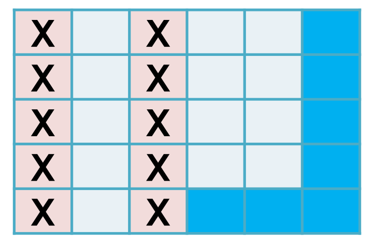
- **RAID 5++ in most cases stronger than RAID 6**
  - Consider rebuild (1 unit fail)
  - RAID 6:
    - Correct all 1 sector fail/row
    - Correct 0 2 sector fail/row
    - fpof **2** sectors + 1 unit
  - RAID 5++ :
    - Correct 2 1 sector fail/row
    - Correct 1 2 sector fail/row
    - fpof **3** sectors + 1 unit



2 rows with 2 fails
RAID 6 ✓ OK
RAID 5++ ✓ OK



1 row with 3 fails
RAID 6 ✗ FAIL
RAID 5++ ✓ OK



2 unit fails
RAID 6 ✓ OK
RAID 5++ ✗ FAIL

- **RAID 5++ is stronger to sector failure on rebuild**
- **RAID5++ is weaker to unit fails**
  - Mitigated by short rebuild time

Flash Memory SUMMIT

DNR Results for RAID 5+ and RAID 5++

RAID Type	RAID 5	RAID 6	RAID 5+	RAID 5++
sparity/pgroup	0	1	1	2
sectors/pgroup	10	10	160	1,280
pgroup/Array	1e9	1e9	6.25e7	7.81e6
RAID data efficiency	0.90	0.80	0.89	0.90

Failure computations				
parrayfail	2.00e-6	2.00e-6	2.00e-6	2.00e-6
psfail	4.55e-15	3.37e-8	7.2e-9	2.47e-7
ECC corr bits	75	55	56	52
Sector efficiency	0.88	0.90	0.90	0.91

We have a winner

Net data efficiency	0.79	0.72	0.81	0.82

DNR Results with PMDS Codes

RAID 5++ makes DNR ECC cost effective

- Efficiency is increased by letting the NRRE (psfail) increase
  - Up to 3% more efficient in this example
- May not sound like much, but worthwhile
  - Goes straight to margin
    - What else would you do for 3 margin points?
  - Can also be used to increase yields
  - May save cost in ECC decoders
  - Allows use of consumer parts in enterprise applications
- This was just a simple example, we may be able to do better with other configurations
- If you need dual failure protection, there are PMDS codes for those as well
  - If 2nd parity is protecting against a second unit failure, it's not available for sector loss protection
    - I have shown you how to do the math

Summary

- **Showed that UBER/NRRE specs for SSDs are inadequate**
  - Time to data loss spec should be similar/better than HDD
- **Have shown how to compute system reliability targets**
- **DNR ECC can achieve higher data efficiency**
  - Allowing higher sector failure rates (NRRE) improves system cost
- **PMDS codes such as RAID 5++ make DNR economical**
- **I will post a spreadsheet for downloading on my blog at smorgastor.DrHetzler.com**
  - Shows the details of the calculations for the interested student
- **If you want to see actual data on SDD bit error rates and causes, attend my Tutorial T1 at 8:30 on Wed 8/6**