

Just One More Maintenance

Operating the perlmutter Supercomputer while
upgrading to Slingshot 11

Douglas Jacobsen, NERSC, LBNL

2023/05/11



Acknowledgments

- Everybody at NERSC, but especially:
 - Computational Systems Group
 - N9 User Integration Team
 - N9 Project Management Team
 - Early User Testers
- HPE/Cray
 - Site Support Team
 - Extended-on-site Installation Team
 - N9 Project Management Team
 - HPE/Cray Engineers on Slack and Zoom
 - Cray EX Leadership Team (Hardware and Software)

NERSC Perlmutter Supercomputer



Phase I: Arrived spring of 2021

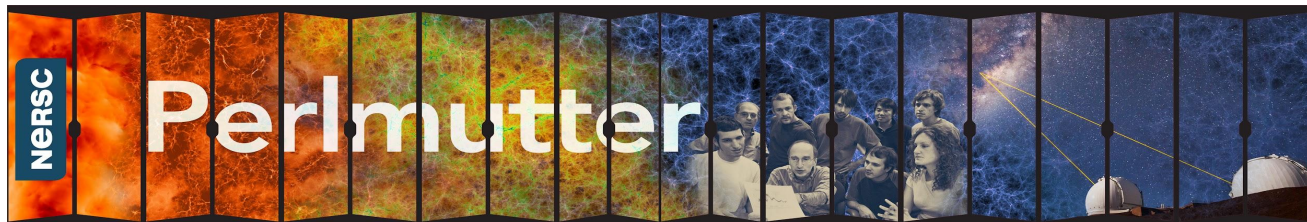
- 1,536 GPU-accelerated nodes
- 1 AMD “Milan” CPU + 4 NVIDIA A100 GPUs per node
- 256 GiB CPU memory and 40 GiB per GPU high BW memory
- 35 PiB FLASH scratch file system
- Slingshot 10

Full Initial Operations November 2021

Phase II: Arrived early 2022

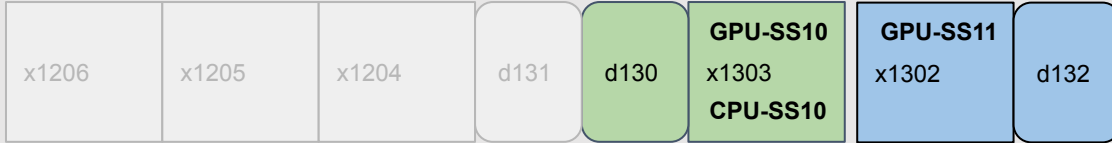
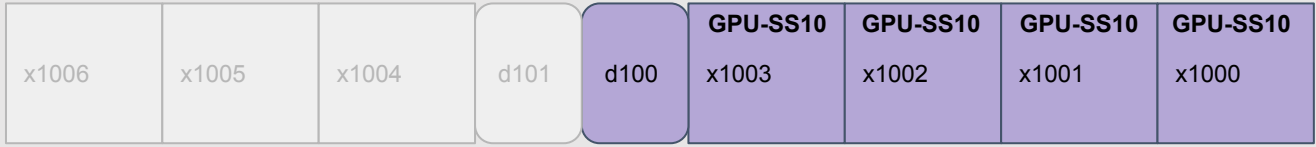
- 3,072 CPU only nodes (2 AMD “Milan” CPUs per node)
- 512 GiB memory per node
- **Upgrade high speed network to Slingshot 11**
- CPU partition will match or exceed performance of entire Cori system
- Additionally added two GPU racks (+256 nodes), NVIDIA A100 with 80GiB HBM per GPU [not phase2, just done at same time]

Getting close to Full Operations (Not There Yet)

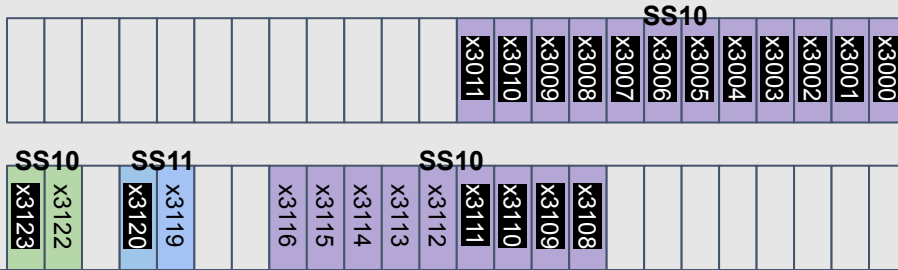


Phase 1

January 2022



-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw



What is a Maintenance?

Typical Rolling Maintenance:

- Any user environment software update
 - SLES, cos, uan, slingshot host software, cpe, slurm, OS configurations, etc
- Done by:
 - Reserve/drain 20 login nodes, 64 GPU compute nodes, 64 CPU compute nodes
 - Rebuild environment on the reserved set
 - Reframe test/checkout of the reserved set
 - All at once, kick users off the live login nodes, then `scontrol reboot ASAP` all the non-reserved compute nodes and login nodes
 - Open reserved login nodes and compute nodes to users

What is a Maintenance? Part Two

- CSM Major Update - Rare
 - Have to rebuild ncn-[smw]* nodes
 - Mostly live/rolling, mostly depending on type of ceph update
 - Biggest complication is DVS during worker node updates
 - see notes at end for details on switching to NFS
 - Can take awhile, automate with wrapper around argo (csm 1.2 and after)

What is a Maintenance? Part Three

- CSM Minor Update - Rare
 - Just the microservices
 - Fully Live Update
 - Biggest complications:
 - Have to rewrite portions of the dns/unbound/kea updates to not empty DNS for us (this is fixed in CSM 1.4, prior to that we just hacked around it)
 - Restarting nexus has a higher than average likelihood of causing urgent problems (no real user impact, but major system operations impact)

What is a Maintenance? Part Four

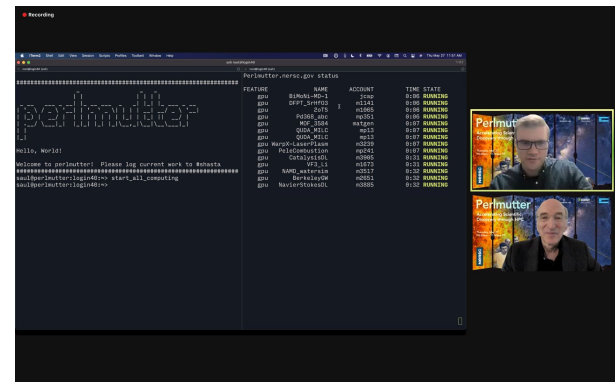
- Slingshot FMN/Switch Firmware/Network Topology Update
 - At present, all switch firmware needs to be uniform
 - Updating firmware completely de-configures the network, massively disruptive
 - SS < 2.0 rewrites of fabric_template.json can change algomacs
 - Because switch identity within group not conserved
 - Can be overcome by using SS 2.0 tooling, or massaging fabric_template.json prior to import
 - Have a well ordered ~120 step procedure for rebuilding fabric. Takes about 35-85 minutes depending on how obstinate the switches are
 - NERSC has a lot of tooling and monitoring around this to evaluate cable quality, network status, and ensure consistent results

What is a Maintenance? Part Five

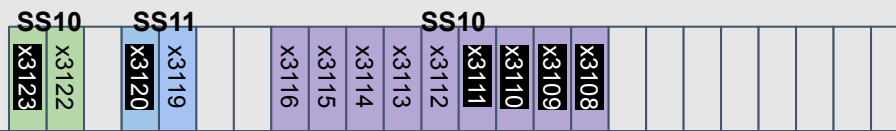
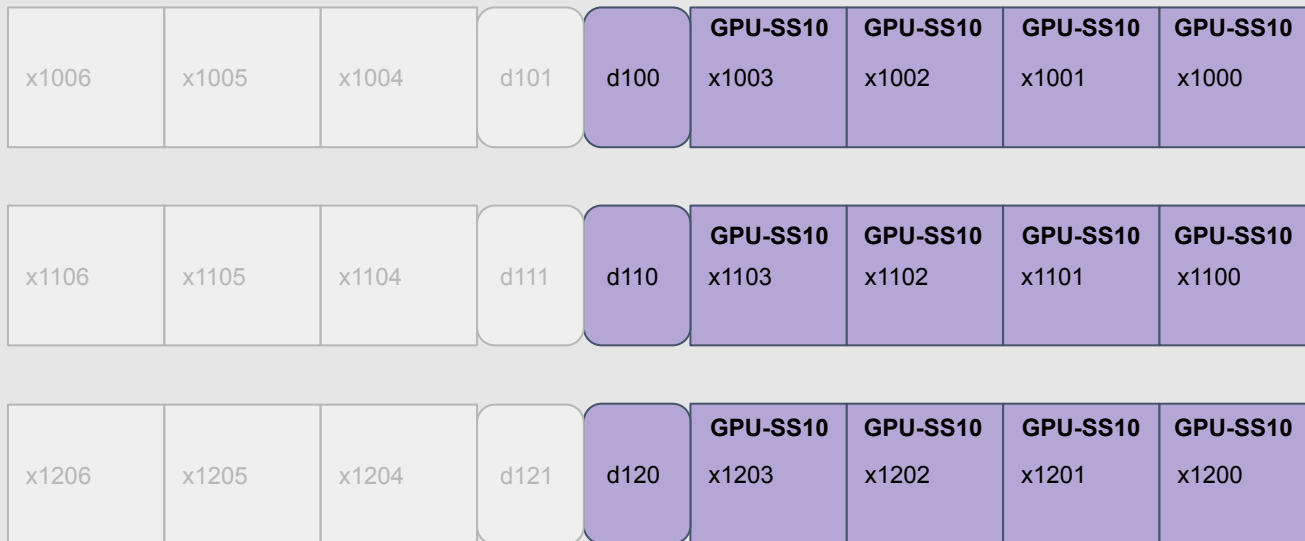
- Lustre Update
 - Almost always done disruptive, have tested rolling
- This is often driven by Slingshot Host Software updates or configuration changes, sometimes lustre, more rare.
- Disruptive update steps:
 - Shut all clients down
 - Do any slingshot network action
 - Run NEO updater [this requires tuning per-maintenance to meet the situation and increases scalability parameters]
 - Bring clients up in new software


Deployment Timeline

- December, 2020: Filesystem Delivery
- February, 2021: GPU Cabinet Delivery
- May, 2021: Perlmutter dedication ceremony
- July 16, 2021: first users on Perlmutter
- July - October, 2021 : Initial users added in waves (NESAP, ECP, SF)
- November, 2021: Full production for phase 1 (not ALL users on)
- February - April, 2022: GPU nodes cycled out for cleaning/conversion to PGW coolant



February 2022



-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw

Test Systems Keep Things Stable for Users

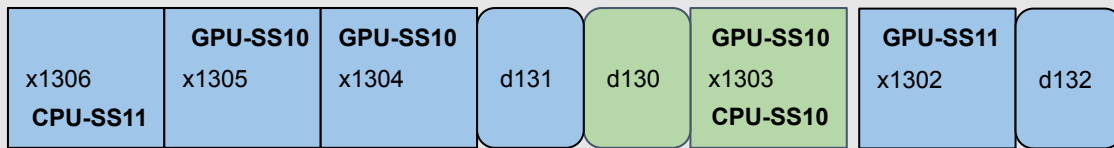
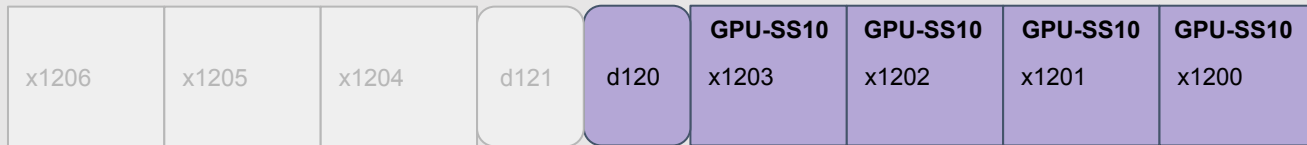
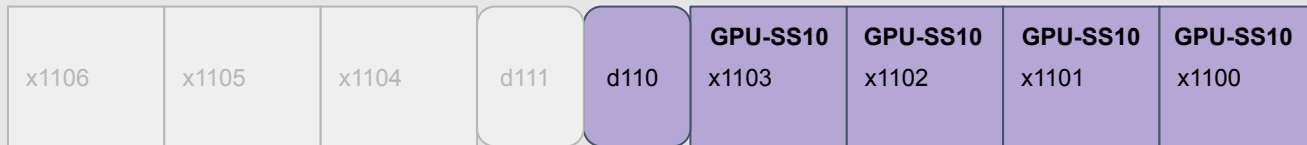
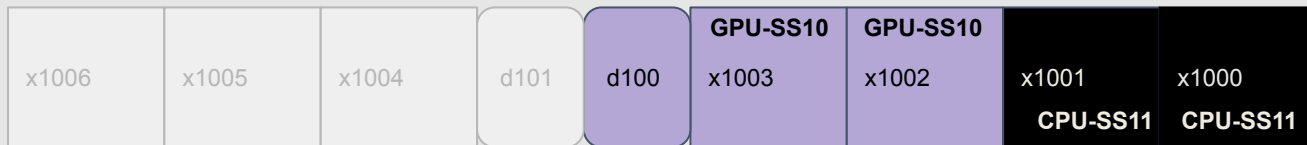
- NERSC has deployed two test systems for Perlmutter
 - Muller: pre-production, test new software and system configurations
 - Alvarez: test far-field changes that require a lot of development
- A major question before phase 2 integration was “Can SS10 and SS11 co-exist?”
 - Needed new procedures for managing hybrid system (no manual covers this because NERSC created it)
 - Does lustre perform with TCP/IP communication (RDMA wasn't possible between SS10 and SS11)?
 - Does the system function at all this way?!?
- Added SS10 GPU and SS11 CPU cabinets to Alvarez
 - Gave systems group a platform to set up configuration without disrupting Perlmutter
 - Early user testing to ensure that SS11 will work and can co-exist with SS10
 - Thanks to all the users who helped with testing!



Deployment Timeline Con't

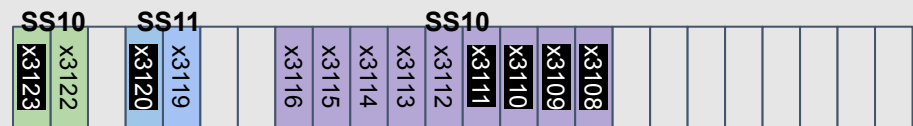
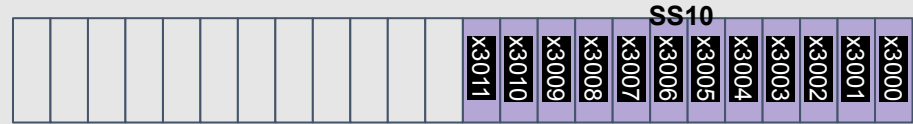
- February - June, 2022: CPU cabinets delivered
- May 11, 2022: First CPU nodes opened to users, **switch lustre from ko2ibld to ksockld**
- May 17, 2022: All NERSC users added to Perlmutter
- June - October, 2022: **remanufacturing**



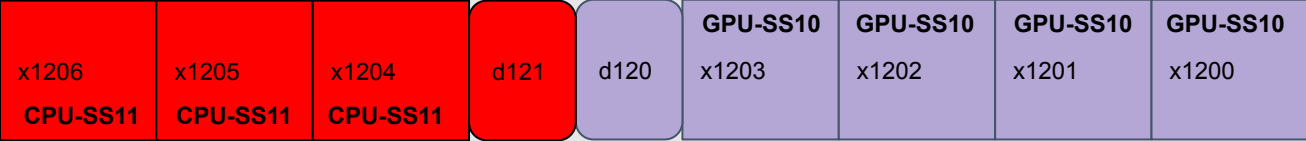
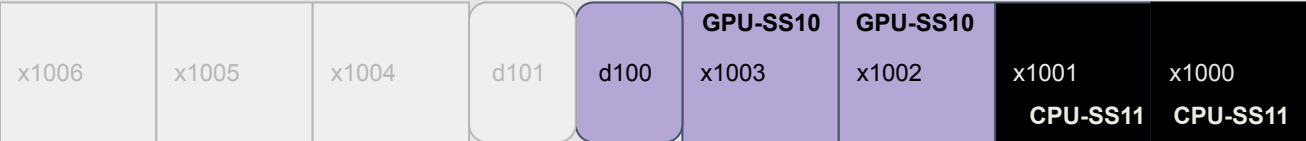
March 2022



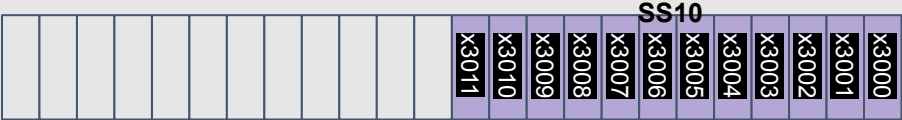
-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw



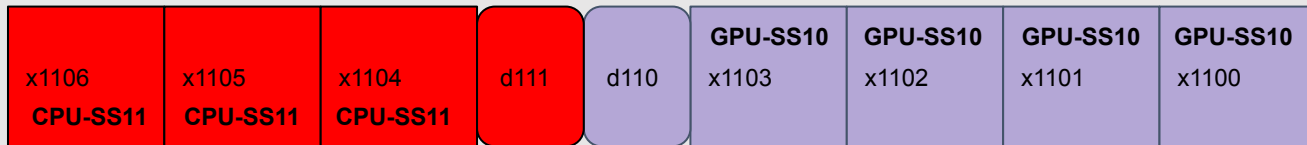
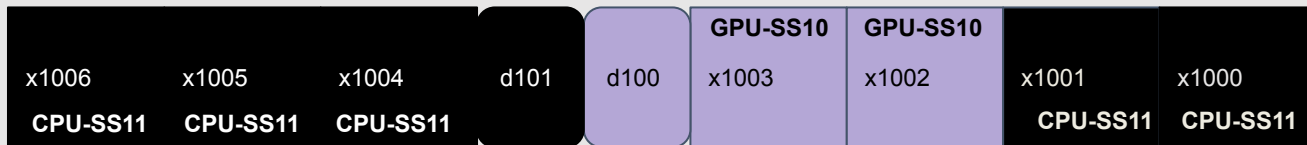
April 2022



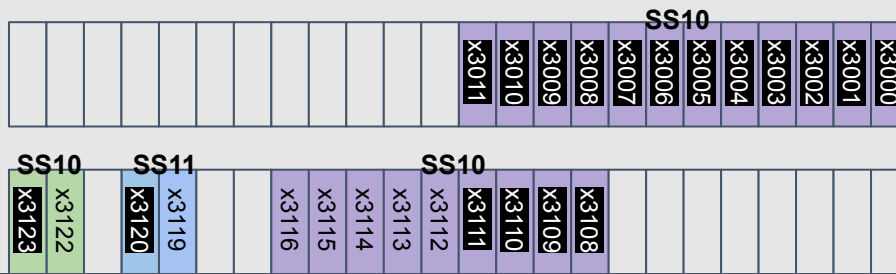
- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw



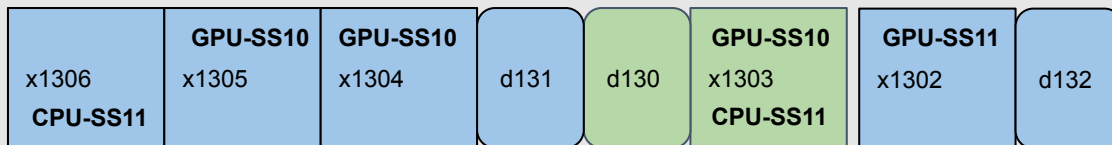
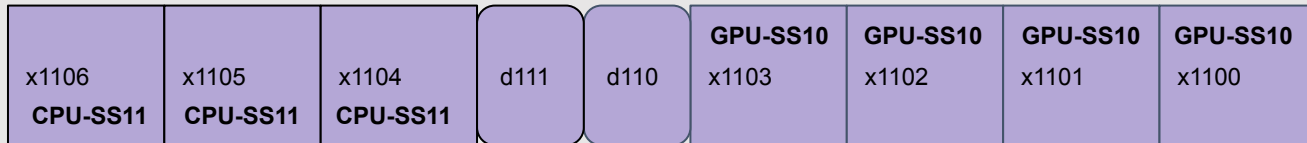
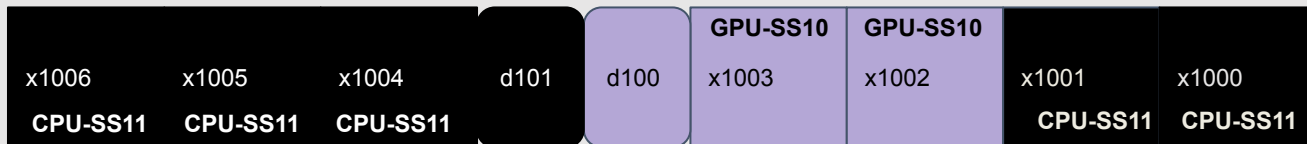
June 2022



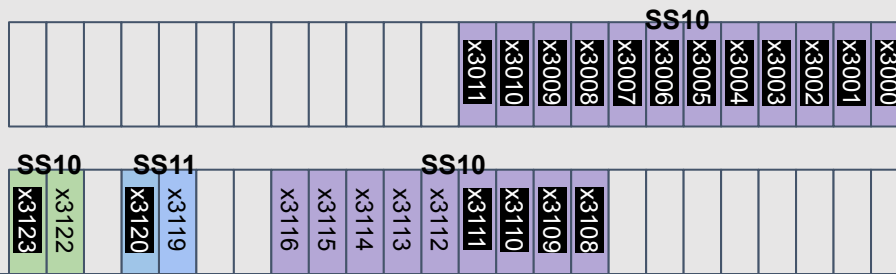
-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw



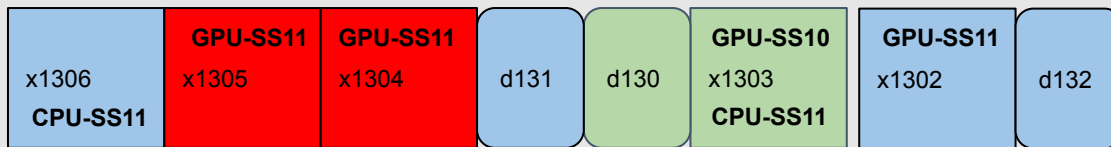
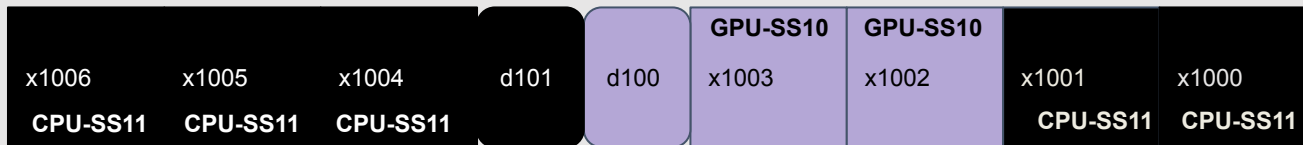
June 2022



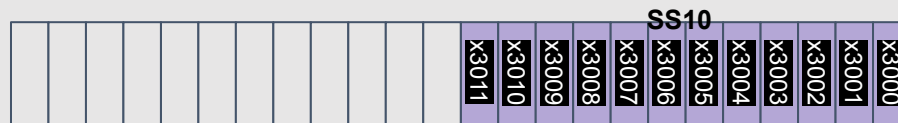
- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw



June 2022

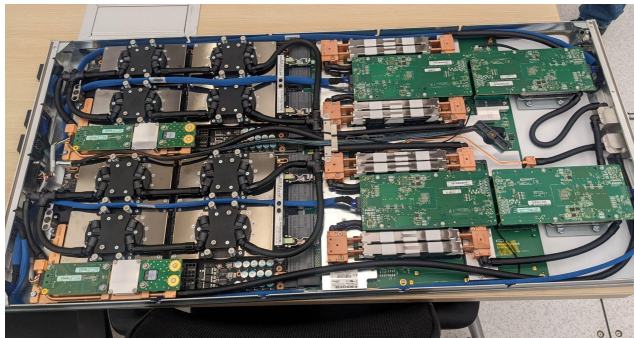


- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw

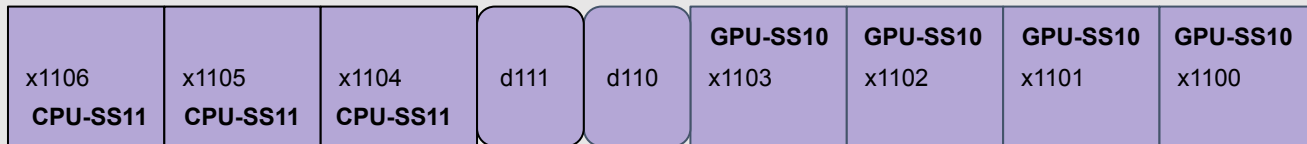
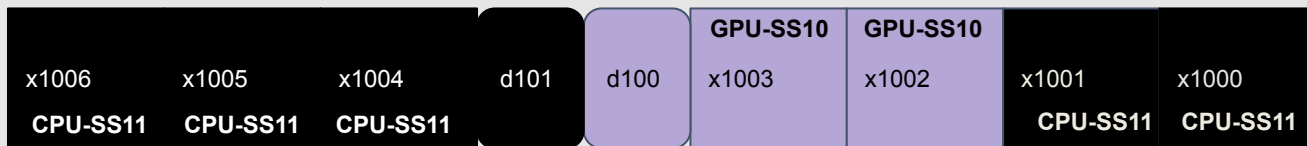


Rebuilding a Supercomputer

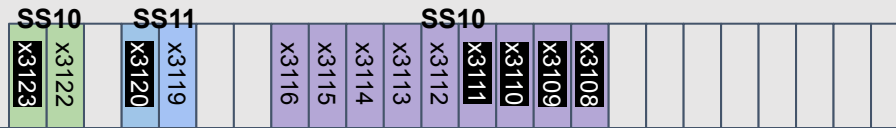
- The Slingshot 11 NIC was delayed, system was delivered with Slingshot 10
- To get to phase II, each GPU node had to be physically upgraded to the hardware configuration needed for Slingshot 11, along with a major software upgrade to fully utilize the new hardware
- The remanufacturing was done over a period of five months by a team of HPE/PGTEK personnel
 - 788 Blades
 - 300 Storage Servers
 - 48 Login Nodes
 - 28 I/O Gateways
 - 36 Mgmt Servers



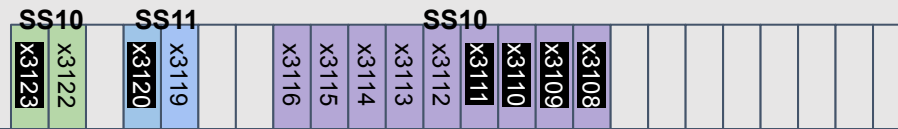
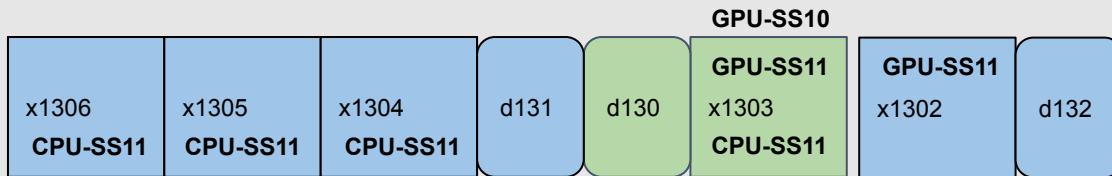
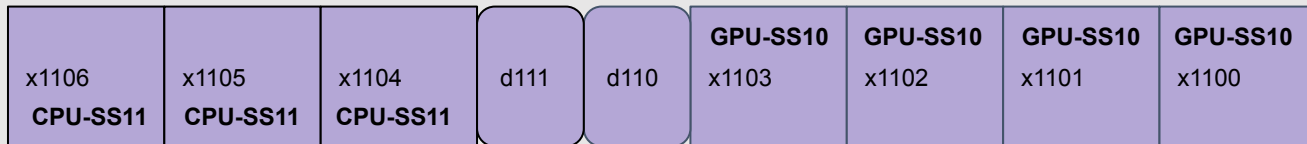
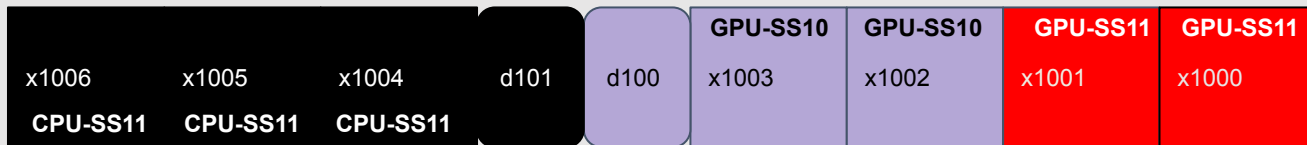
July 2022



- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw

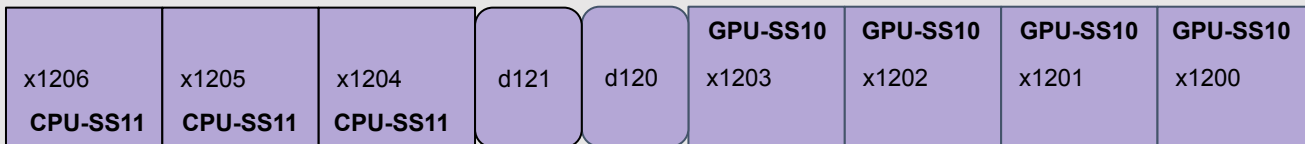
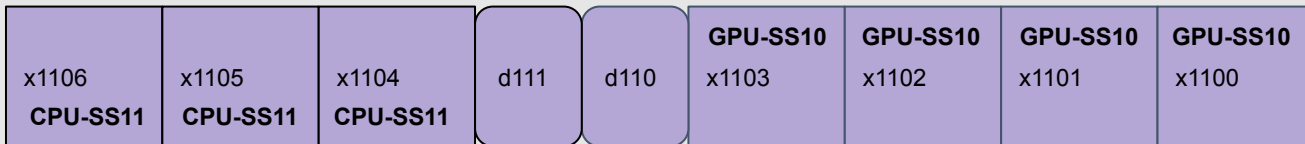
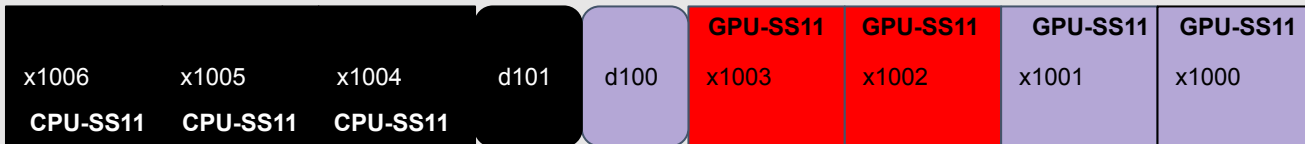


July 2022



-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw

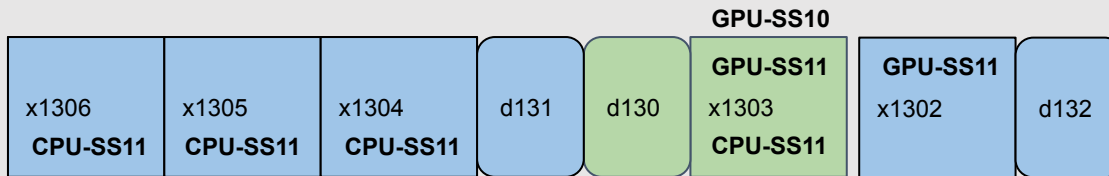
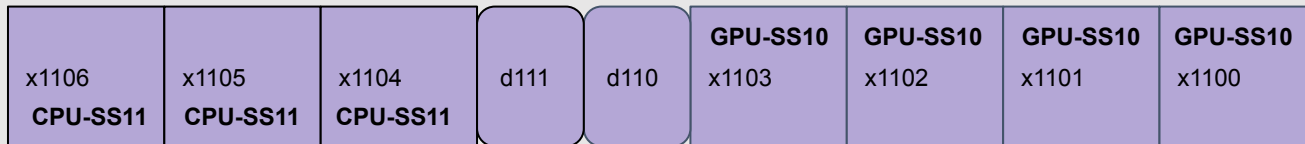
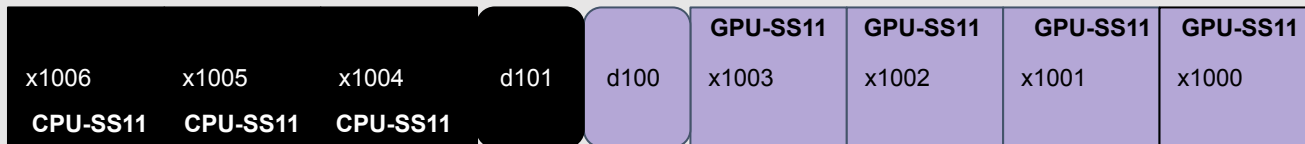
July 2022



-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw

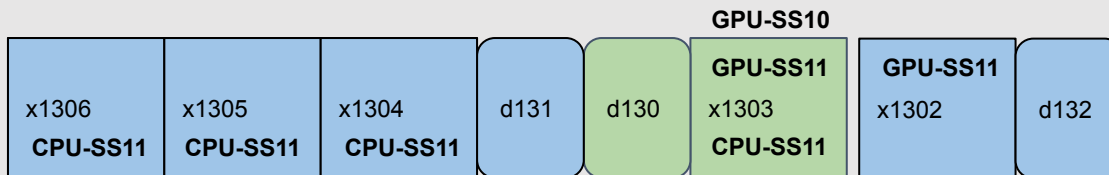
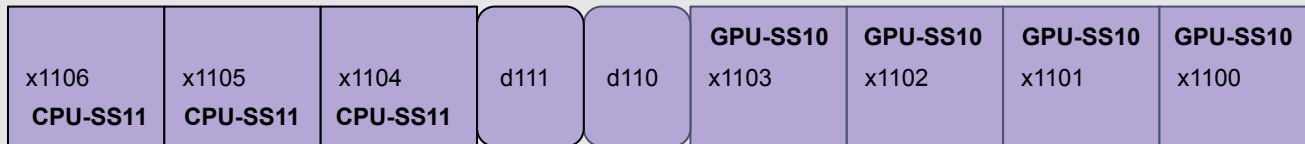
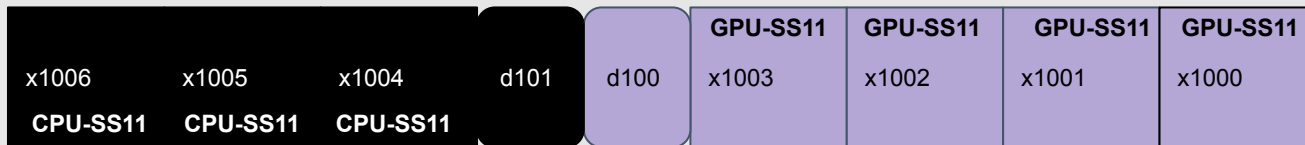


July 2022



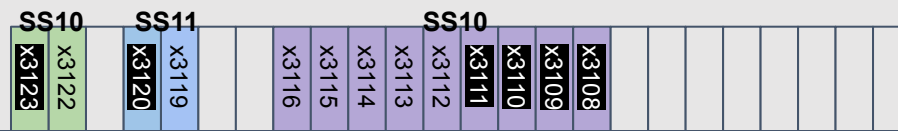
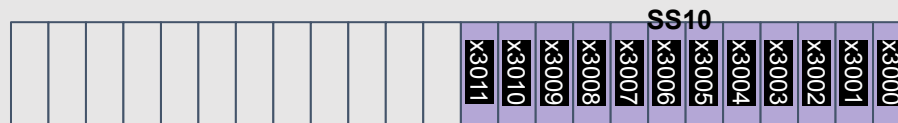
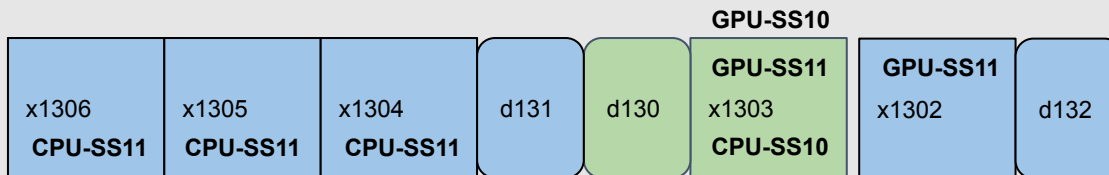
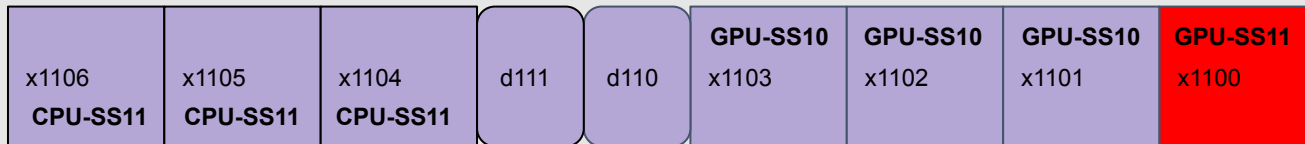
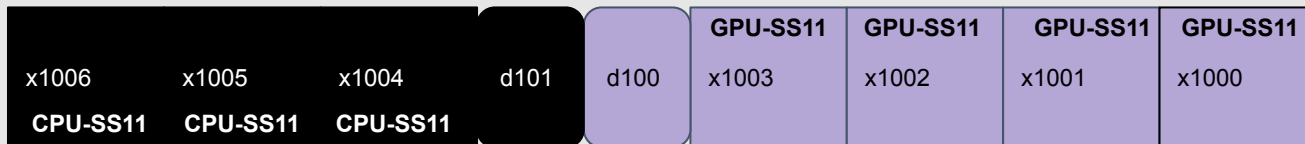
- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw

July 2022



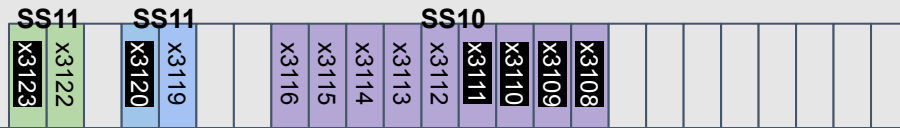
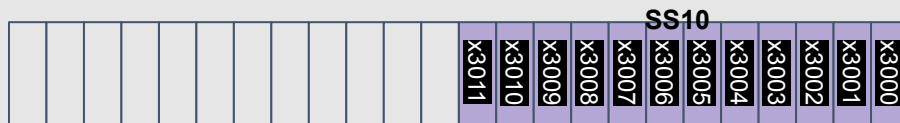
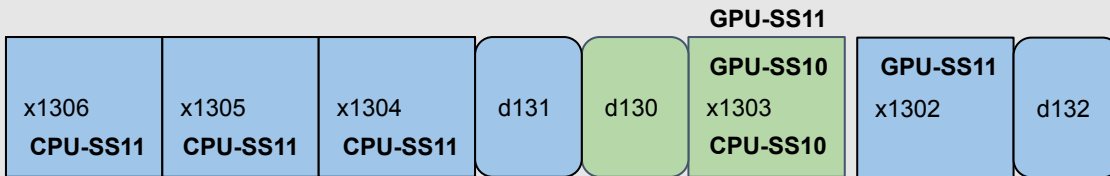
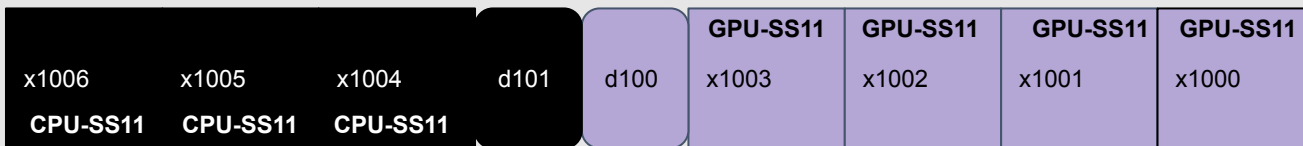
- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw

August 2022



- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw

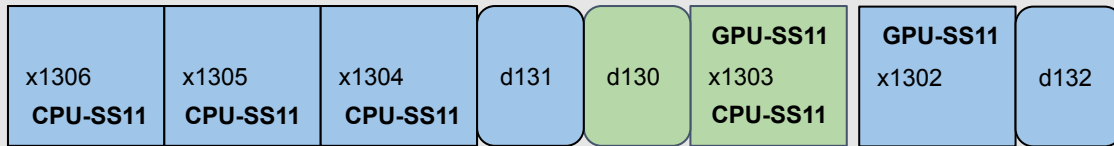
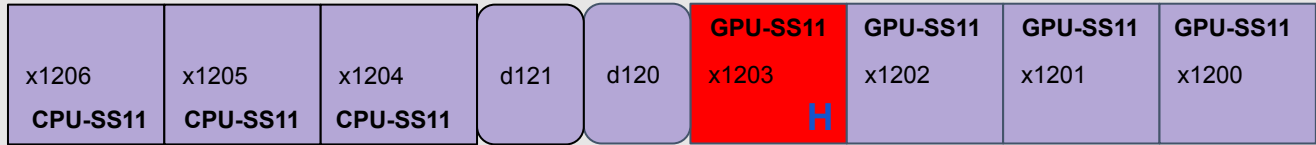
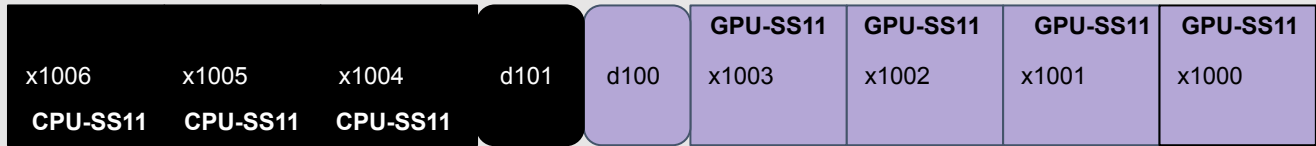
August 2022



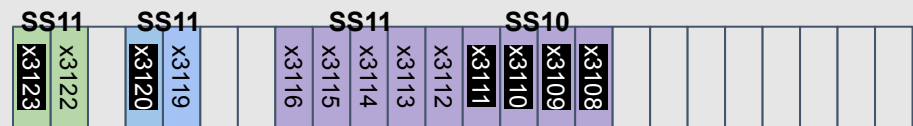
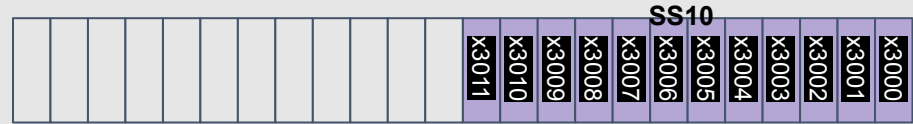
- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw

August 2022

Major CSM
Upgrade to 1.2



- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw



August 2022

x1006	x1005	x1004	d101	d100	GPU-SS11	GPU-SS11	GPU-SS11	GPU-SS11
CPU-SS11	CPU-SS11	CPU-SS11			x1003	x1002	x1001	x1000

x1106	x1105	x1104	d111	d110	GPU-SS11	GPU-SS11	GPU-SS11	GPU-SS11
CPU-SS11	CPU-SS11	CPU-SS11			x1103	x1102	x1101	x1100

x1206	x1205	x1204	d121	d120	GPU-SS11	GPU-SS11	GPU-SS11	GPU-SS11
CPU-SS11	CPU-SS11	CPU-SS11			x1203	x1202	x1201	x1200

x1306	x1305	x1304	d131	d130	GPU-SS11	GPU-SS11	d132
CPU-SS11	CPU-SS11	CPU-SS11			x1303	x1302	

-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw

SS10

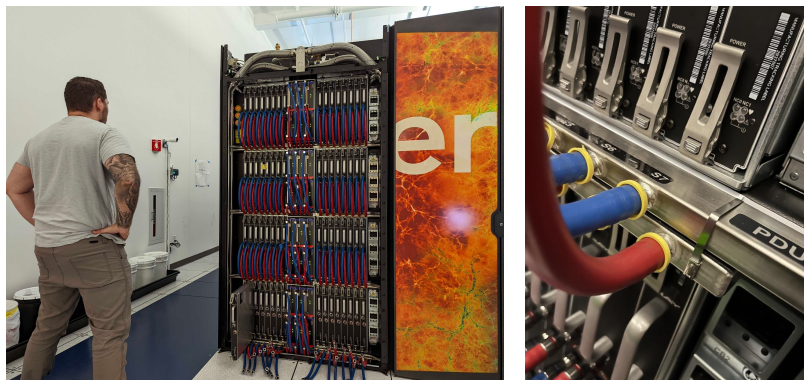
											x3011	x3010	x3009	x3008	x3007	x3006	x3005	x3004	x3003	x3002	x3001	x3000
--	--	--	--	--	--	--	--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

	SS11	SS11					SS11				SS10												
	x3123	x3122					x3116	x3115	x3114	x3113	x3112	x3111	x3110	x3109	x3108								

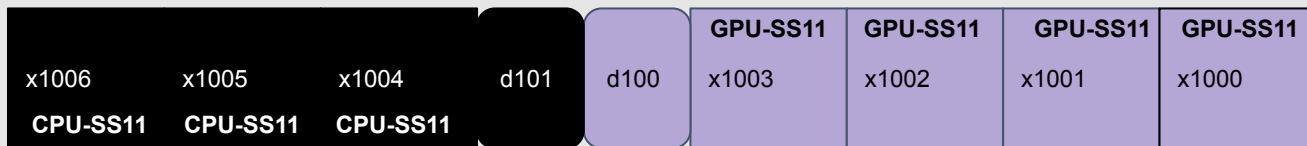


Deployment Timeline Con't

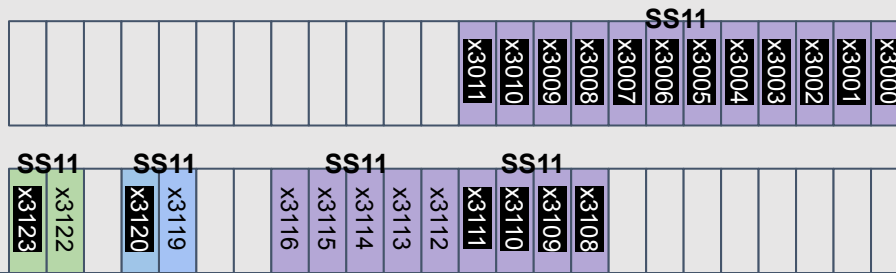
- July 11, 2022: First SS11 GPU nodes opened to users
- August 15, 2022: All SS10 GPU nodes are removed from the system
- August 15 - ~~August 23~~ September 15, 2022: All Lustre Servers upgraded from SS10 to SS11 - **No Lustre on Perlmutter**
- October 11, 2022: Remanufacturing finished, I/O network protocol changed to the final phase II configuration (kflind), lustre upgraded to 2.15



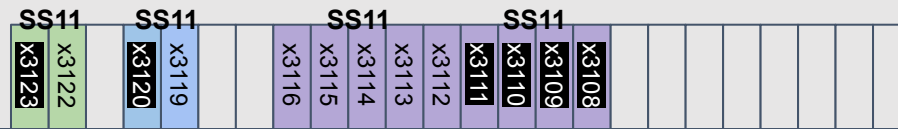
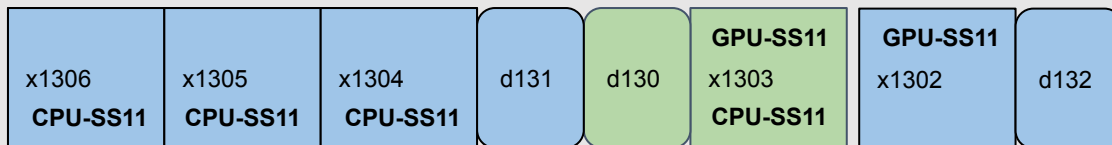
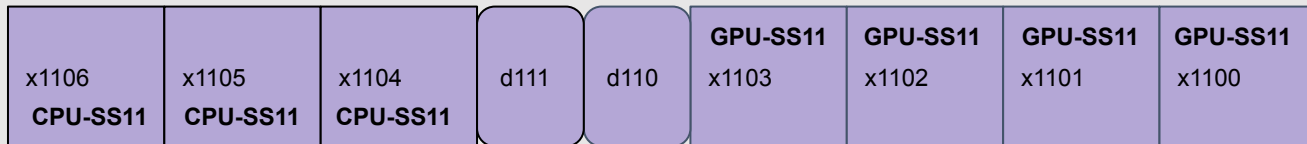
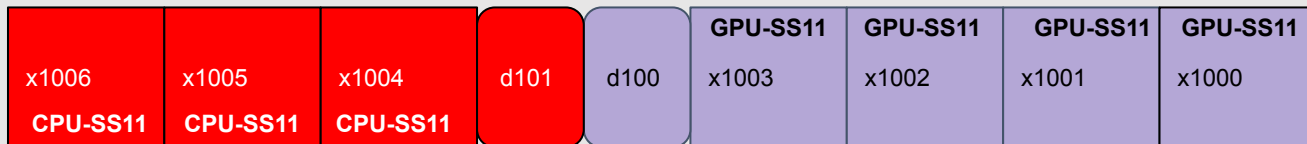
September 2022



- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw



October 11, 2022

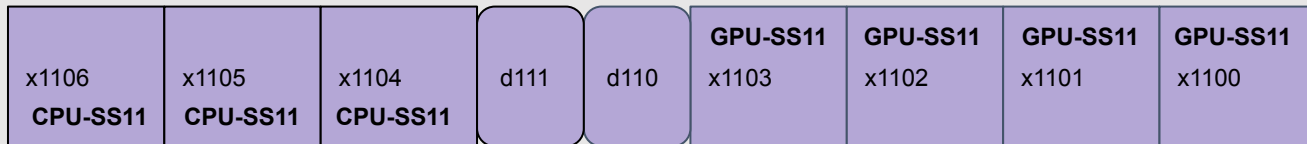
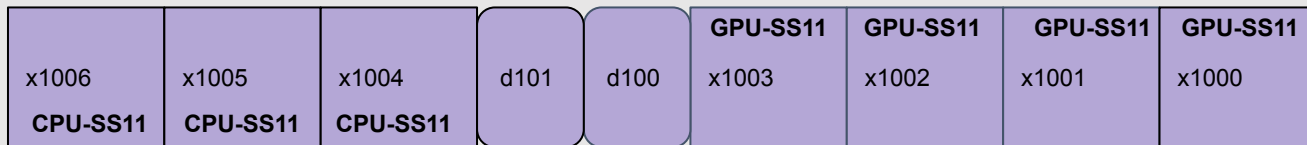






- testing
- not present
- perlmutter
- muller
- alvarez
- x storage/gw

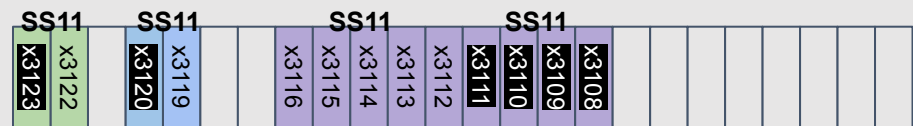
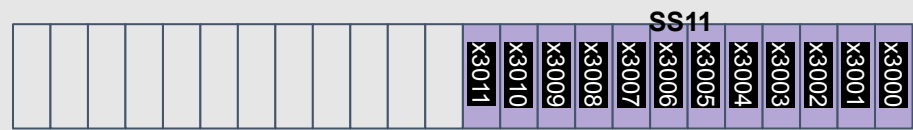


October 25, 2022

CSM 1.3 Major
Upgrade
Cos 2.4
Slingshot 2.0

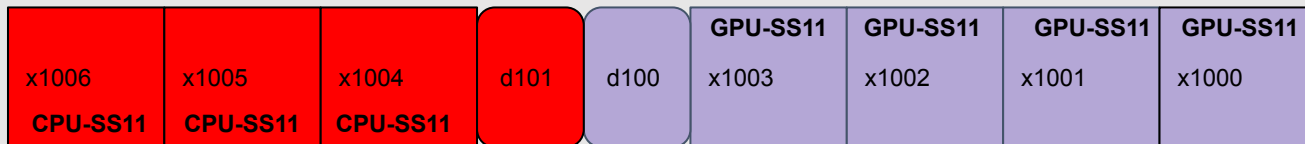


-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw

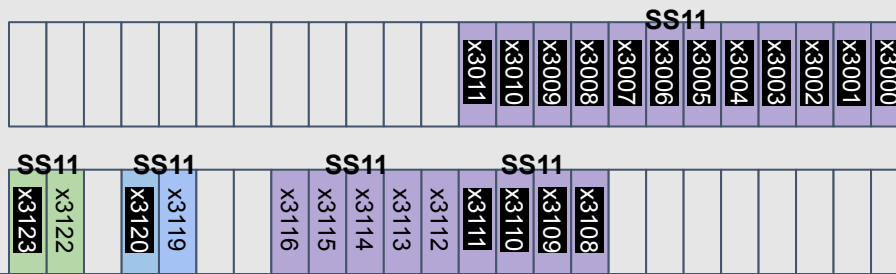


October 26, 2022

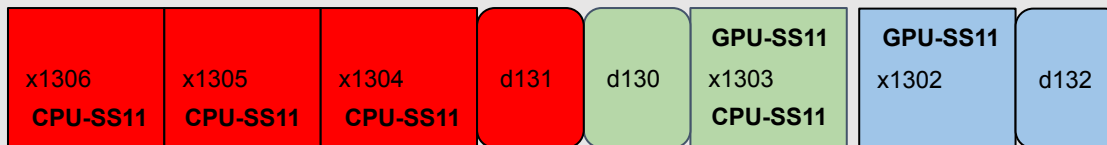
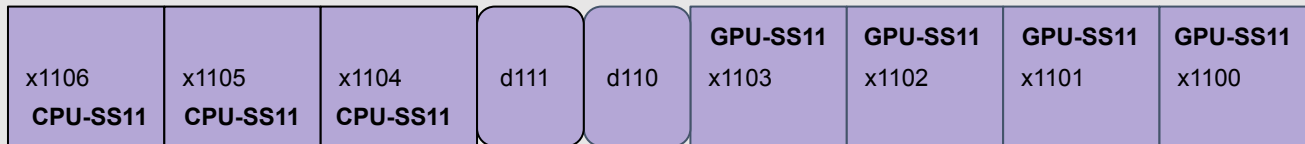
Major system wide issues tied to network. Remove recently added row of CPU cabinets to correct and send back to test. Issues with L1 cabling and switches.



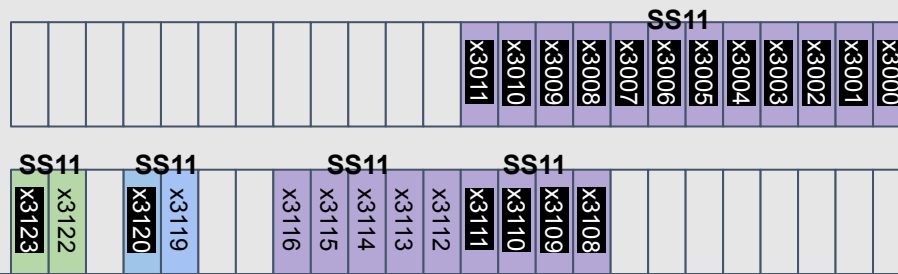
-  testing
-  not present
-  perlmutter
-  muller
-  alvarez
-  storage/gw



November 5, 2022



- testing
- not present
- perlmutter
- muller
- alvarez
- storage/gw



Summary of Phase 2 Upgrade

- Extensive test protocol on "SuperAlvarez" test system:
 - Does SS11 work at all yet?
 - Does mixed SS10/SS11 work?
 - Does GPU/SS11 work for our users?
 - Does kfilnd for DVS? For Lustre work correctly?
- Had parallel activities to integrate CPU/SS11 nodes and upgrade GPU/SS10 -> GPU/SS11
 - All hardware checkout/validation done using an HPCM test cart
- Onsite large scale service of all nodes is *challenging*
 - River nodes are concerningly "pet"-like
 - Lustre viking node upgrade much more difficult than expected
 - Specific hardware issue (resolved)
 - Cable arms a difficulty for onsite
 - HSN cabling challenging, especially so on NEO

Hardware Stabilization (2023/01 - Present)

- Blade swaps in/out of hospital have diminishing returns
 - In CSM undiscovery/discovery is a complicated and lengthy process
 - Reasonably high likelihood of having blade seating issues or other hardware problems with frequent moves
- Created a software "debug" capability in CSM
 - Same compute image, different configuration
 - No Parallel Filesystems
 - Limited Ansible (fast boot)
 - Reserved Slurm access for HPE personnel to test nodes using target software environment

Software Stabilization (2023/01 - Present)

- Once system in final hardware configuration, slingshot 2.0 (never tested on SuperAlvarez), brought online many features for the HPC protocol - and many challenges
- NERSC has been iterating COS, SHS, Slingshot, and NVIDIA driver software in search of a version suitable for user needs
 - Have a lot of automation driving upgrades and process leveraging CSM APIs to ensure high fidelity transfer from test systems to production (and scale up)
 - DVS on worker nodes limited productivity by tying the system up with unnecessarily complex maintenance burden -> NFS
 - Reduction of role of CFS/Ansible has sped boot and improved productivity (number of nodes reaching fully ready state)

NERSC Stack:

- * CSM 1.3.2 equivalent
- * cos 2.4.116 (DVS)
- * shs 2.0.2-108 (libfabric)
- * NEO 6.3-023 (SHS)
- * NVIDIA 525 driver (libfabric)

One More Maintenance

Nov 2 2021 Rolling
Nov 16 2021 Rolling
Dec 8 2021 Rolling CSM Major
Dec 21 2021 Rolling
Jan 11 2022 Rolling
Jan 24 2022 Rolling
Feb 24 2022 Rolling
Mar 10 2022 Rolling
Mar 24 2022 Rolling
Apr 7 2022 Rolling
Apr 30 2022 Rolling
May 17 2022 Slingshot
Jun 7 2022 Slingshot
Jul 11 2022 Slingshot

Jul 18 2022 Slingshot
Aug 2 2022 Slingshot
Aug 8 2022 Slingshot CSM Major
Aug 23 2022 Slingshot
Sep 8 2022 Slingshot
Sep 15 2022 Slingshot
Oct 6 2022 Slingshot
Oct 24-26 2022 Slingshot CSM Major
Dec 21 2022 Slingshot
Jan 19 2023 Slingshot
Feb 9 2023 Slingshot
Feb 16 2023 Slingshot
Feb 23 2023 Slingshot
Mar 8 2023 Slingshot
Apr 11 2023 Rolling
Apr 17-21 2023 Dedicated Testing
Apr 27 2023 Rolling
May 4 2023 Rolling

Overall Summary

- NERSC has successfully made perlmutter available to users during a very complex install/upgrade process
- Decision to phase the upgrade did deliver more user cycles and has driven tremendous community productivity
- Large scale work remains challenging
 - New software features = New bugs and time to iterate them out
- Medium scale testing is successful, especially bringing key users into that process
 - Will productionize this further with future cabinet additions to perlmutter
- Minimizing complex dependencies (e.g., dvs -> nfs for OS boot) greatly increases flexibility

Next Steps

- Resolve remaining hardware and software blocking issues (very few)
- Enter initial production for phase 2
- Start brining k3s/UAs online in support of improved workflow capabilities
- Vacation (for everyone, not all at once)



Thank You

General Notes on Managing Slingshot

- Especially during times of high changes
 - NERSC `cable-check.sh/parseNetwork.py` are run after EVERY cable change. HPE has related tooling now shipping with recent releases
- If using SS11, there are new features being enabled and many bug fixes tied to adapting the software stack to those changes
 - **RUN THE LATEST SLINGSHOT HOST SOFTWARE**
- With kfilnd / HPC protocol jobs running, we think its essentially impossible to replace a switch live at this time.
 - Have still not found a path through that drains links before taking it down

DVS -> NFS for CSM Notes (Part 1)

Server side:

We wanted to implement something that didn't get in the way of the existing DVS infrastructure and did not preclude using both schemes.

Uses kernel nfs-server running on ncn-w's

A cephfs subvolume holds the boot artifacts in the same directory structure as the boot-images s3 bucket. We were not able to come up with a way to export the boot-images s3fs mount directly from the workers in a way that allowed redundancy and fault-tolerance because of the issues with s3fs inode numbering.

Someday a single cps content manager pod could manage the content in the cephfs subvolume thinking it's /var/lib/cps-local (in legacy non-S3FS mode), but for now we sync the necessary artifacts from the boot-images s3 bucket when we create BOS session templates in order to leave the existing CPS/DVS configuration untouched.

A set of workers mount and export the directory. On muller and alvarez, it's all 5 workers although fewer would be enough. On perlmutter we just chose 8 workers I think. We'll see how the first boot goes.

Exported via nfsv3 so no nfsv4 session/state problems when moving from one server to another.

Because it's the same cephfs directory on the workers, inode numbers are the same on the workers. Therefore NFSv3 file handles are portable across the workers.

Load balancing via keepalived using a single IP dedicated to the NFS service. All the mount requests appear to happen on whichever server actually had the IP on an interface, but the port 2049 traffic is distributed pretty fairly.

Failover was tested by stopping and starting keepalived on the workers while many clients frantically dropping cache and reading the mounted squashes; failover seems to work fine.

Possible alternative scheme we didn't think of initially and haven't tested: Ganesh servers on the workers with NFSv3 proxy to the ceph cluster instead of the workers needing to mount and export the CephFS directory. I think this alternative may allow direct NFS export of the boot-images bucket via NFS on the ceph cluster so no copying of anything required. But I am not sure if the ceph cluster might restrict itself to NFSv4, and not sure what session/state problem that might introduce. Also, not sure if proxy (vs caching offered by the cephfs mounts on the workers) would be too much for the NMN.

DVS -> NFS for CSM Notes (Part 2)

Client side:

NERSC BOS session template scripts manage the rootfs transport specification on the kernel command line via the existing `rootfs_provider_passthrough` specification in the session template.

Transport `dvs` works like it always did.

Transport `nfs` implies a default server `server`

Transport `nfs=foo` allows specification of a non-default server by name or IP for testing or any other reason.

The only client-side file that needed to change was `/opt/cray/cps-utils/bin/cpsmount.sh`

Changes to `cpsmount.sh` accomplished by patching from a Nersc ansible role at image build time - needs to make it into `initrd` to cover the `rootfs` mounts. No changes to `cray` ansible required.

Trivial change to `cpsmount.sh`: add the `nfs` case to the `transport` case statement (2 or 3 lines)

Trivial change to `cpsmount.sh`: examine `/proc/cmdline` to decide which `transport` to use, ignoring command line argument - allows all the `cray` ansible mounts that call `cpsmount.sh -t dvs` to use NFS without realizing it.

Trivial change to `cpsmount.sh`: Skip the call to CPS for NFS mounts. Originally we preserved the call (with fake `transport DVS`) so that it could generate the path even for NFS mounts - this would have allowed a single CPS pod to manage the NFS cache if we ever wanted it to. But in the end it was more satisfying to stop interacting with the CPS service.

Had to add an invocation of `/opt/cray/dvs/bin/dvs-load.sh` to Nersc's early ansible layer to ensure DVS/LNet kernel modules were loaded; they're typically loaded when the `rootfs` is mounted, but when the `rootfs` is mounted via NFS, they are not. Cray ansible & startup assumes they're there and fails if they're not present.