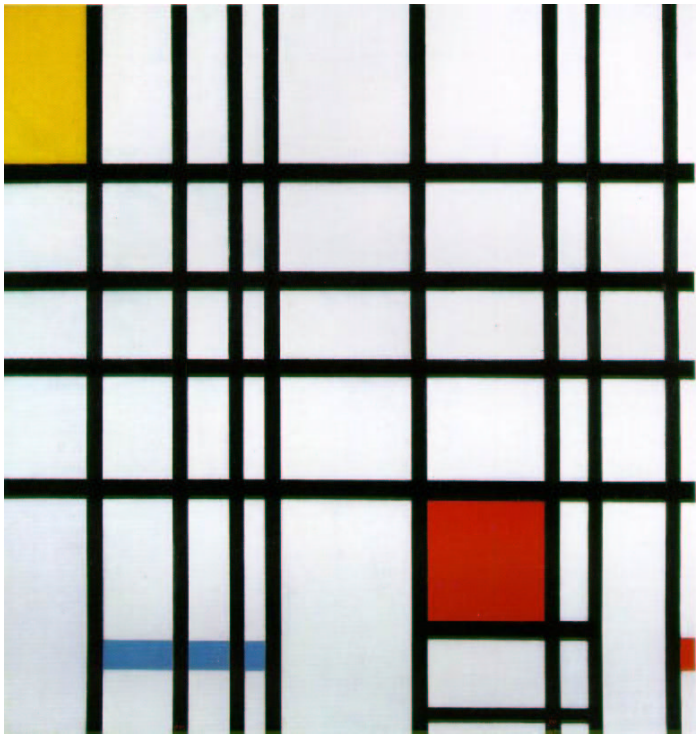
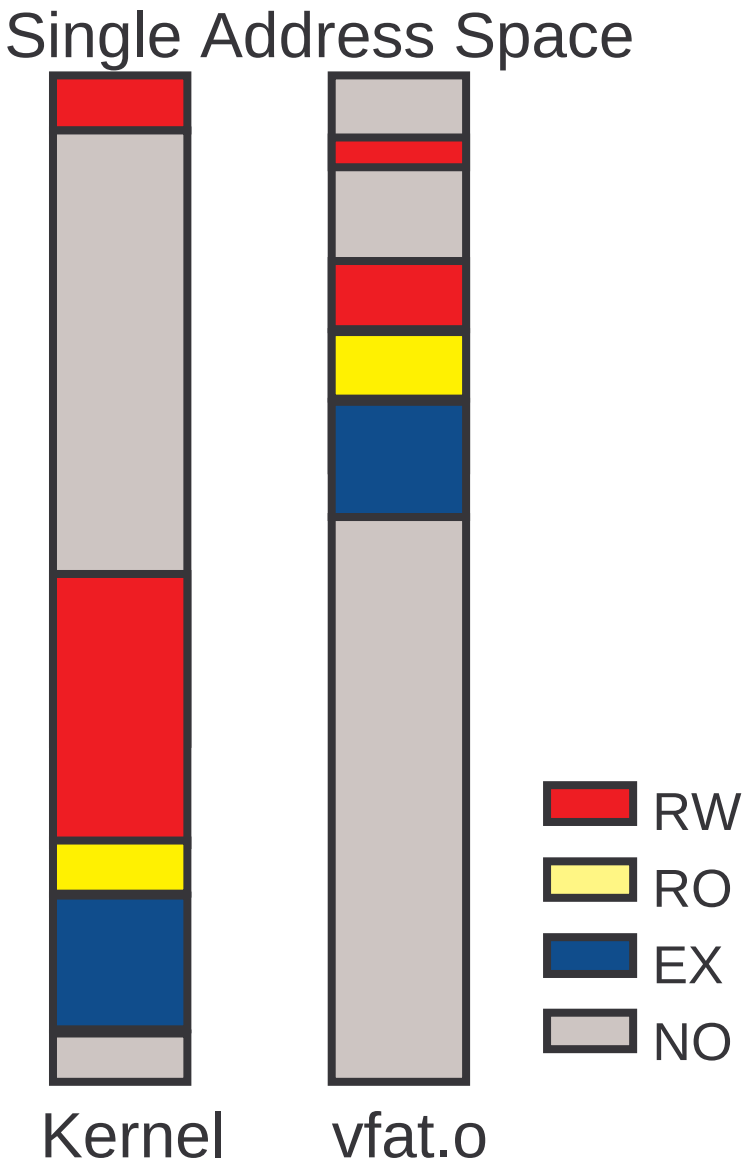
Mondrian Memory Protection



Emmett Witchel
Josh Cates
Krste Asanović
MIT Lab for Computer Science

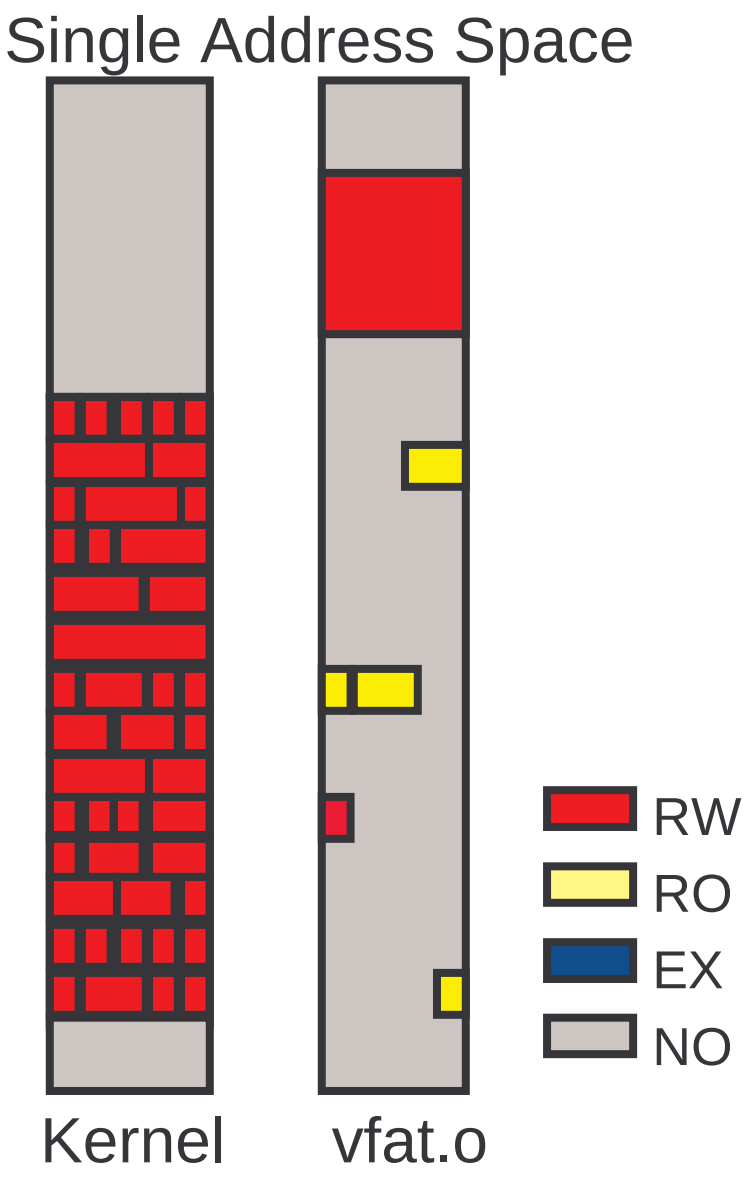
Software Has Needs

- Plug-ins have won as the extensible system model.
  - Fast & data sharing is convenient.
- Software is written for a model not directly supported by current hardware and OSes.
  - No protection.



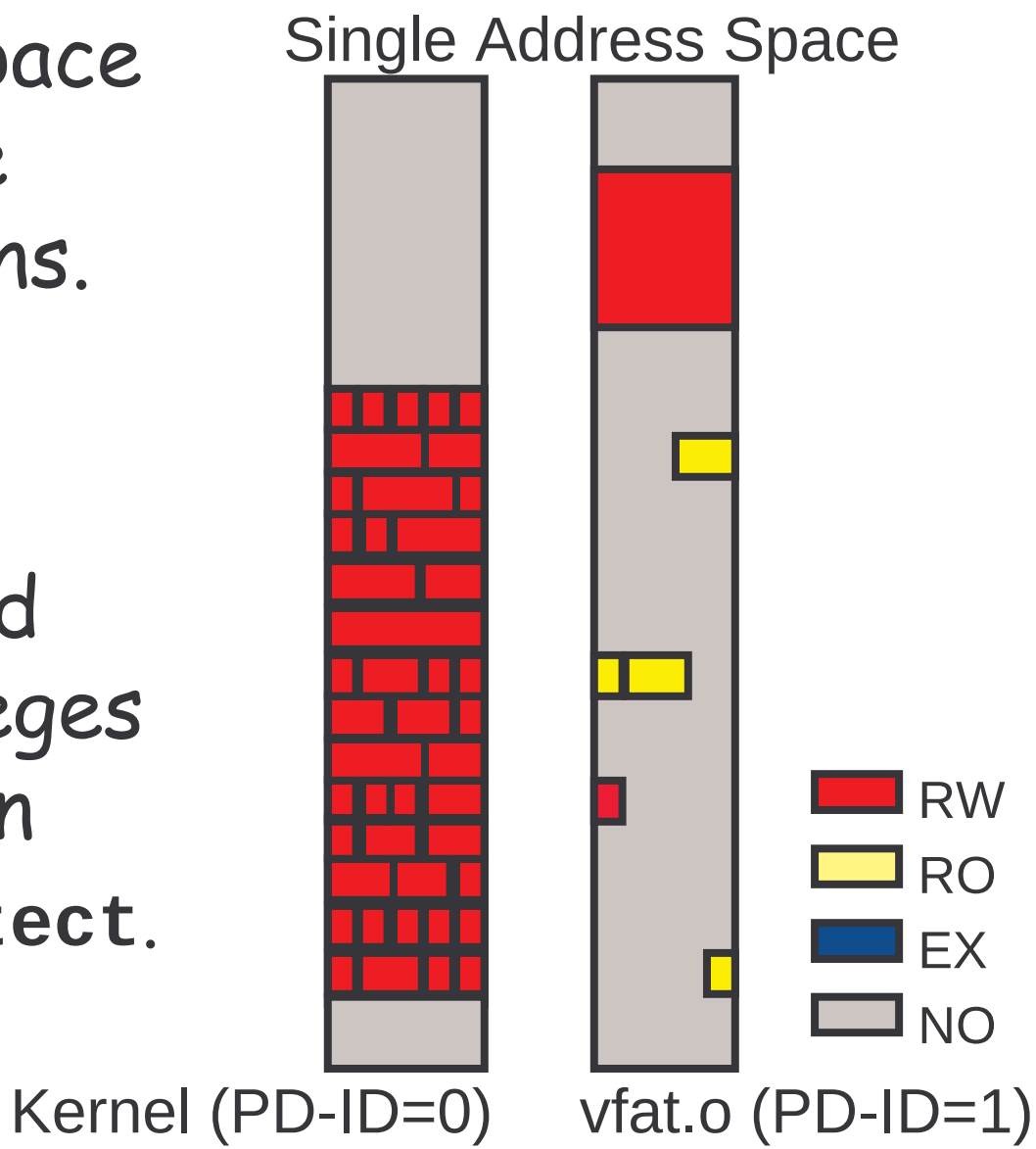
Currently, Protection Is Not Provided

- Plug-ins need access to different, small data structures.
  - Word level protection at word boundaries.
- Placing every possibly shared data on its own page is very difficult.
  - Some data structures imposed by hardware.



Mondrian Memory Protection

- Single address space split into multiple protection domains.
- A domain owns a region of the address space and can export privileges to another domain
  - Similar to **`mprotect`**.



Word Level Protection Is Not New

- Segmentation is a traditional solution.
  - + Provides word-level protection.
  - - Explicit segment registers [B5000,x86]
  - - Non-linear addressing
- Capability based machines.
  - + Fine-grained sharing.
  - - Revocation difficult [System/38, M-machine].
  - - Different protection for different domains via shared capability is hard.

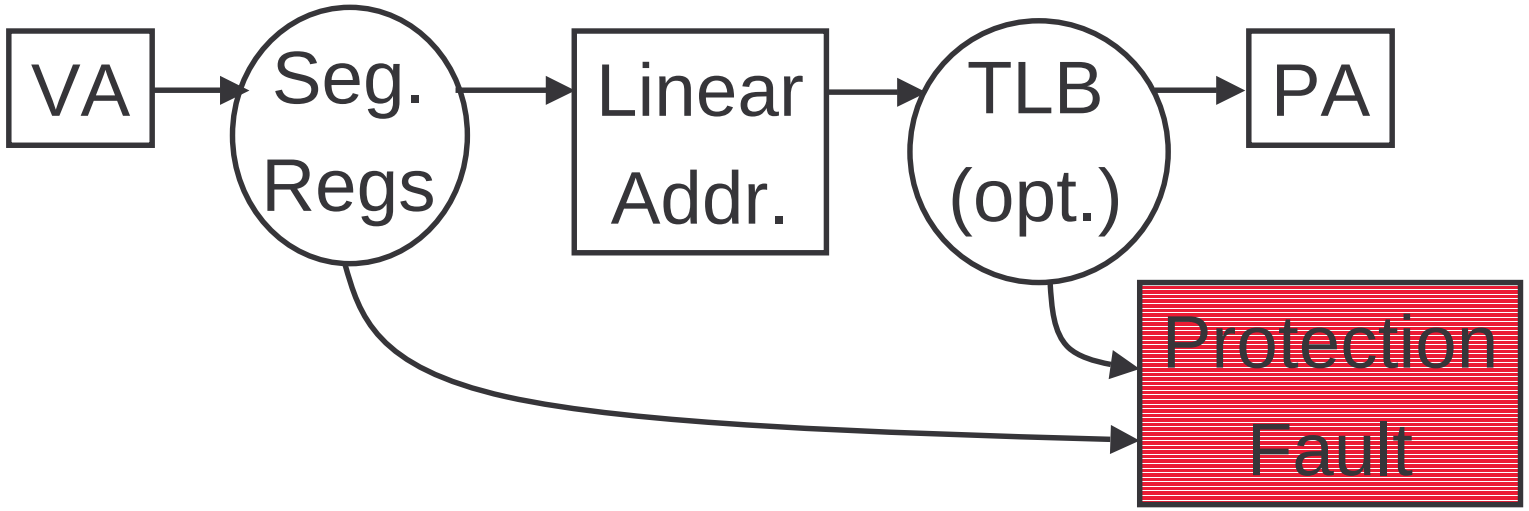
MMP is a New Solution

- Segmentation semantics without the problems.
  - MMP provides fine-grained protection and data sharing.
  - MMP uses linear addressing.
  - MMP is compatible with existing ISAs
  - MMP has no segment registers.
  - MMP has easy perm. revocation.
  - MMP does not have tagged pointers.
- MMP is all the fun of segmentation without the headaches.

There's No Free Lunch

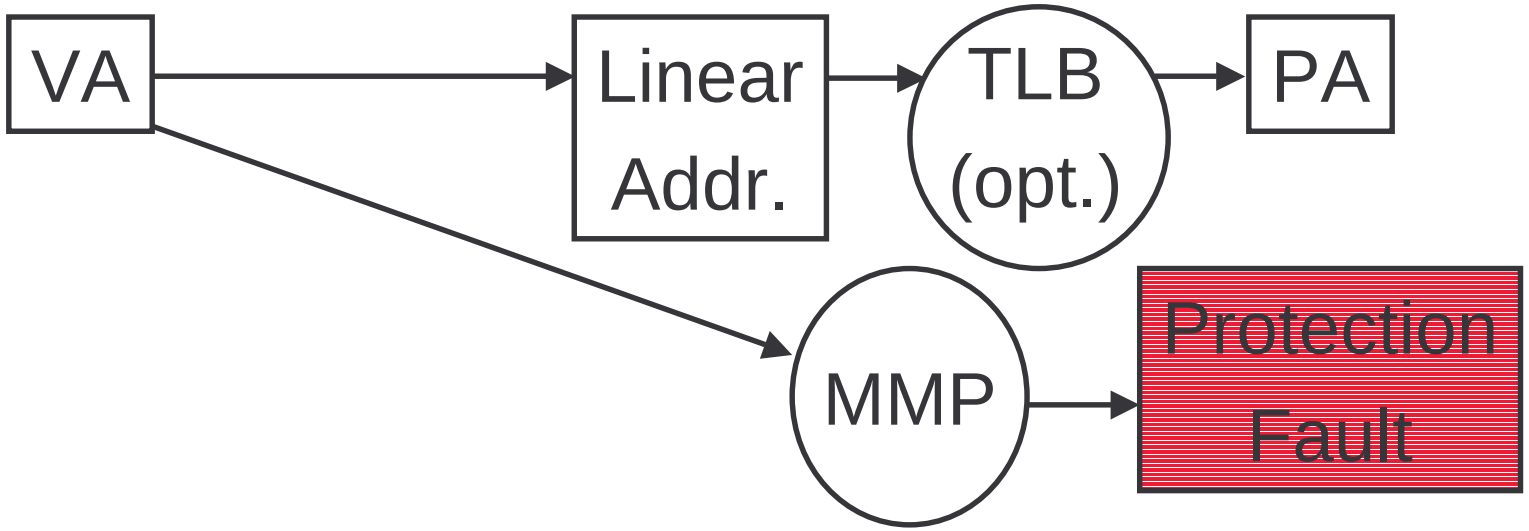
- MMP requires extra memory to store permissions tables.
  - Good engineering keeps tables small.
- MMP requires CPU & memory system resources to access tables.
  - Good engineering provides an effective cache for permissions information so table access is infrequent.

Segmentation Timeline



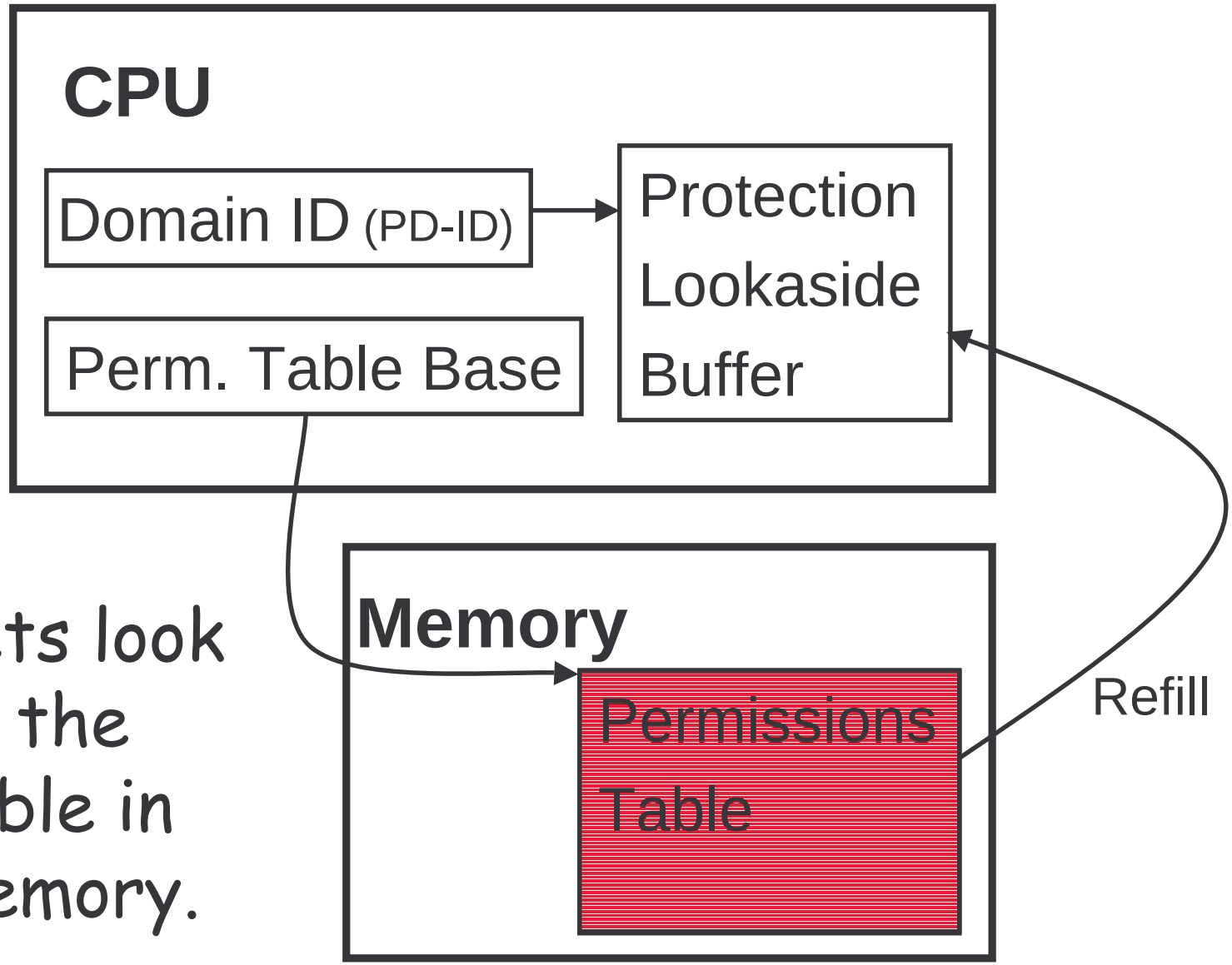
- VA - constructed by processor.
- LA - post segmentation.
- PA - post TLB translation.

MMP Timeline



- MMP checks virtual addresses.
  - Protection check only needs to happen before instruction graduation (not in critical path).

MMP Implementation —Tables



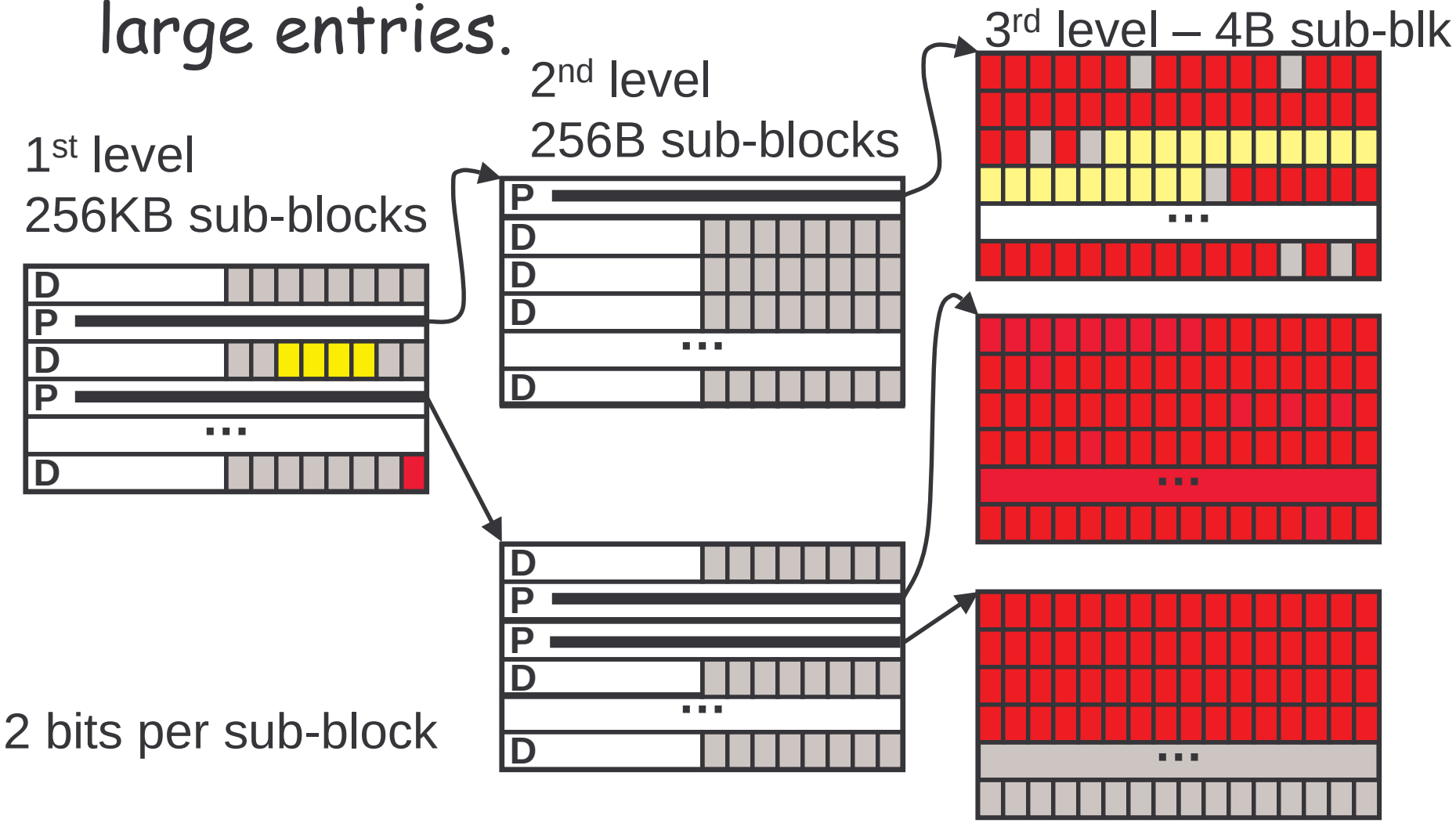
- Lets look at the table in memory.

Permission Table Requirements

- Entries should be compact.
  - 2 bits of permissions data per word (none, read-only, read-write, execute-read).
- Should represent different sized regions efficiently.
  - Any number of words at a word boundary.
- Organized like a hierarchical page table (trie).

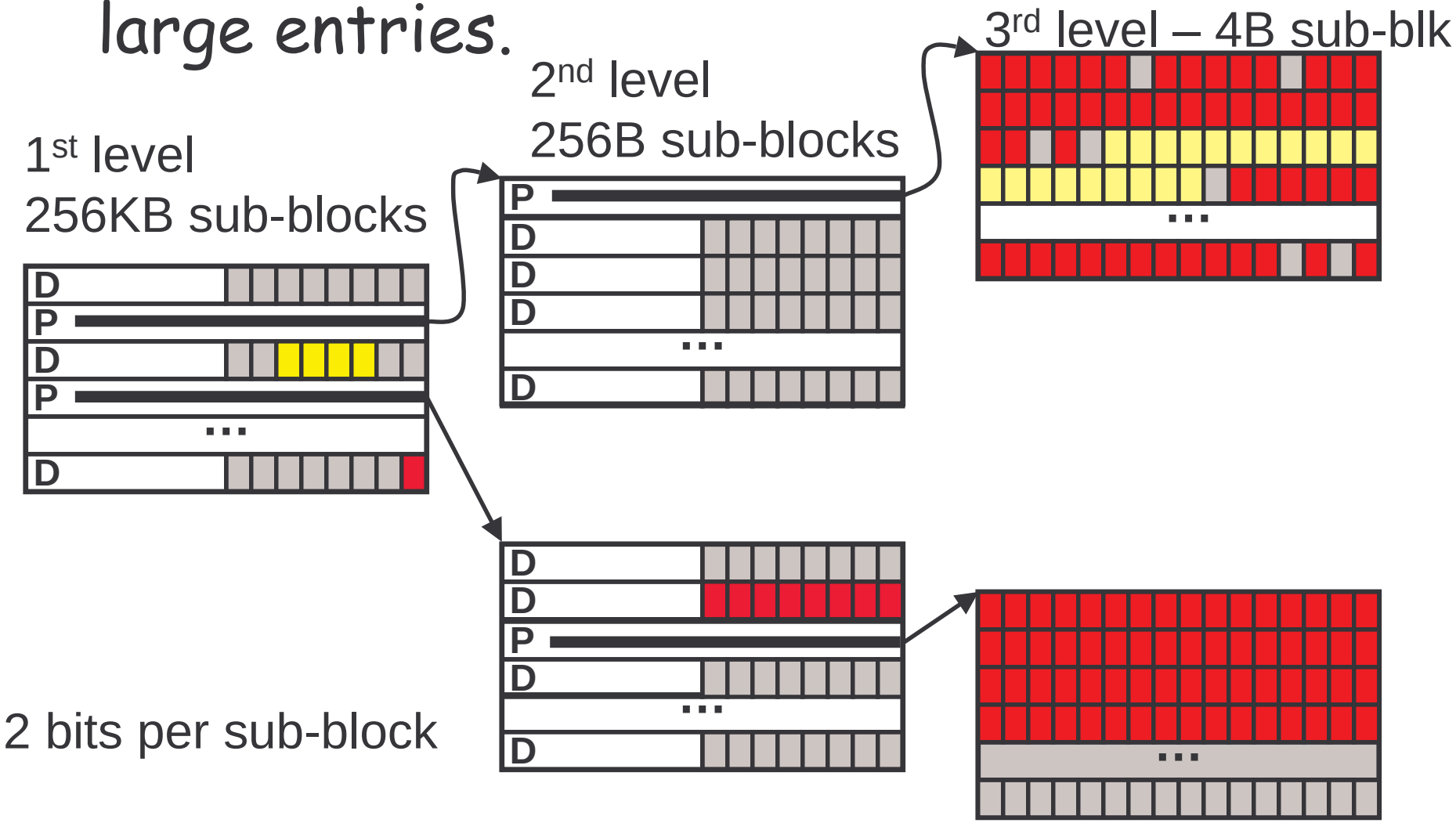
Representing Large Regions Efficiently

- Upper level entries are typed, enabling large entries.



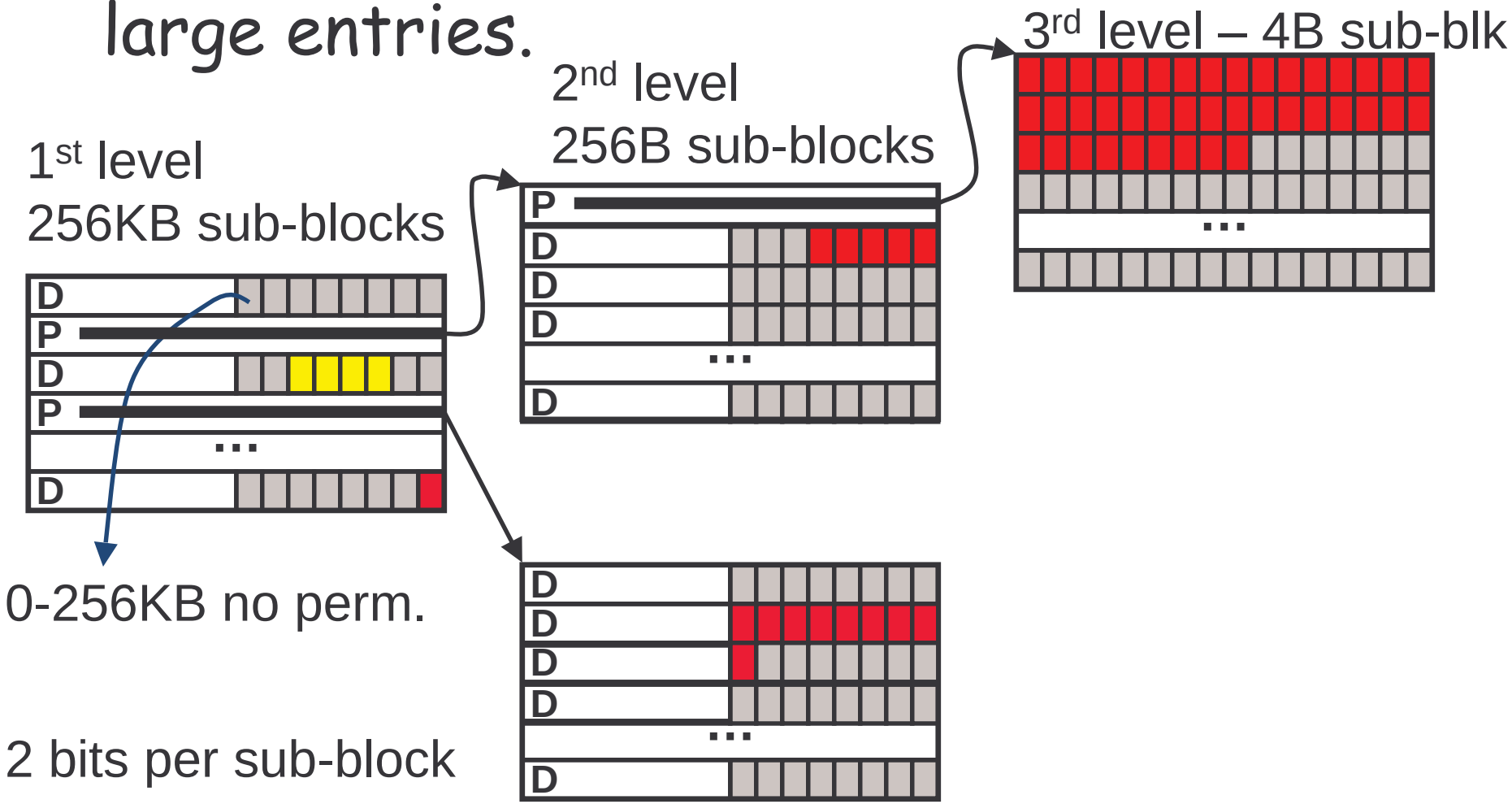
Representing Large Regions Efficiently

- Upper level entries are typed, enabling large entries.



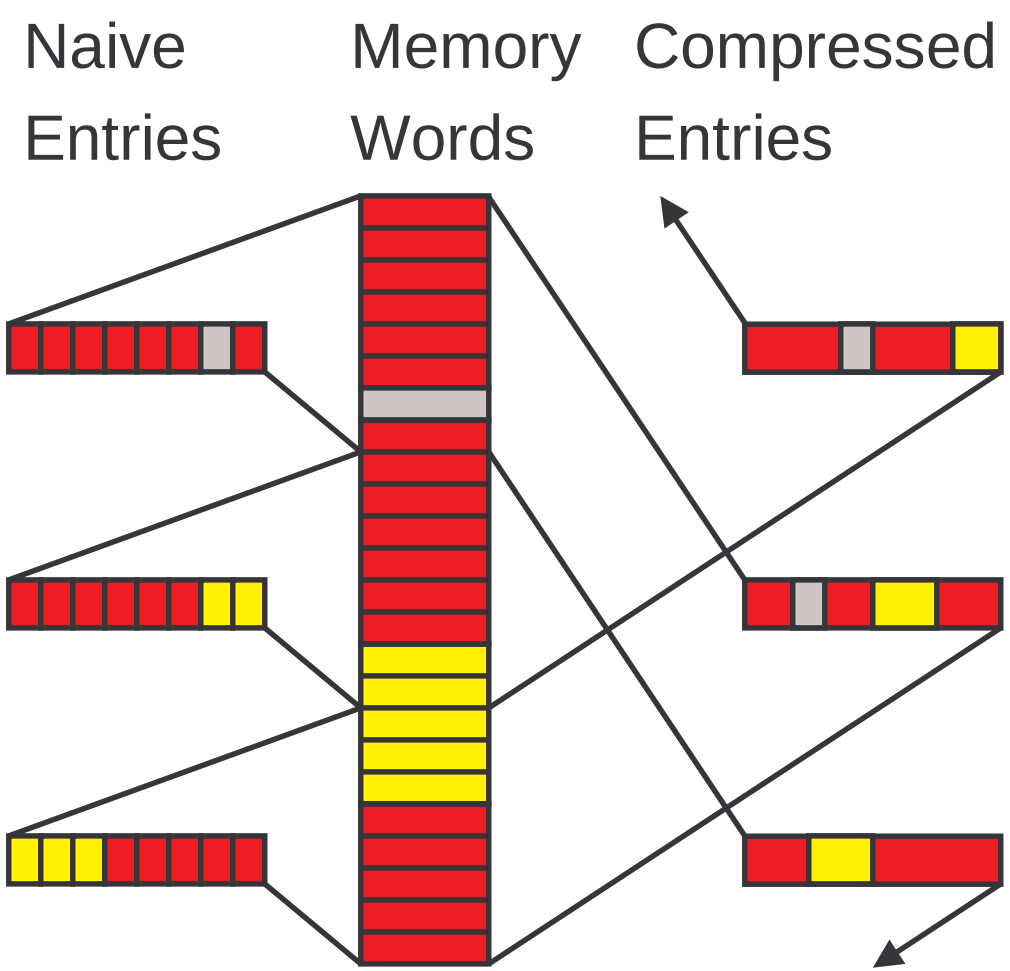
Representing Large Regions Efficiently

- Upper level entries are typed, enabling large entries.

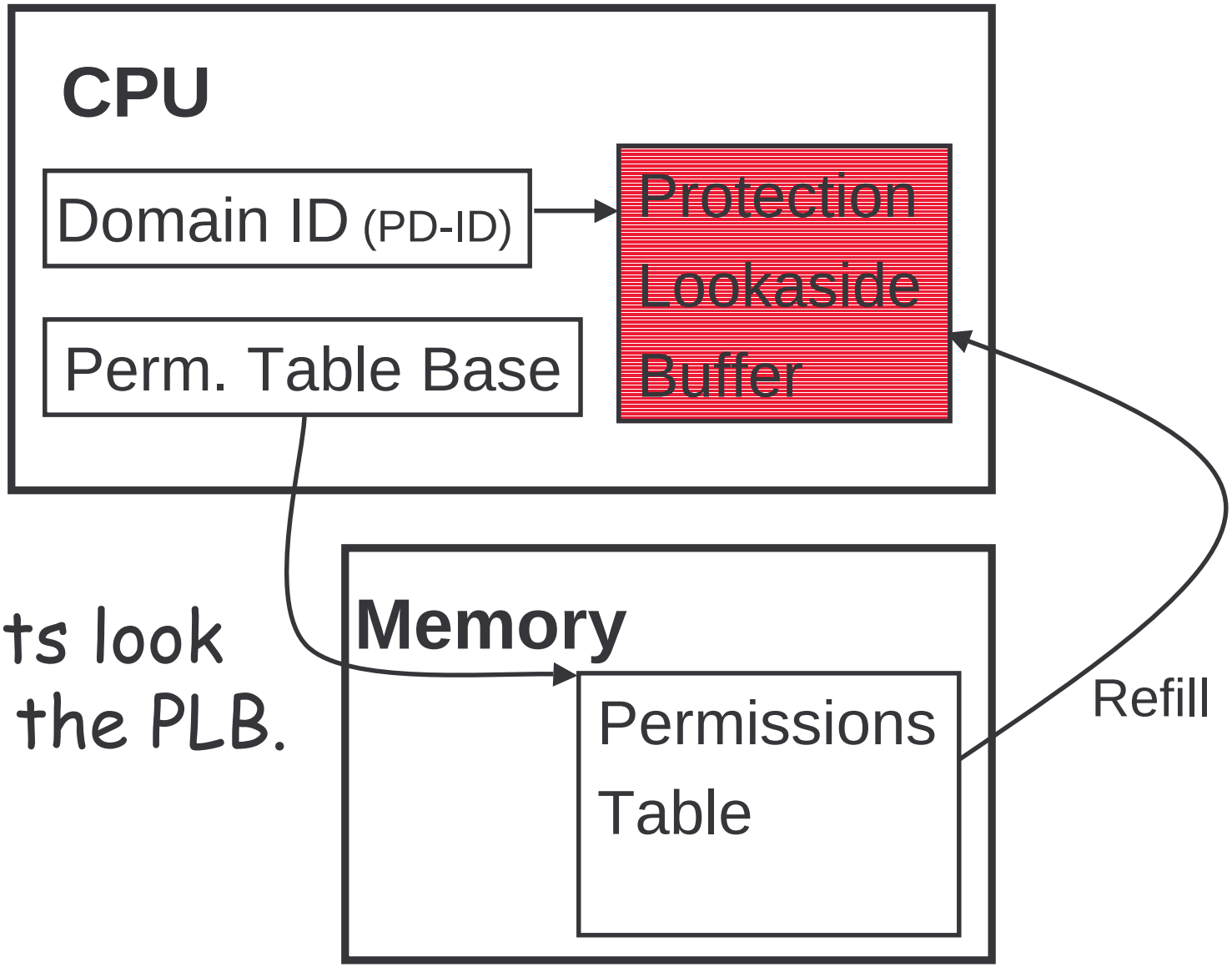


Compressing The Entry Format

- Most words have same perm. as neighbor.
  - Compressed entries represent longer, overlapping regions.
  - Compressed entries are the same size, but represent more information.



MMP Implementation — PLB

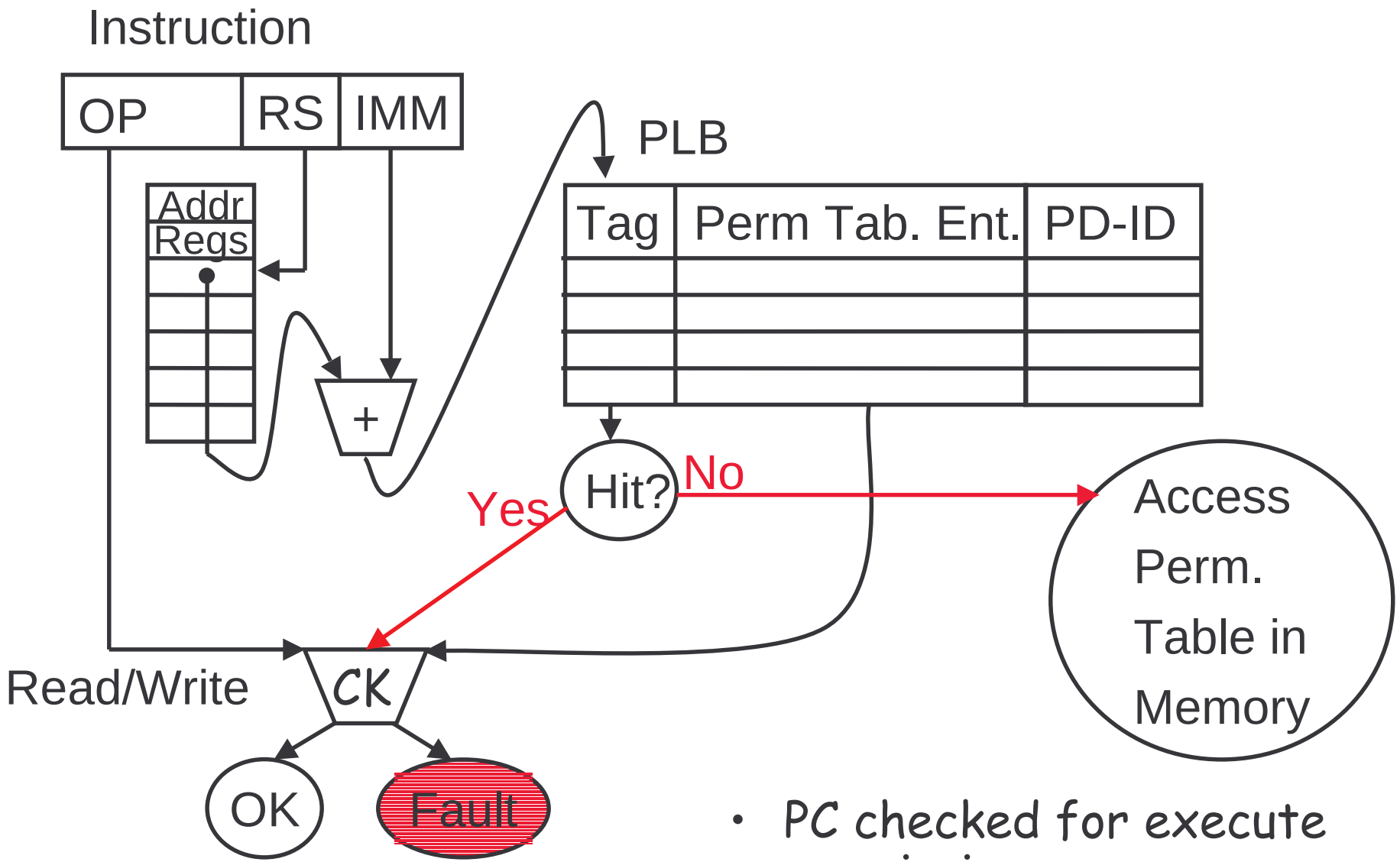


- Lets look at the PLB.

PLB Requirements

- The PLB caches protection table entries tagged by Domain-ID.
  - Like a TLB but without translation.
  - Like a TLB but variable ranges, not just page sizes.

PLB Permissions Check Flow



- PC checked for execute permissions.

PLB Requirements

- PLB task—index permissions data from different sized memory chunks.
  - Loads from different addresses can get permissions information from different levels in the table.

1st level or 2nd level

Protection Look aside Buffer (PLB)

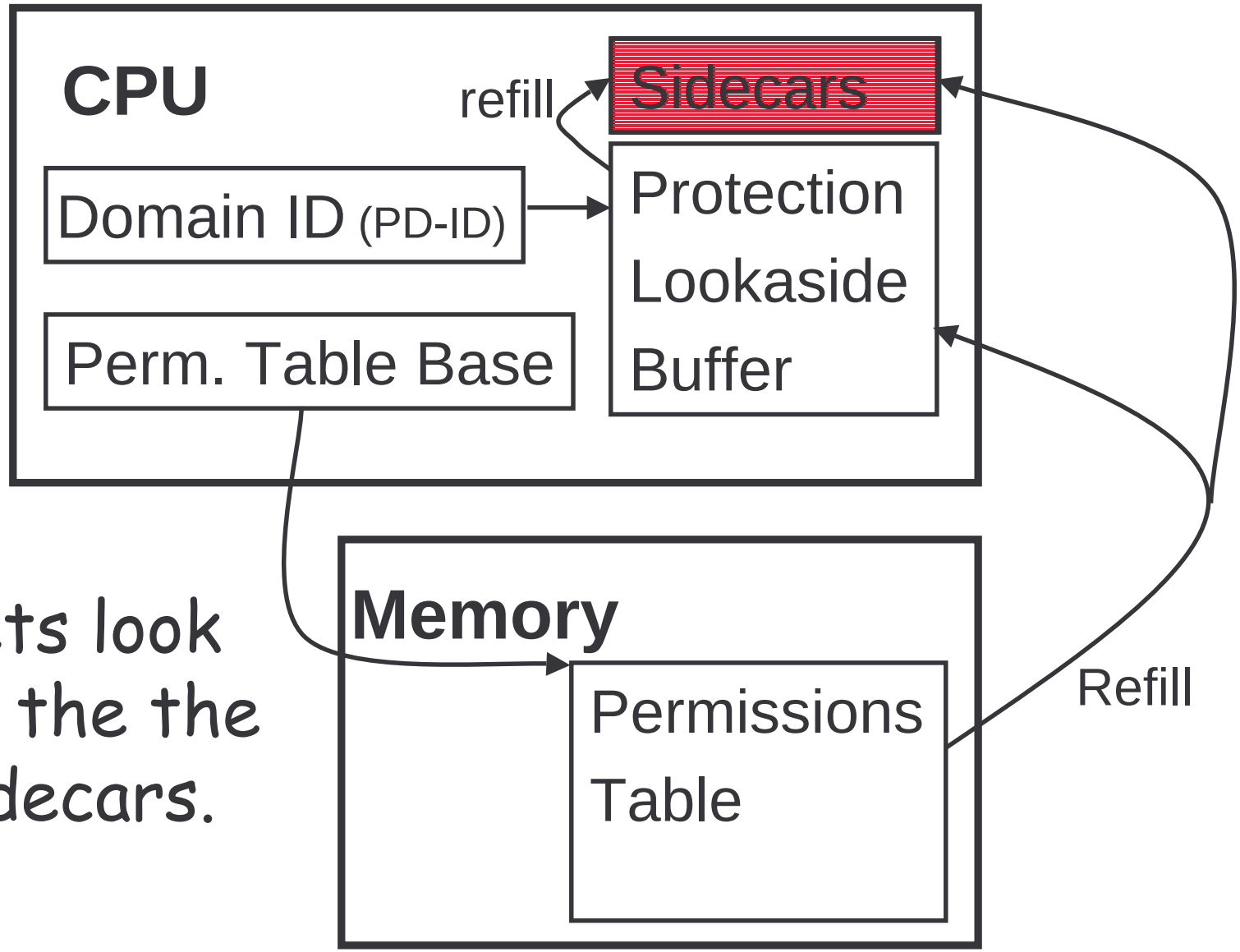
- PLB index implemented by ternary CAM.
  - Like superpages in a TLB, but protection superpages are easy for OS—they don't require lots of contiguous physical memory.
  - PLB index limited to power-of-two size.

PLB (Xs are don't-care bits)

	Tag (26 bits)	Perm. Table Ent.	PD-ID
1st level ent.	0x07 XX XX	D	0
2nd level ent.	0x09 87 XX	D	0
3rd level ent.	0x09 20 58		0

  - The compressed format has intermediate number of don't-care bits, and non power-of-two sized regions.

MMP Implementation — Sidecars

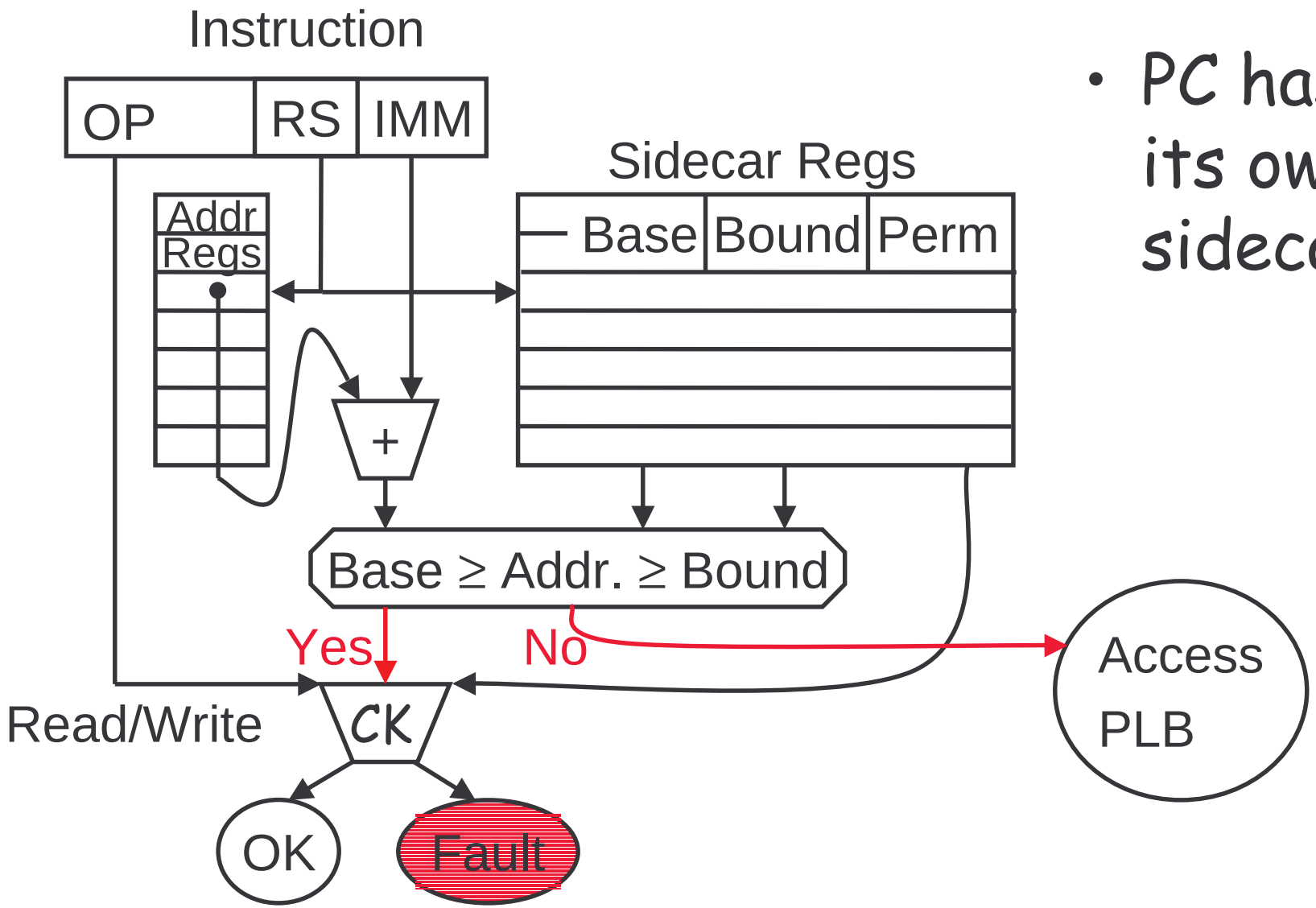


- Lets look at the the sidecars.

Register Sidecars

- Sidecars allow permissions checks without accessing the PLB (register level cache).
  - Base, bounds and permissions information in sidecar.
  - Lower access energy for sidecar than PLB.
- Increased hit rate with compressed entry format because non power-of-two sized regions are not fully indexed by PLB.
  - Fewer table accesses than PLB alone.

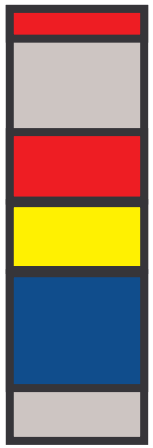
Sidecar Permissions Check Flow



- PC has its own sidecar.

Coarse-Grained Evaluation

- Coarse-grained protection equivalent to current UNIX semantics (text, ro-data, data, bss, stack).
  - One protection domain.



- Application mix from SPEC2000, SPEC95, Java, Media bench, and Olden.
  - Compiled with **`gcc -O3 (egcs-1.0.3)`**
  - Address traces fed to MMP simulator.

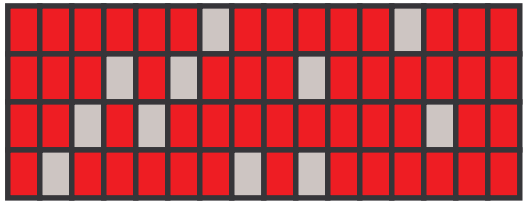
Coarse-Grained Protection Results

	60 Entry PLB	60 Entry TLB
Ref. to MMP tables / Application refs	**0.00-0.56%**	**0.00-2.59%**
Table size / App. data	**0.04-0.62%**	**0.02-0.22%**
Sidecar miss rate	**1-40%(12%)**	**--**

- Comparison with TLB is just for scale, a TLB is still useful with MMP.
  - MMP is 2 bits of protection, not 4 bytes of translation + protection.

Fine-Grained Evaluation

- Fine-grained protection: Every malloc-ed region goes in its own protection region with inaccessible header words between regions.
  - malloc library is protected subsystem.



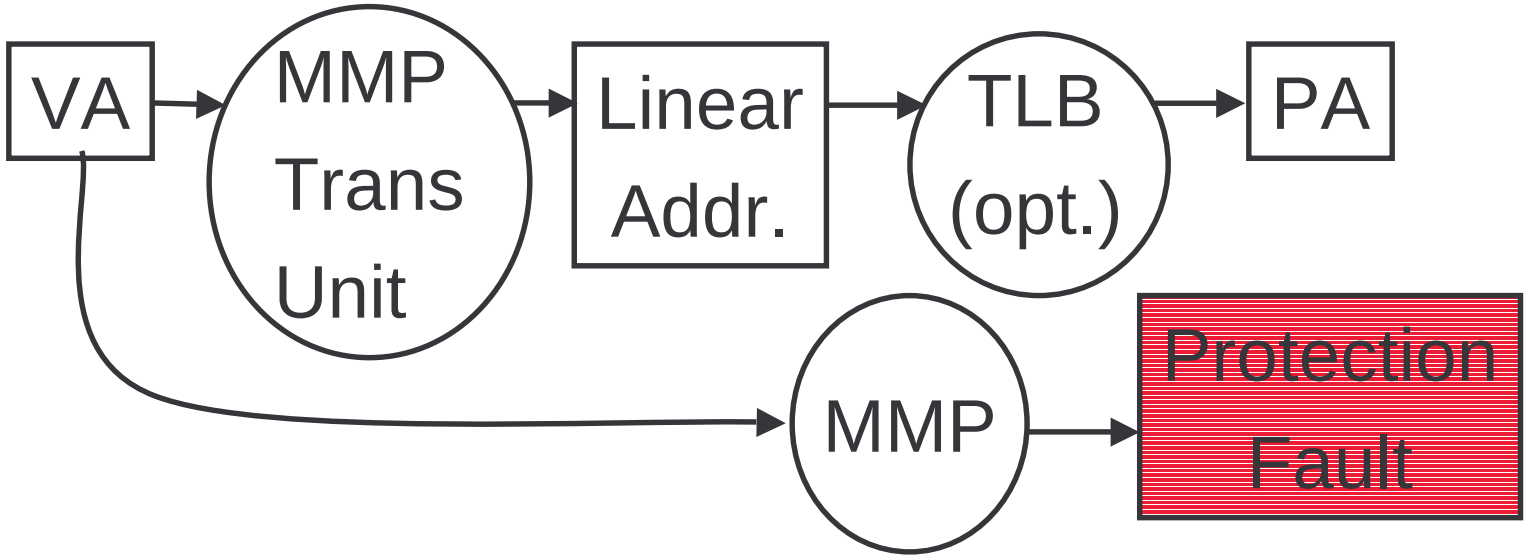
- Very demanding evaluation, almost worst case.
  - Protected subsystems will likely not have to export every region malloc-ed.
  - Functionality similar to purify.

Fine-Grained Protection Results

	60 Entry PLB
$\frac{\text{Ref. to MMP tables}}{\text{Application refs}}$	**0.0- 7.5% (0.1-19%)**
Table size / App. data	**0.4- 8.3%**
Table references eliminated by sidecars	**0.6-11.0%**

- Time and space overheads very small.
  - Results include table updates.
  - Minimal cache disturbance (study in paper).
  - Sidecar helps eliminate table references.
  - Paper compares different entry formats.

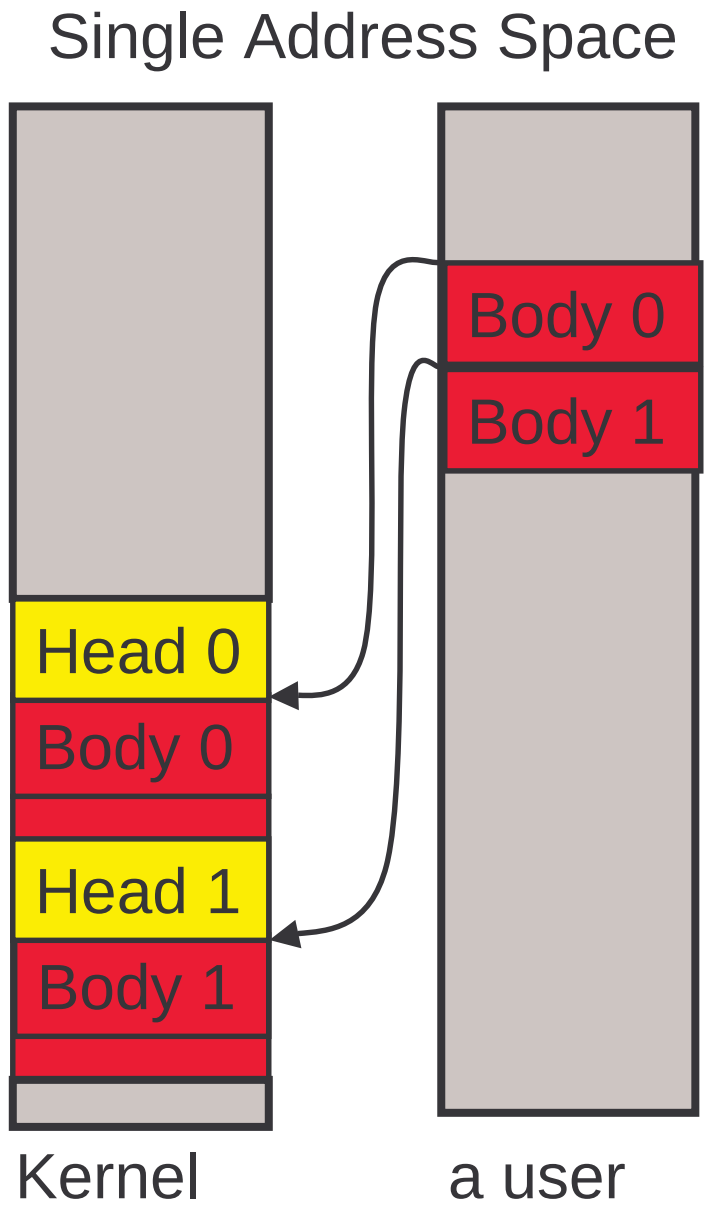
MMP Timeline With Translation



- MMP can add an offset to the VA, providing translation.
  - Protection check happens on pre-translated address.
  - Address generation is 3-to-1 add on critical path.

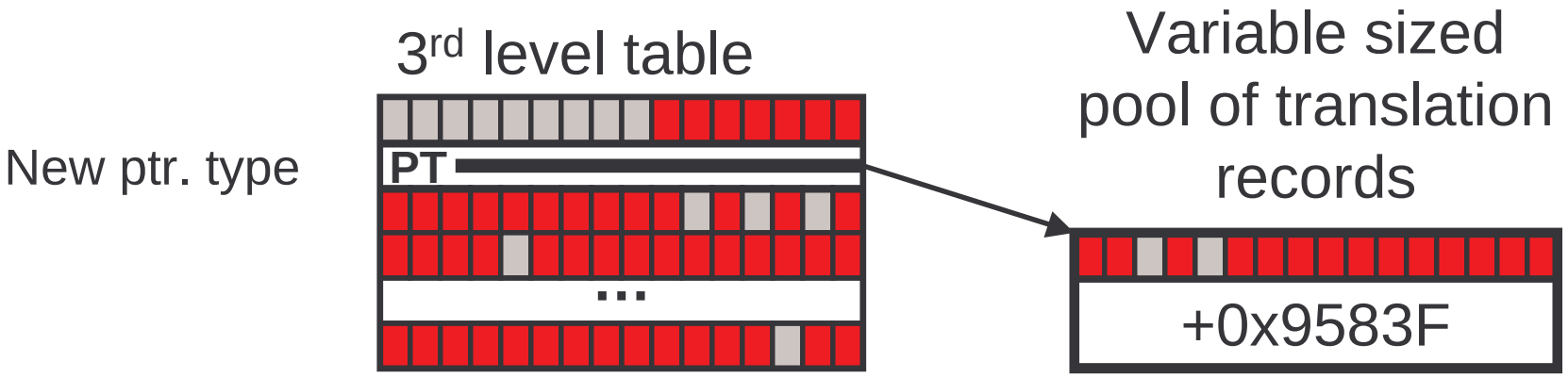
Why Translation?

- Implement zero-copy networking.
- Translation lets memory discontiguous in one domain appear contiguous in another.
- No cache aliasing problem, translation before cache access.



Implementing Translation

- MMP entry format is flexible, allowing additional pointer types.
  - Pointer to permissions and byte-level translation offset.



- Translation information held in sidecar.

MMP Networking Results

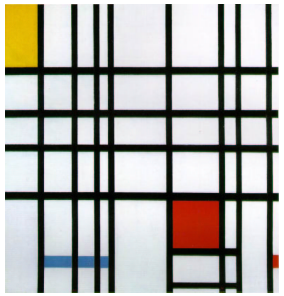
- Simulated a zero-copy networking implementation that uses unmodified **read** system call.
  - Web client receiving 500KB.
- Eliminates 52% of memory references relative to a copying implementation.
  - Win includes references to update and read the permissions tables.
  - 46% of reference time saved.

Related Work

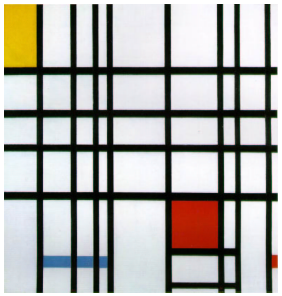
- Capabilities [Dennis65, IBM AS400].
- Domain Pages [Koldinger ASPLOS92].
- Guarded pointers [Carter ASPLOS94].
- Guarded page tables [Liedke 94].
- IP longest prefix match [Waldvogel TOCS 01].

Possible Applications

- Safe kernel modules.
  - Safe plug-ins for apache and web browsers.
- Eliminate memory copying from kernel calls.
  - Provide specialized kernel entry points.
- Support millions of threads, each with a tiny stack.
- Implement C++ const.
- Use meta-data for cache coherence.
- Make each function its own protection domain.
  - Buffer overrun much more difficult.



Conclusion



- Fine-grained protection is the solution for safe, extensible systems.
- Fine-grained protection can be provided efficiently.
- Mondrian Memory Protection will enable more robust software.
  - It matches the way we think about code.
  - It can be adopted incrementally (e.g., 1st just change malloc library).