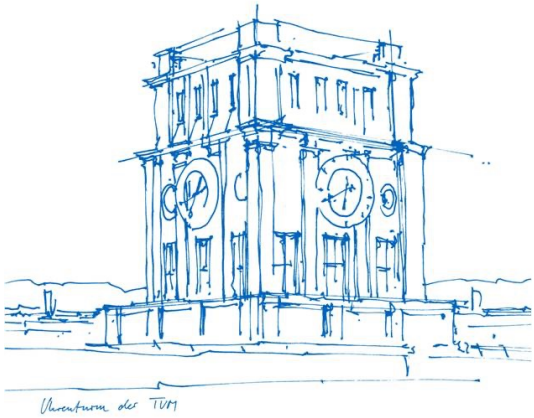
Prefetch & Security Units for In-Vehicle Memory Hierarchies

Yuanji Ye[1], Oliver Lenke[1], Jens Nöpel[2],
Thomas Wild[1], Georg Sigl[2], Andreas Herkersdorf[1]

[1]Chair of Integrated Systems
[2]Chair of Security in Information Technology
Technical University of Munich, Germany

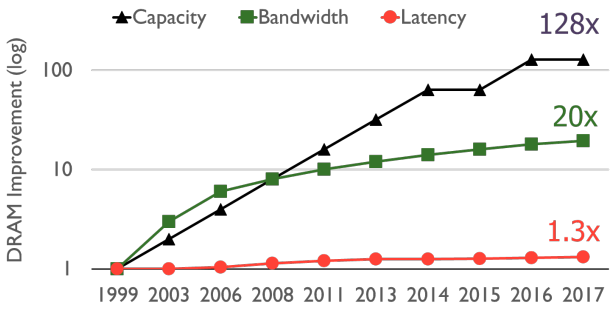
MPSoC Forum, Victoria, CA
June 22, 2026



1

Data Locality Driven MPSoC Design Paradigm

- DRAM access latency has barely improved over past 25 years
  - Processors facing the "memory wall"
- Prefetching: Hide access latency by fetching data before actual CPU demand

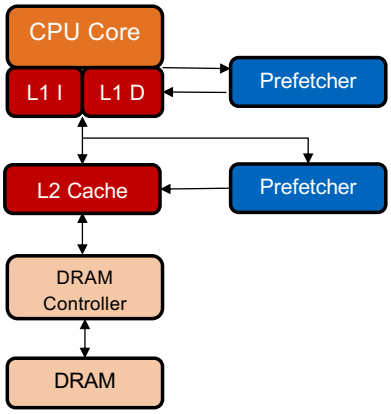


2

2

CPU-Side Prefetcher

- Located close to CPU caches (L1/L2/LLC)
- Access to PC, fine-grained memory access pattern
- Prefetch data to caches

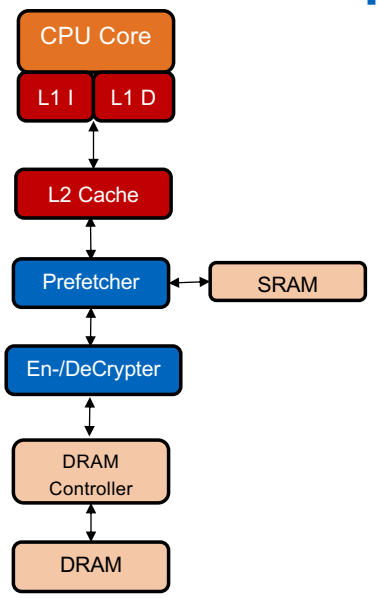


3

3

Memory-Side Prefetcher

- Located close to DRAM Controller
- Monitor aggregated memory requests from all cores
- Only sees LLC misses
- Prefetch data to a separate SRAM buffer

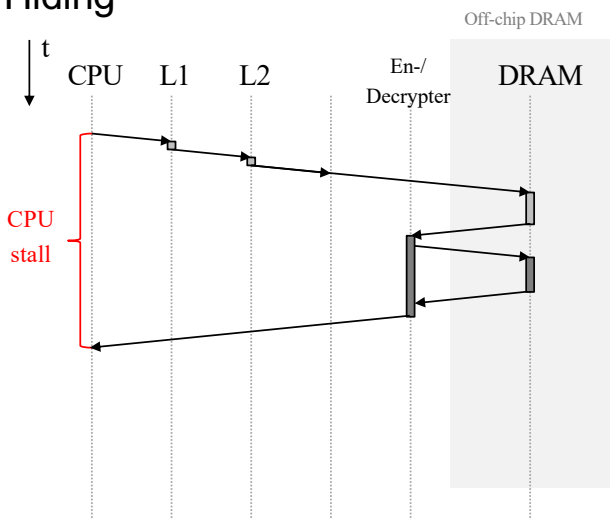


4

4

DRAM Access & Crypto Latency Hiding

- CPU Ld/St instructions involve multiple cumulative latencies:
  - L1 and L2 cache lookups
  - Off-chip memory (DRAM) access
  - Optional: Data en-/decryption
- En-/Decryption → additional metadata (tags, counters, integrity trees) fetches from DRAM

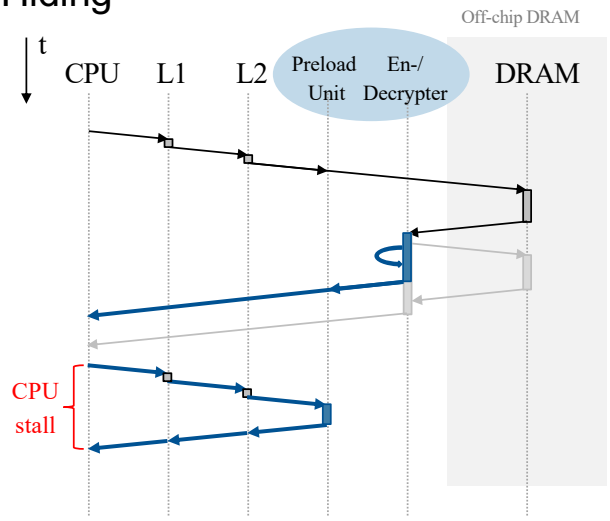


5

5

DRAM Access & Crypto Latency Hiding

- CPU Ld/St instructions involve multiple cumulative latencies:
  - L1 and L2 cache lookups
  - Off-chip memory (DRAM) access
  - Optional: Data en-/decryption
- En-/Decryption → additional metadata (tags, counters, integrity trees) fetches from DRAM
- Our proposed solution combines
  - On-the-fly streaming crypto methods that reuse metadata to reduce access overheads
  - Latency hiding prefetch strategies



6

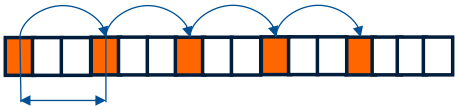
6

Prefetch Mechanisms

Streaming Prefetcher

Assume back to back memory accesses

Stride Prefetcher:



Detects constant-distance k between consecutive memory accesses for each PC

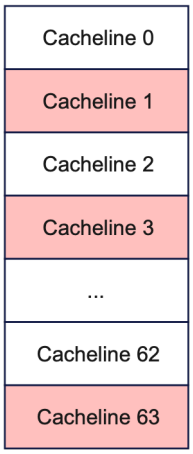
Footprint Prefetcher

Records historical cache line access sequences within a page and replays them when trigger conditions are met

7

Spatial Footprint Prefetcher

- The memory footprint of a region is represented as a bit vector
- Store the footprint vector in a pattern history table
- Event Keys: Footprint lookups are triggered by keys such as the program counter (PC), address or the page offset

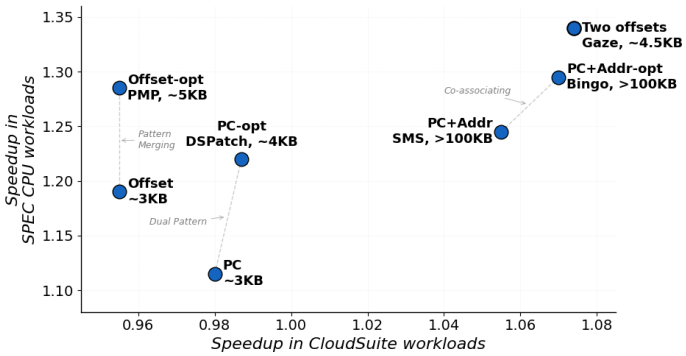


Page Footprint Vector:
0101…01

Footprint Pattern Example of a 4KB Page with 64-byte Cachelines

8

Related Work: L1/L2 Prefetch Mechanisms



PMP[2]
- Use a simple event key

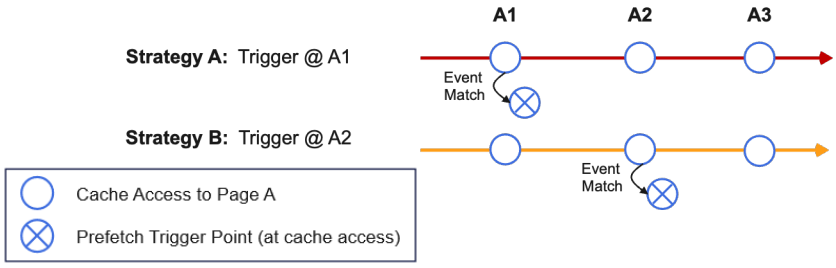
Bingo[3]
- Combine specific key and simple key
- Simple key as the fallback to the specific key

Gaze[4]
- Uses a chain of simple keys (page offset)
- Improves confidence by waiting for more context

9

9

Related Work: Fixed Trigger Time Point

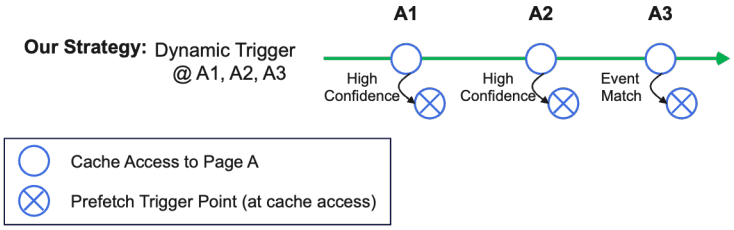


- Existing mechanisms optimize the prediction at a predefined trigger point
- Limitation of fixed trigger timing:
  - Early Trigger: good timeliness, but little context
  - Late Trigger: more context, but lost early opportunities

10

10

STEP: Multi-Point Prefetching Triggers



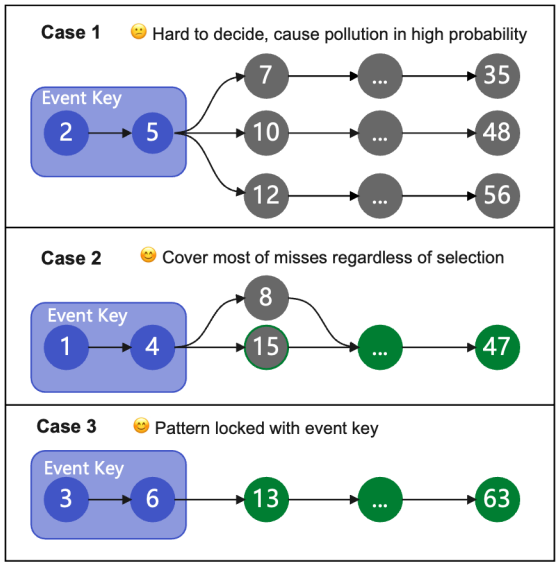
- Move from a "one-shot" trigger to a sequence of decisions.
- Combine the advantages of early trigger and late trigger together
- Use a **confidence evaluator** to decide which time point is suitable for prefetching

STEP makes the prefetch trigger a multi-stage runtime decision.

11

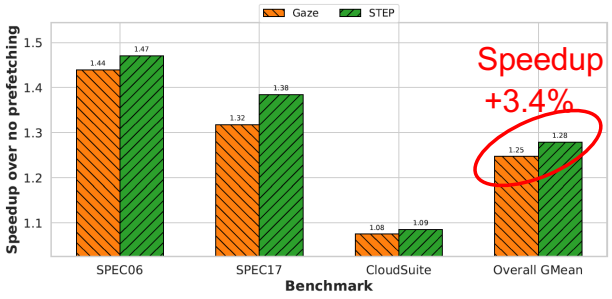
STEP: Confidence Evaluation

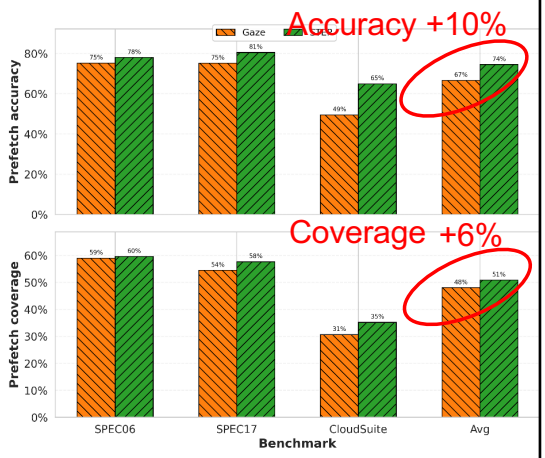
- Event Key @ A2: two-offset chain
- Three cases:
  - Case 1: High Footprint Divergence, hard to decide the correct pattern
  - Case 2: Low Footprint Divergence, prefetch the common part
  - Case 3: Pattern locked with event key, prefetch the whole footprint pattern
- Prefetch in Cases 2 and 3;
  Wait for more evidence in Case 1



12

Single Core Results at L2 Cache Level



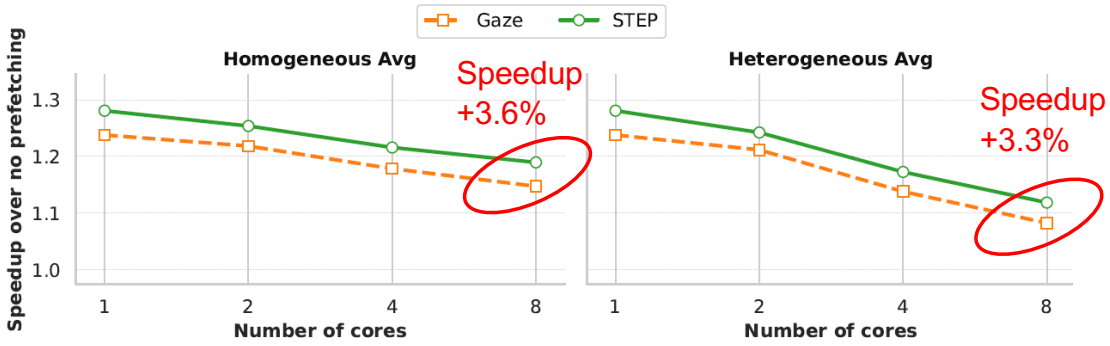


- Outperforms baseline Gaze by 3.4% in performance
- Both accuracy and coverage are improved

13

13

Multicore Results



- Stable improvement in both homogeneous and heterogeneous multicore workloads
- Want more details? See ISCA'26 proceedings (06/27 – 07/01, Raleigh, NC)

14

14

Take Aways

- STEP revisited trigger timing in spatial CPU-side footprint prefetching and move the decision from a fixed design choice to a multi-stage runtime decision
- STEP is a lightweight instantiation based on the Gaze prefetcher and outperforms it in our evaluated settings
- Memory-side cache prefetchers can meaningfully be combined with near-memory compute accelerators
  - E.g., on-the-fly data en-/decryption

15

15

Thank you & Questions!

Supported by German Federal Ministry of Research, Technology and Space under the funding indicator: 16ME0800K.
MANNHEIM CeCaS.

Bundesministerium für Forschung, Technologie und Raumfahrt

16

16