ECE 585Simulated Cache Test PlanTeam 4Portland State UniversityMaseeh College of Engineering & Computer ScienceDepartment of Electrical & Computer Engin eeri ngAuthors:Kane-Pa rdy, ChrisMelli, AmeerMonti, CalebNovemb er 19, 2024ECE 585Microprocessor System Design

Simulated Cache Test Plan (Team 4) ECE 585Contents1 Debug Mo de (Verb ose) 22 Normal Mo de Functionality 63 Silent Mo de Functionality 104 Asso ciativity Verication 145 Alignment Verication 196 Pseudo-LRU Replacement Policy Verication 237 Write-Allo cate Validation 288 MESI State Accuracy 339 Cache Coherence 4310 Inclusivity Verication 5511 Write-Once Policy 6012 Cache Statistics Rep orting 661

Simulated Cache Test Plan (Team 4) ECE 5851 Debug Mo de (Verb ose)Ob jectives‹Ensure Full Verb ose Output with Internal State Tracking:Verify thatwhen Debug Mo de (Verb ose) is enabled, the simulation outputs detailed in-formation on every internal op eration, including cache actions, bus op erations,sno op resp onses, internal state transitions, decisions (like evictions and misses),and any `fprintf` debug messages used throughout the co de.‹Verify Mo de Switching Works Correctly Without Recompilation:En-sure that the simulation can easily switch b etween mo des (Normal and Debug)via command-line arguments, with Debug Mo de providing the most granularinformation, including verb ose cache op eration logs and any `fprintf` output,and without needing to recompile the system.2

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Ensure Full Verb ose Output and Track Internal State Tran-sitions1.1 Run a Trace File in Debug Mo de (Verb ose)Pro cedure:1. Run a trace le that includes cache read and write op erations, enabling DebugMo de (Verb ose).2. Verify that the output includes:‹Detailed logs of every cache read/write op eration, including address, ac-tion, and cache state changes.‹All bus op erations (e.g., READ, WRITE, INVALIDATE) with relevantaddress and data (if applicable).‹Sno op results (e.g., HIT, HITM) and how the system handles these sno ops.‹Intermediate state transitions, such as evictions, misses, and replacementsduring cache op erations.‹Any `fprintf` debug messages inserted in the co de (e.g., debug prints show-ing variable states, function calls, or cache-related actions).‹Information ab out the application of the replacement p olicy (e.g., pseudo-LRU), showing how cache access patterns inuence eviction and replace-ment decisions.‹A nal summary of statistics after the trace execution, including:{Total cache reads{Total cache writes{Cache hits{Cache misses{Cache hit ratio3

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Outcome:The verb ose output should show comprehensive logs for every op eration, cache statechange, bus interaction, internal decision-making p oint (e.g., eviction or miss deci-sions), and all debug messages from `fprintf` statements. At the end of the trace,the output should include the accurate nal summary statistics (reads, writes, hits,misses, hit ratio).2. Verify Mo de Switching Works Correctly Without Recom-pilation2.1 Toggle Between Normal and Debug Mo de (Verb ose) Using Command-Line ArgumentsPro cedure:1. Run a trace le in Normal Mo de and verify the output includes only basic logs(e.g., cache read/write, hits/misses, bus op erations).2. Switch to Debug Mo de (Verb ose) via command-line arguments, without re-compiling the system.3. Re-run the same trace and verify that the output now includes detailed logsfor cache op erations, bus activity, sno op resp onses, and `fprintf` debug output.4. Toggle back to Normal Mo de and verify that the output returns to basic logswithout `fprintf` debug mess ages.Exp ected Outcome:The mo de can b e toggled b etween Normal and Debug Mo de (Verb ose) seamlessly viacommand-line arguments, with Debug Mo de displaying signicantly more detailedoutput, including cache op erations, state transitions, bus interactions, and `fprintf`debug messages.4

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Verb ose Output with State Tracking:Debug Mo de should provide fulland detailed logs for every op eration, cache transaction, and sno op result,including internal state transitions, decision-making (e.g., evictions, misses,replacements), and any `fprintf` debug messages used in the co de. Additionally,a nal summary of statistics (reads, writes, hits, misses, hit ratio) should b edisplayed after the trace execution.‹Mo de Switching:The simulation should allow toggling b etween Normal andDebug Mo de (Verb ose) via command-line arguments without recompilation.Debug Mo de should output signicantly more detailed information, including`fprintf` debug output.5

Simulated Cache Test Plan (Team 4) ECE 5852 Normal Mo de FunctionalityOb jectives‹Validate Detaile d Output in Normal Mo de:Ensure that when running inNormal Mo de, the simulation provides b oth real-time logs of cache op erations,bus activities, sno op results, and internal state transitions , as well as a nalsummary of statistics (cache reads, writes, hits, misses, and hit ratio).‹Ensure Mo de Switching Work s Correctly Without Recompilation:Verify that Normal Mo de and Silent Mo de can b e toggled seamlessly viacommand-line arguments, adjusting the output format accordingly, withoutrequiring recompilation of the simulation.6

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Validate Output in Normal Mo de1.1 Execute a Trace File in Normal Mo de and Che ck OutputPro cedure:1. Run a trace le with a mix of cache read/write op erations, sno op requests, andbus transactions, ensuring Normal Mo de is enabled.2. Observe the output during execution and verify it includes:‹Per-op eration logs for cache op erations, such as hits, misses, writes, andreads.‹Detailed bus op erations (e.g., READ, WRITE, INVALIDATE).‹Sno op results (e.g., HIT, HI TM ).‹No information is omitted from the real-time logs during the trace.3. After the trace completes, conrm that the following summary s tatistics aredisplayed:‹Total cache reads‹Total cache writes‹Cache hits‹Cache misses‹Cache hit ratio4. Verify that these statistics are accurate based on the trace op erations.Exp ected Outcome:In Normal Mo de, the simulation should display detailed logs during the trace, includ-ing cache op erations, bus transactions, and sno op results, followed by an accuratenal summary of statistics (reads, writes, hits, misses, hit ratio).7

Simulated Cache Test Plan (Team 4) ECE 5852. Ensure Mo de Switching Works Correctly Without Recom-pilation2.1 Toggle Between Normal and Silent Mo de Using Command-Line Ar-gumentsPro cedure:1. Run a trace le with Normal Mo de enabled and verify that detailed logs andthe nal statistics are shown.2. Without recompiling, switch to Silent Mo de via the command-line argumentsand re-run the same trace.3. Conrm that Silent Mo de only shows the summary statistics at the end, whileNormal Mo de shows b oth detailed logs during execution and the summary atthe end.Exp ected Outcome:Normal and Silent Mo des should b e toggleable via command-line arguments withoutrecompilation. In Normal Mo de, the output should show detailed logs during theexecution and the summary at the end. In Silent Mo de, only the summary s tatisticsshould b e shown.8

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Full Output in Normal Mo de:Normal Mo de displays detailed logs duringthe trace, including cache op erations, bus activities, and s no op results, followedby accurate summary statistics (reads, writes, hits, misses, hit ratio).‹Mo de Switching:Normal and Silent Mo des can b e toggled without recompi-lation. Normal Mo de outputs b oth detailed logs and the nal statistics, whileSilent Mo de shows only the summary statistics.9

Simulated Cache Test Plan (Team 4) ECE 5853 Silent Mo de FunctionalityOb jectives‹Verify Suppression of Detailed Logs in Silent Mo de:Ensure that alldetailed logs, including cache op erations, bus activities, and sno op results, aresuppressed in Silent Mo de so that only summary statistics app ear at the endof the trace.‹Ensure Prop er Display of Cache Usage Summary:Conrm that SilentMo de correctly displays a concise summary of cache us age statistics at theend of each trace, including reads, writes, hits, misses, and hit ratio, withoutdetailed intermediate output.‹Conrm Mo de Toggle Functionality Without Recompilation:Verifythat Silent Mo de can b e toggled on or o without recompilation, ideally viacommand-line arguments, to conrm exible mo de selection for testing anddeployment.10

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Suppression of Detailed Logs and Display of CacheUsage Summary in Silent Mo de1.1 Run a Trace File in Silent Mo de and Check Summary StatisticsPro cedure:1. Execute a trace le in Silent Mo de that includes a variety of op erations (reads,writes, sno op requests).2. Observe the output during execution and conrm that:‹No intermediate logs for individual cache op erations, bus transactions, orsno op results are displayed.‹At the end of the trace, only a concise summary with the following statis-tics is displayed:{Total cache reads{Total cache writes{Cache hits{Cache misses{Cache hit ratioExp ected Outcome:Silent Mo de suppresses all detailed logs and only displays the summary statisticsat the end of the trace. The statistics accurately reect the counts from the trace.11

Simulated Cache Test Plan (Team 4) ECE 5852. Conrm Mo de Toggle Functionality Without Recompila-tion2.1 Toggle Silent and Normal Mo de with Command-Line ArgumentPro cedure:1. Run a trace le rst in Silent Mo de and conrm that only the summary isdisplayed.2. Without recompiling, run the same trace le in Normal Mo de (by providing adierent command-line argument) and conrm that detailed logs and summarystatistics app ear.Exp ected Outcome:Silent and Normal Mo des can b e toggled easily without requiring recompilation,demonstrating mo de exibility and allowing either mo de to b e selected as needed.12

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryAfter executing each test:‹Suppression of Detailed Logs:Detailed logs are correctly suppressed inSilent Mo de, with no intermediate cache op eration logs or sno op results dis-played.‹Summary Statistics:Silent Mo de displays only the summary statistics atthe end of the trace, covering the key metrics: reads, writes, hits, misses, andhit ratio.‹Mo de Toggle Functionality:Mo de switching works without recompilation,enabling seamless toggling b etween Silent and Normal Mo des.13

Simulated Cache Test Plan (Team 4) ECE 5854 Asso ciativity VericationOb jectives‹Verify 16-Way Asso ciativity:Ensure the cache can store up to 16 uniquelines in a single set. In a 16-way asso ciative cache, each set can hold 16 dierentcache lines, allowing multiple memory addres ses mapping to the same set indexto co exist.‹Check Cache Set Overow:Conrm that when more than 16 unique linesare mapp ed to the same set, the cache correctly evicts lines according to theeviction p olicy, maintaining a maximum of 16 lines p er set.‹Validate Address Mapping and Set Distribution:Ensure that lines withthe same set index but dierent tags are stored in dierent ways within eachset, conrming correct mapping and distribution.14

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify 16-Way Asso ciativity1.1 Fill a Set with 16 Unique LinesPro cedure:1. Select a single set index and generate 16 unique addresses that should map tothis set.2. Load each address into the cache sequentially and conrm that all 16 lines tinto the set without eviction.Exp ected Outcome:All 16 lines should b e s tored in the selected set, conrming that the set can hold upto 16 unique lines in a 16-way set asso ciative cache.1.2 Verify Prop er Distribution of Lines Across 16 WaysPro cedure:1. Load 16 unique addresses into a single set.2. Verify that each address maps to a dierent way within the set, ensuring thatthe cache uses all 16 ways for the given set.Exp ected Outcome:Each address is correctly mapp ed to a separate way within the set, demonstratingprop er distribution and usage of all 16 ways.15

Simulated Cache Test Plan (Team 4) ECE 5852. Check Cache Set Overow2.1 Exceed Capacity of a Set (Eviction on 17th Line)Pro cedure:1. Load 16 unique addresses into a single set.2. Add a 17th unique address and conrm that it causes an eviction in the selectedset, which already holds 16 lines. (This do es not need to explicitly test pseudo-LRU, just that an "overow" in our set causes an eviction.Exp ected Outcome:The 17th line triggers an eviction in the selected set, conrming that the cachecorrectly enforces the set's capacity limit of 16 lines.16

Simulated Cache Test Plan (Team 4) ECE 5853. Mapping and Set Distribution3.1 Verify Cross-Set Mapping (Dierent Index & Tags )Pro cedure:1. Cho ose multiple addresses with dierent indexes and tags.2. Load these addresses into the cache and conrm that they are mapp ed todierent sets.Exp ected Outcome:Addresses with dierent set indexes should map to dierent sets, ensuring that thecache uses b oth the index and tag to determine set placement correctly.3.2 Verify Mapping When Tags Are Identical but Indexes Di erPro cedure:1. Cho ose addresses with identical tags but dierent indexes.2. Load these addresses into the cache and verify that they are placed in dierentsets according to their indexes.Exp ected Outcome:Addresses with the same tag but dierent indexes should b e placed in dierent sets,conrming that the index correctly determines s et placement.3.3 Hit VericationPro cedure:1. Fill one set halfway (same index, but 8 dierent tags).2. Reference one of the tags that already resides in the set again.Exp ected Outcome:The set should have 8 total entries, accessing the same element again should yield ahit and not a line ll.17

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹16-way asso ciativity is maintained:Each set holds up to 16 unique lines ,and no more, conrming that the cache b ehaves as a 16-way set asso ciativecache.‹Cache overow b ehavior is correct:When more than 16 lines are mapp edto a set, the cache evicts one line to maintain the 16-line capacity p er set.‹Address mapping and distribution is accurate:Addresses are correctlymapp ed to the appropriate sets based on their index, and the cache ensuresthat no two lines with the same index and tag (duplicates) reside in the cache.18

Simulated Cache Test Plan (Team 4) ECE 5855 Alignment VericationOb jectives‹Ensure Prop er Handling of Address Alignment:Verify that the cachecorrectly handles memory addresses, ensuring that they align with the cacheline size (64 bytes) as required by the system.‹Verify Cache Line Addressing Do es Not Span Across Multiple Lines:Conrm that memory references landing on cache line b oundaries result in asingle cache line ll, without spanning multiple lines.‹Check for Correct Op eration when Address is Misaligned:Ensure thesystem correctly fetches the appropriate 64-byte chunk based on the misalignedaddress, aligning it with the correct cache line b oundaries.19

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Ensure Prop er Handling of Address Alignment1.1 Test Aligned Address A cce ssPro cedure:1. Run a trace le where the rst address reference is aligned to the cache linesize (64 bytes).2. Make subsequent accesses that dier in address (Byte Select) but corresp ondto the same 64-byte chunk of memory.Exp ected Outcome:All requests after the initial ll should result in a hit to the line cached with the rstaccess, conrming prop er handling of aligned addresses.1.2 Test for Cache Line Boundary Integrity (Accesses Near Boundaries)Pro cedure:1. Run a trace le that includes references near cache line b oundaries (e.g., 0x3F,0x40, 0x7F, 0x80, etc.).2. Ensure that each access:‹Results in a separate cache miss for each cache line.‹Do es not span across multiple cache lines, ensuring that data from dier-ent cache lines is fetched correctly without overlap.3. Test byte-level accesses near the b oundary (e.g., 0x3F, 0x40) to verify that nodata crosses the b oundary b etween two cache lines.Exp ected Outcome:Each address (including byte-level accesses) should b e conned to its resp ective 64-byte cache line. No access should span across cache lines, and each access near a20

Simulated Cache Test Plan (Team 4) ECE 585b oundary (whether byte-level or word-level) should result in a miss and a cache linell for the appropriate line.2. Check for Correct Op eration when Address is Misaligned2.1 Test for Misaligned AccessPro cedure:1. Run a trace that includes misaligned address references (e.g., addresses notdivisible by 64).2. Verify that the correct 64-byte cache line containing the requested address isfetched from DRAM.Exp ected Outcome:The fetched cache line should contain the selected byte, with the address correctlyoset from the aligned 64-byte line fetched from memory. For example, if the ad-dress ends with 0x4F, the line fetched should span from 0x40 to 0x7F, including therequested address.2.2 Test for Handling of Multiple Misaligned A ddres sesPro cedure:1. Run a trace with multiple misaligned addresses, each within a dierent 64-bytecache line (e.g., 0x00, 0x4F, 0x80, 0xC7).2. Verify that each misaligned address results in a correct cache line fetch fromDRAM.Exp ected Outcome:The system should correctly handle each misaligned address and fetch the appro-priate 64-byte chunk from memory, ensuring that no byte access es spill over intoadjacent cache lines.21

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Aligned Addresses:All memory addresses should b e aligned with the cacheline size (64 bytes), with no errors in cache op erations.‹Cache Line Boundary Integrity:Memory accesses should never span acrossmultiple cache lines, as p er the pro ject assumption. Accesses near cache lineb oundaries (including byte-level) should correctly map to the appropriate cacheline.‹Misaligned Access Handling:The system should correctly handle mis-aligned addresses by fetching the correct 64-byte chunk, ensuring prop er align-ment or adjusting the access as required.22

Simulated Cache Test Plan (Team 4) ECE 5856 Pseudo-LRU Replacement Policy VericationOb jectives‹Verify Pseudo-LRU on Set Overow:Ensure that when a 17th line isadded to a set already containing 16 lines, the cache correctly evicts the leastrecently used line according to the pseudo-LRU p olicy.‹Validate Pseudo-LRU Tracking on A ccess:Verify that each cache access(read/write) prop erly up dates the pseudo-LRU tracking bits to reect the mostrecently used cache line.‹Test E dge Cases for Eviction Order:Test eviction in scenarios whereonly a subset of lines have b een accessed, ensuring the least recently used lineis evicted even if it has not b een accessed for some time.23

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Pseudo-LRU on Set Overow1.1 Insert 17 Unique Lines into a Set and Test EvictionPro cedure:1. Fill a set with 16 unique lines.2. Access some lines in a sp ecic order (e.g., lines 1, 2, 3) to establish a pseudo-LRU state.3. Add a 17th line and verify that the least recently used line (according to thepseudo-LRU p olicy) is evicted.Exp ected Outcome:The cache evicts the pseudo least recently used line, conrming that the pseudo-LRUp olicy works correctly on overow and that the eviction resp ects the access history.24

Simulated Cache Test Plan (Team 4) ECE 5852. Validate Pseudo-LRU Tracking on Access2.1 Up date Pseudo-LRU Tracking on AccessPro cedure:1. Fill a set with 16 lines to establish the initial pseudo-LRU state.2. Access lines in various orders (e.g., access lines 5, 3, 8, etc.).3. After each access, check that the pseudo-LRU tracking bits are up dated, mark-ing the accessed line as the most recently used.Exp ected Outcome:Each access correctly up dates the pseudo-LRU state, with the most recently accessedline marked as "most recent" and the rest in their correct relative order.25

Simulated Cache Test Plan (Team 4) ECE 5853. Test Edge Cases for Eviction Order3.1 Evict the Least Recently Used Line After Partial AccessPro cedure:1. Fill a set with 16 lines.2. Access only a subset of lines (e.g., lines 1, 3, 5), leaving others untouched.3. Add a 17th line and verify that the least recently used untouched line is evicted.Exp ected Outcome:The cache evicts the least recently used line that has not b een accessed, even if ithas not b een accessed since its initial load.3.2 Consistent Eviction with Mixed Access PatternsPro cedure:1. Load 16 lines into the cache.2. Access the lines in a non-sequential pattern (e.g., 1, 16, 2, 15, etc.).3. Add a 17th line and conrm that the eviction follows the pseudo-LRU order,evicting the leas t recently used line.Exp ected Outcome:The least recently used line is evicted, regardless of the access pattern, ensuringconsistent eviction b ehavior.26

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Pseudo-LRU Replacement on Set Overow:The cache correctly evictsthe least recently used line when a 17th line is added to a fully p opulated set,according to the pseudo-LRU p olicy.‹Accurate Tracking of Cache Accesses:The pseudo-LRU bits are correctlyup dated after each cache access, reecting the most recently used state.‹Edge Case Handling:Even with partial or non-sequential accesses, the cachecorrectly evicts the least recently used line, demonstrating robust eviction b e-havior.27

Simulated Cache Test Plan (Team 4) ECE 5857 Write-Allo cate ValidationWrite-Allo cate Test PlanOb j ective:Verify that the cache correctly implements the Write-Allo cate p olicyfor write misses. Ensure prop er cache line allo cation, loading, and state transitions,while also verifying that cache statistics are up dated accordingly.‹Verify Write-Allo cate Behavior on Write Misses :Conrm that a cachemiss on a write results in the allo cation of a new cache line and writing thedata into the cache instead of bypassing it.‹Ensure Cache Line is Prop erly Allo cated and Loaded:Ensure thatwhen a new line is allo cated due to a write miss, the entire cache line (typically64 bytes) is fetched from the next level of memory and placed into the cache.‹Validate Subsequent Writes to the Allo cated Line:Ensure that after awrite miss triggers a write-allo cate, subsequent writes to the same address hitthe cache.‹Conrm Cons istency with MESI Proto col:Verify that the cache line'sstate transitions according to the MESI proto col after a write-allo cate (e.g.,transitioning to Mo died).‹Verify Statistics Up date for Write-Allo cate Events:Ensure that cachestatistics, such as write misses and cache allo cations, are correctly up datedduring write-allo cate op erations.28

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Write-Allo cate Behavior on Write Misses1.1 Perform Write Miss on an Empty CachePro cedure:1. Use a trace le where the rst op eration is a write to an address not currentlyin the cache.2. Observe the cache b ehavior: conrm that a write miss o ccurs and that thecache allo cates a new cache line for the missed address.Exp ected Outcome:The cache rep orts a write miss, and a new cache line is allo cated for the missedaddress.2. Ensure Cache Line is Prop erly Allo cated and Loaded Dur-ing Write-Allo cate2.1 Check Cache Line Loading After Write MissPro cedure:1. Use a trace le with a write miss to an address in a full set.2. Verify that the cache evicts the prop er line via Pseudo-LRU and loads the fullcache line (e.g., 64 bytes) from the next level of memory during the write-allo cate pro cess.Exp ected Outcome:The Pseudo-Least-Recently-Used cache line is evicted, and the line corresp onding tothe missed address is fetched and loaded into the cache.29

Simulated Cache Test Plan (Team 4) ECE 5853. Validate Subsequent Writes to the Allo cated Line3.1 Perform Multiple Writes After Write-Allo catePro cedure:1. Use a trace where a write miss o ccurs, followed by additional writes to thesame address.2. Verify that the subsequent writes result in cache hits.Exp ected Outcome:The rst write caus es a miss and triggers write-allo cate. Subsequent writes to thesame address should hit the cache.4. Conrm Consistency with MESI Proto col4.1 Verify State Transition After Write-Allo catePro cedure:1. Use a trace le that triggers a write miss, res ulting in a write-allo cate.2. Observe the MESI state of the allo cated line to conrm it transitions appro-priately to Mo died.Exp ected Outcome:The cache line should transition to the correct MESI (Mo died) state after the write-allo cate op eration.30

Simulated Cache Test Plan (Team 4) ECE 5855. Verify Statistics Up date f or Write-Allo cate Events5.1 Check Cache Statis tics After Write-Allo catePro cedure:1. Use a trace le with a known numb er of write misses.2. Verify that the numb er of write misses and cache allo cations rep orted in thestatistics matches the exp ected values.Exp ected Outcome:Cache s tatistics should accurately reect the numb er of write misses and allo cationscaused by write-allo cate op erations.31

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryAfter executing each test:‹Write Miss Behavior:Write misses should trigger the allo cation of a newcache line (write-allo cate).‹Line Loading:The entire cache line should b e loaded into the cache from thenext level during the write-allo cate pro cess.‹Subsequent Writes:Subsequent writes to the allo cated line should result incache hits.‹MESI Proto col Compliance:Cache line state transitions should follow theMESI proto col (e.g., to Mo died) after a write-allo cate.‹Statistics:Cache statistics should correctly track write misses and allo cationstriggered by the write-allo cate p olicy.This test plan ensures that the write-allo cate p olicy is correctly implemented,with prop er cache line allo cation, loading, subsequent hits, MESI state transitions,and accurate statistics tracking.32

Simulated Cache Test Plan (Team 4) ECE 5858 MESI State AccuracyOb jectiveTo thoroughly test the correct implementation of the MESI (Mo died, Exclusive,Shared, Invalid) proto col in cache memory systems. This includes verifying correctstate transitions during cache read, write, sno op, and eviction op erations, ensur-ing cache coherence in a multi-pro cessor environment, and tracking MESI-relatedstatistics.MESI States Overview‹Mo died (M): The cache has a copy of the data that has b een mo died, andit is the only cache with this data. The cache is resp onsible for writing backthe data when evicted.‹Exclusive (E): The cache has the only copy of the data, but the data has notb een mo died. This means no other cache has the line, and it is clean in thiscache.‹Shared (S): The cache line is present in multiple caches, and it is not mo died(it is consistent with memory).‹Invalid (I): The cache do es not have a valid copy of the data.33

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. State Transitions on Cache ReadsA cache read can result in dierent state transitions based on the current state ofthe cache line.1.1 Read to a Shared Line (S)Pro cedure:1. Simulate a read request to a line in theSharedstate, which is cached bymultiple pro cessors.2. Conrm that no state change o ccurs b ecause the cache line is shared.Exp ected Outcome:The cache should remain in theSharedstate. The line is shared among multiplecaches, so no transition o ccurs.1.2 Read to an Invalid Line (I)Pro cedure:1. Simulate a read request to a line in theInvalidstate.2. Verify the state transition based on the cache conguration (i.e., whether thedata is fetched from memory or another cache).Case 1: Fetching from a CachePro cedure:1. The data is cached by another pro cessor, and the cache line transitions toSharedin the sending pro cessor (if previously Exclusive).2. The line is markedSharedin the requesting cache.34

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Outcome:The cache line transitions toShared(S) in b oth caches since it is b eing sharedb etween them.Case 2: Fetching from MemoryPro cedure:1. The cache needs to fetch the line from memory.2. The cache line entersExclusive(E) since this is the only pro cessor accessingthe data at that p oint.Exp ected Outcome:The line enters theExclusivestate, and it is clean.1.3 Read to a Line in the Exclusive State (E)Pro cedure:1. Simulate a read request to a line that is in theExclusivestate, where onlyone cache has it.2. Conrm that no transition o ccurs if the requesting pro cessor is the same as theowner.Exp ected Outcome:No state change o ccurs, and the line remains in theExclusivestate if the cache hasthe only copy of the data.35

Simulated Cache Test Plan (Team 4) ECE 5852. State Transitions on Cache WritesWriting to a cache line triggers state changes in b oth the requesting cache and othercaches, if they have the same line.2.1 Write to an Invalid L ine (I)Pro cedure:1. Simulate a write request to a line in theInvalidstate.2. Verify that the cache line transitions to theMo diedstate after fetching thedata.Exp ected Outcome:The cache transitions the line to theMo diedstate (M). The cache is the only onethat has a copy of the data, and it is mo died.2.2 Write to a Shared Line (S)Pro cedure:1. Simulate a write request to a line in theSharedstate.2. The cache should broadcast an invalidation to other caches that may have acopy of the line.3. Conrm that the line transitions to theMo diedstate in the requesting cache.Exp ected Outcome:The requesting cache moves the line toMo died(M), and all other caches invalidatethe line (I).2.3 Write to an Exclusive Line (E)Pro cedure:1. Simulate a write request to a line in theExclusivestate.36

Simulated Cache Test Plan (Team 4) ECE 5852. The cache should transition the line to theMo diedstate.Exp ected Outcome:The cache moves the line toMo died(M), indicating that the cache has the onlycopy of the line and has mo died it.37

Simulated Cache Test Plan (Team 4) ECE 5853. Sno oping BehaviorSno oping is the pro cess by which caches observe the bus to detect and resp ond tothe actions of other pro cessors.3.1 Handle Bus Read Op eration to a Mo died L ine (M)Pro cedure:1. Simulate a bus read op eration requesting data from a line in theMo diedstate.2. The sending and requesting cache should transition the line toSharedandprovide the data.Exp ected Outcome:The line transitions toShared(S), and the cache supplies the data on the bus tob e snarfed while the up dated data is written back to the DRAM.3.2 Handle Bus Read with Intent to M o dify (RWIM)Pro cedure:1. Simulate a bus read with intent to mo dify (RWIM) for a line in theMo diedstate.2. The cache should invalidate the line and provide the data to the requestingpro cessor.Exp ected Outcome:The line transitions toInvalid(I), and the cache provides the data to the bus.38

Simulated Cache Test Plan (Team 4) ECE 5853.3 Handle Bus Write Op eration to a Shared Line (S)Pro cedure:1. Simulate a bus write op eration that writes to a line in theSharedstate.2. The requesting cache should invalidate the line in other caches.Exp ected Outcome:The cache transitions the line toMo died(M), and all other caches invalidate theircopy of the line.4. Shared State (S) ManagementShared state management refers to situations where the cache line is present inmultiple caches and must remain consistent.4.1 Verify Multiple Reads by Dierent Pro cessorsPro cedure:1. Simulate multiple pro cessors reading the same line.2. Conrm that the line remains in theSharedstate across all caches.Exp ected Outcome:The line remains in theSharedstate, as it is b eing accessed by multiple pro cessors.4.2 Handle Read/Write ConictPro cedure:1. One pro cessor p erforms a read, and another p erforms a write to the same line.2. Verify that the line transitions to theMo diedstate for the writing cache,and other caches are invalidated.Exp ected Outcome:The line in the writing cache transitions toMo died(M), while the read cacheremains inShared(S) until the write op eration invalidates the shared copies.39

Simulated Cache Test Plan (Team 4) ECE 5855. Exclusive State (E) ManagementThe Exclusive state represents when a cache has the only copy of the data and isclean.5.1 Verify Exclusive State EntryPro cedure:1. Simulate a read to a cache line that no other cache has accessed.2. Conrm that the line enters theExclusivestate.Exp ected Outcome:The cache line transitions to theExclusivestate since this is the only pro cessoraccessing the data and no mo dication has b een made.6. Mo died State (M) ManagementThe Mo died state represents when a cache holds the only copy of the mo died data.6.1 Test Eviction of Mo died LinePro cedure:1. Simulate a cache replacement for a line in theMo diedstate.2. Verify that the data is written back to memory b efore eviction.Exp ected Outcome:The line is written back to memory b efore b eing evicted to maintain consistency.40

Simulated Cache Test Plan (Team 4) ECE 5857. Cache Coherence Across Multiple Pro cessorsIn multi-pro cessor systems, the MESI proto col ensures that all caches are kept co-herent.7.1 Simulate Multi-Pro cessor AccessPro cedure:1. Simulate multiple pro cessors accessing and mo difying the same cache line.2. Verify that all caches maintain coherence, including the prop er invalidationsand state transitions.Exp ected Outcome:Cache coherence is maintained across all pro cessors, with appropriate invalidations,state transitions, and data up dates.8. Statistics Up date During MESI Op erations8.1 Check Statistics for State TransitionsPro cedure:1. Use a trace le designed to exercise all MESI state transitions.2. Verify that state transition counts and related statistics are correctly recorded.Exp ected Outcome:Statistics should match the exp ected numb er of transitions and op erations, reectingcorrect MESI proto col b ehavior.41

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryAfter executing each test:‹State Transitions:All MESI state transitions are correct for reads, writes,and sno oping.‹Shared, Exclusive, and Mo died States:Each state is entered and man-aged correctly based on access patterns.‹Sno oping Behavior:Cache resp onds correctly to bus op erations from otherpro cessors.‹Cache Coherence:MESI proto col maintains consistency across all caches .‹Statistics:Cache statistics accurately reect MESI op erations.42

Simulated Cache Test Plan (Team 4) ECE 5859 Cache CoherenceOb jectives‹Verify Execution of Bus Read and Write Op erations:Ensure the LLCinitiates, pro cesses, and correctly handles all bus op erations, including read,write, invalidate, and RWIM, while maintaining consistency with memory andother caches.‹Validate MESI Proto col Compliance:Conrm the prop er implementa-tion of all MESI state transitions during cache reads, writes, sno oping, andevictions.‹Test Multi-Pro cessor Shared Memory Op erations:Verify cache coher-ence and conict resolution when multiple pro cessors read, write, and mo difyshared cache lines.‹Evaluate Sno oping Behavior:Ensure the LLC observes and resp onds ac-curately to bus op erations initiated by other caches or pro cessors.‹Monitor Rep orting to Higher-Level Caches:Conrm that the LLC com-municates evictions, state changes, and cache statistics accurately and e-ciently.‹Measure and Rep ort Cache Statistics:Ensure precise tracking of MESItransitions, sno oping resp onses, invalidations, and bus op erations, with statis-tics matching exp ected outcomes.43

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Execution of Bus Read Op erations1.1 Cache Miss Triggers Bus ReadPro cedure:1. Simulate a cache miss for a read request.2. Verify that the cache initiates a bus read op eration to fetch the required data.Exp ected Outcome:The cache sends a bus read request, and the data is fetched.1.2 Sno op Result HIT on Bus ReadPro cedure:1. Simulate a bus read op eration where another cache has the line in an unmo di-ed valid state.2. Verify that the cache that already contains the line sends a HIT signal alongwith its MESI state on the bus.Exp ected Outcome:The cache receives the data, HIT is sent on the bus, and the line transitions correctly.1.3 Cache Miss for Line in Exclusive State in Another CachePro cedure:1. Simulate a read request for a line in an Exclusive state in another cache.2. Ensure the other cache transitions to the Shared state.Exp ected Outcome:The requesting cache transitions the line to Shared. The original Exclusive cachealso transitions the line to Shared.44

Simulated Cache Test Plan (Team 4) ECE 5852. Verify Execution of Bus Write Op erations2.1 Writeback on Mo died Line EvictionPro cedure:1. Simulate the eviction of a line in the Mo died state.2. Conrm that the cache initiates a bus write op eration to write back the mo d-ied data to memory.Exp ected Outcome:Data is written back to memory via a bus write op eration.2.2 Eviction of a Shared Line in 2 L1'sPro cedure:1. Simulate the eviction of a line in a Shared state from one L1, that remains inone other L1.Exp ected Outcome:No write-back to memory o ccurs. The line transitions to I nvalid, and the other cachetransitions to exclus ive.45

Simulated Cache Test Plan (Team 4) ECE 5853. Verify Bus Read with Intent to Mo dify (RWIM)3.1 RWIM for Line in Shared StatePro cedure:1. Simulate a write request to a line in the Shared state.2. Verify that the cache initiates a RWIM op eration on the bus to gain exclusiveownership of the line.Exp ected Outcome:The CPU sends an RWIM request, the requesting cache transitions the line to theMo died state, and the line b ecomes invalid in the other shared caches.3.2 RWIM for Line in Invalid StatePro cedure:1. Simulate a write request to a line in the Invalid state.2. Verify that the cache initiates a RWIM op eration and fetches the data frommemory b efore mo dication.Exp ected Outcome:The line transitions to Mo died, and the data is fetched b efore mo dication.46

Simulated Cache Test Plan (Team 4) ECE 5854. Verify Bus Invalidate Op erations4.1 Invalidate Shared Lines on WritePro cedure:1. Simulate a write request to a line in the Shared state.2. Verify that the cache broadcasts an invalidate op eration on the bus to the othercaches.Exp ected Outcome:All other caches invalidate their copies of the line.4.2 Invalidate Other Lines During RWIMPro cedure:1. Simulate a RWIM op eration for a line shared across multiple caches.2. Conrm that other caches invalidate their copies in resp onse to the RWIM.Exp ected Outcome:All other caches receive an invalidate bus op eration and their copies transition tothe invalid state.47

Simulated Cache Test Plan (Team 4) ECE 5855. Validate Resp onses to Bus Sno oping5.1 Resp ond with HITPro cedure:1. Simulate a sno op ed bus read op eration for a line in the Shared or Exclusivestate.2. Conrm the cache rep orts a HIT resp onse.Exp ected Outcome:The cache rep orts HIT.5.2 Resp ond with HITMPro cedure:1. Simulate a sno op ed read bus op eration for a line in the Mo died state.2. Conrm the cache rep orts a HI TM resp onse and writes back the data toDRAM.Exp ected Outcome:The cache rep orts HITM and writes back the line.5.3 No Resp onsePro cedure:1. Simulate a sno op ed bus read op eration for a line in the invalid state in allcaches.2. Verify there is no resp onse for the other caches observed on the bus .Exp ected Outcome:The caches do not reply with HIT or HITM.48

Simulated Cache Test Plan (Team 4) ECE 5856. Verify Cache Statistics for Bus Op erations6.1 Count Bus Reads and WritesPro cedure:1. Use a trace le that triggers multiple read and write op erations.2. Verify that the cache correctly counts the numb er of bus reads and writes.Exp ected Outcome:Statistics for bus reads and writes match the exp ected count.6.2 Count Invalidate and RWIM Op erationsPro cedure:1. Use a trace le with scenarios requiring invalidation and RWIM op erations.2. Conrm that the cache accurately tracks the count of these op erations.Exp ected Outcome:Statistics for invalidations and RWIM match exp ectations.49

Simulated Cache Test Plan (Team 4) ECE 5857. Rep orting Results to Higher-Level Caches7.1 Eviction of Mo died LinePro cedure:1. Simulate the eviction of a line in the Mo died state.2. Conrm that the L1 sends the mo died data to the LLC for write-back.Exp ected Outcome:‹Mo died data is written back to memory. The MESI state transitions to in-valid.7.2 Rep orting Line State ChangesPro cedure:1. Simulate a sno oping event that causes a transition of a cache line from Mo diedto Shared or Exclusive.2. Verify that the LLC up dates the higher-level cache with the new state.Exp ected Outcome:‹Higher-level cache is informed of state transitions.‹No inconsistencies o ccur b etween the LLC and higher-level cache.50

Simulated Cache Test Plan (Team 4) ECE 5857.3 Rep orting Cache StatisticsPro cedure:1. Run a workload trace that triggers multiple state transitions and bus op era-tions.2. Verify that the LLC aggregates and rep orts accurate statistics, including hits,misses, invalidations, and RWIM counts.Exp ected Outcome:‹Rep orted statistics match exp ected values based on the trace.‹No loss of accuracy or data during rep orting.51

Simulated Cache Test Plan (Team 4) ECE 5858. Multi-Pro cessor Shared Memory Conguration8.1 Simultaneous Reads by Multiple Pro cessorsPro cedure:1. Simulate multiple pro cessors issuing read requests for the same cache line.2. Verify that the cache line remains in the Shared state across all pro cessors.Exp ected Outcome:Line remains in the Shared state. Data consistency is maintained across all pro-cessors.8.2 Simultaneous Write ConictsPro cedure:1. Simulate multiple pro cessors attempting to write to the same cache line.2. Monitor how the LLC prioritizes the op erations and resolves conicts.Exp ected Outcome:One pro cessor must wait until the data has b een written to and read back to theDRAM b efore completing its write. The rst pro cessor to write to the line must alsotransition into the invalid state.8.3 Mixed Read/Write Op erationsPro cedure:1. Simulate a scenario where one pro cess or reads a line while another attemptsto write to the same line.2. Verify prop er handling of invalidations and MESI trans itions.52

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Outcome:The reading pro cessor receives valid data b efore the writing one. When the writingpro cessor receives the line, it transitions the line to Mo died, and all other copiesmove to invalid.53

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryUp on executing all tests:‹Bus Op erations:All bus reads, writes, invalidations and RWIM op erationsexecute correctly, maintaining consistency with memory and other caches.‹MESI Proto col:State transitions comply fully with MESI rules, ensuringcorrect handling of reads, writes, sno oping, and evictions.‹Multi-Pro cessor Coherence:Shared memory consistency is maintainedacross all pro cessors during simultaneous reads, writes, and conicts.‹Sno oping Behavior:The LLC resp onds accurately to all sno oping op era-tions, including HIT, HITM, and NOHIT, while maintaining coherence.‹Statistics Rep orting:Cache statistics, including hits, misses, invalidations,and RWIM op erations, match exp ected values with no inaccuracies.‹Higher-Level Cache Communication:The LLC correctly rep orts statechanges, evictions, and aggregated statistics to the higher-level cache.54

Simulated Cache Test Plan (Team 4) ECE 58510 Inclusivity VericationOb jectives:Ensure that the Last-Level Cache (LLC) is inclusive of all lines present in the higher-level cache (L1), and verify correct eviction handling, mo dication propagation, andcache consistency across multiple pro cessors.‹Verify Line Inclusion:Every line in the higher-level cache (L1) must alsob e present in the LLC.‹Eviction Handling:When a line is evicted from the LLC, it must b e invali-dated in the higher-level cache (L1) as well.‹Mo dication Propagation (M ESI Proto col):Mo dications in the higher-level cache (L1) s hould propagate to the LLC, ensuring consistency.‹Concurrent Access by Multiple Pro cessors:LLC should maintain inclu-sivity and consistency when multiple pro cessors access shared data.‹Replacement Policy (Pseudo-LRU):Ensure the LLC eviction follows Pseudo-LRU, and evicted lines are also removed from L1.‹Forced Invalidation via Sno oping:Invalidation commands from the busmust propagate to b oth LLC and L1 caches.‹Cache Clearing Op erations:Clearing the LLC should invalidate corre-sp onding lines in the higher-level cache (L1).55

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Line Inclusion in LLC1.1 Basic Op erations for Line InclusionPro cedure:1. Perform a read op eration for a line not present in LLC.2. The line should b e loaded into LLC and also propagated to L1 if not alreadypresent there.Exp ected Outcome:The cache line should app ear in b oth LLC and L1, conrming line inclusion.2. Mo dication Propagation via MESI Proto col2.1 Reect Mo dications from L1 to LLCPro cedure:1. Load a line into b oth LLC and L1.2. Mo dify the line in L1 (e.g., write op eration).3. Verify that the mo dication in L1 is propagated to LLC, and LLC reects thesame state according to the MESI proto col.Exp ected Outcome:LLC should reect the mo dication from L1, ensuring data consistency across caches.56

Simulated Cache Test Plan (Team 4) ECE 5853. Concurrent Access by M ultiple Pro cessors3.1 Inclusivity with Shared DataPro cedure:1. Simulate two pro cessors loading the same line into their L1 caches.2. Evict the line from LLC due to access by a third pro cessor.3. Verify that eviction from LLC invalidates the line in b oth L1 caches.Exp ected Outcome:The evicted line in LLC should b e invalidated in all L1 caches that had a copy of it,ensuring consistency in a multi-pro cessor setup.4. Replacement Policy Verication (Pseudo-LRU)4.1 Eviction and Replacement in LLC and L1Pro cedure:1. Populate LLC with unique addresses until it reaches capacity.2. Access a new address, triggering a Pseudo-LRU eviction in LLC.3. Verify that any evicted line in LLC is also evicted in L1 if it was present.Exp ected Outcome:Evicted lines from LLC should b e removed from L1 as well, consistent with thepseudo-LRU replacement p olicy.57

Simulated Cache Test Plan (Team 4) ECE 5855. Forced Invalidation via Ex ternal Sno oping5.1 Handle External Invalidation RequestsPro cedure:1. Load a line into b oth LLC and L1.2. Issue an external invalidate command for this address via bus sno oping that isseen by the LLC.3. Verify that the line is invalidated in b oth LLC and L1.Exp ected Outcome:The line should b e invalidated in b oth LLC and L1, ensuring cons is tency in res p onseto external invalidation.6. Cache Clearing Op erations6.1 Clear All Cache Entries in LLC and L1Pro cedure:1. Populate b oth LLC and L1 caches with data.2. Issue a clear command for LLC, eectively resetting the cache.3. Verify that all entries are removed from b oth LLC and L1.Exp ected Outcome:Both LLC and L1 should b e cleared of all entries, conrming that cache clearingop erates across the entire hierarchy.58

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Line Inclusion in LLC:Every line in L1 should b e mirrored in the LLC.‹Eviction Handling:Evictions in LLC must b e reected in L1 by invalidatingor removing the evicted lines.‹Mo dication Propagation:Mo dications in L1 should propagate to LLC,ensuring data consis tency.‹Concurrent A ccess:Evictions in LLC due to multiple pro cessors must in-validate the corresp onding lines in L1.‹Replacement Policy (Pseudo-LRU):Evicted lines from LLC should b eremoved from L1 in accordance with the Pseudo-LRU p olicy.‹Forced Invalidation via Sno oping:External sno oping commands mustinvalidate lines in b oth LLC and L1.‹Cache Clearing:Clearing LLC should also clear L1, resetting the cachesystem.This test plan veries the LLC's inclusivity and ensures the correctness of evic-tion, mo dication propagation, concurrent access, and cache clearing op erationsacross the entire cache hierarchy.59

Simulated Cache Test Plan (Team 4) ECE 58511 Write-Once PolicyOb jectivesEnsure that the L1 cache correctly implements the write-once p olicy where the rstwrite to a line is write-through to the LLC, and subsequent writes to the same lineare write-back. The test plan should verify the b ehavior of the L1 cache and itsinteractions with the LLC while maintaining inclusivity.‹First Write Verication:Conrm that the rst write to a line in the L1cache results in a write-through to the LLC, up dating b oth caches.‹Subsequent Write Verication:Validate that subsequent writes to thesame line in the L1 cache are handled as write-back, up dating only the L1cache.‹MESI State Transitions:Verify that the L1 cache transitions through thecorrect MESI states to reect the write-once p olicy and ensure data consistencyb etween L1 and LLC.‹Inclusivity Mainte nance:Ensure that despite the write-back p olicy for sub-sequent writes, the LLC maintains inclusivity by retaining a copy of the line.60

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. First Write to a Cache Line1.1 Write to an Empty LinePro cedure:1. Initialize the L1 cache as empty.2. Simulate a write request from the pro cessor to a new address that maps to theL1 cache.Exp ected Outcome:‹The L1 cache rep orts a write miss b ecause the line is not present.‹The write is p erformed as write-through, up dating b oth the L1 cache and theLLC.‹The L1 cache line transitions to the Mo died state, reecting that it holds theonly up-to-date copy of the data.‹The LLC receives the write data through a bus write op eration and up datesits corresp onding line.61

Simulated Cache Test Plan (Team 4) ECE 5852. Subsequent Writes to the Same Cache Line2.1 Write to a Mo died LinePro cedure:1. Simulate a write request from the pro cessor to an address that already residesin the L1 cache and is in the Mo died or Exclusive state.Exp ected Outcome:‹The L1 cache rep orts a write hit as the line is already present.‹The write is handled as write-back, meaning only the L1 cache is up dated.‹The L1 cache line remains in the Mo died state, indicating the data has b eenmo died.‹The LLC is not involved in this write op eration.62

Simulated Cache Test Plan (Team 4) ECE 5853. Statistics and State Tracking3.1 Verify Write Counts and State TransitionsPro cedure:1. Execute a trace le containing a sequence of write requests, including b oth rstwrites and subsequent writes to the same cache lines.2. Monitor the L1 cache to track the numb er of write-through and write-backop erations.3. Observe the MESI state transitions of the L1 cache lines to ensure they correctlyreect the write-once p olicy.Exp ected Outcome:‹The L1 cache statistics accurately rep ort the numb er of write-through op era-tions, matching the count of rst writes to the lines.‹The L1 cache statistics accurately rep ort the numb er of write-back op erations,corresp onding to the subsequent writes to the same lines.‹The MESI states of the L1 cache lines transition as exp ected:{First write: transitions to Mo died.{Subsequent writes: remains in the Mo died s tate.63

Simulated Cache Test Plan (Team 4) ECE 5854. Inclusivity4.1 Check Inclusivity After Write-BackPro cedure:1. Perform a write to a line in the L1 cache (Write-Through).2. Perform subsequent writes to the line (so that the LLC has an outdated copyof the data).3. Initiate an eviction of the corresp onding line from the LLC.Exp ected Outcome:‹The only up-to-date data should b e found in the L1, so it should rst b e writtenback to the LLC.‹After writing to the LLC the L1 should invalidate its line.‹The LLC should write the data back to DRAM, also invalidating its copy.64

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm:‹First Write Handling:The rst write to a line in the L1 cache is treated aswrite-through, up dating b oth the L1 cache and the LLC.‹Subsequent Write Handling:Subsequent writes to the same line in the L1cache are p erformed as write-back, up dating only the L1 cache.‹MESI State Compliance:The L1 cache correctly transitions MESI states toensure data consistency b etween itself and the LLC according to the write-oncep olicy.‹Inclusivity:Inclusivity is maintained b etween the L1 cache and the LLC,even with the write-back p olicy used for subsequent writes. The LLC retainsa copy of the line even if it is not up dated with every write.65

Simulated Cache Test Plan (Team 4) ECE 58512 Cache Statistics Rep ortingOb jectives‹Verify Tracking of Cache Reads and Writes:Ensure that each read andwrite request to the cache is correctly counted, providing an accurate record ofthe op erations for each trace.‹Conrm Cache Hit and Miss Count Accuracy:Validate that cache hitsand misses are tracked accurately, with each cache access correctly up datingthe hit or miss count based on the presence or absence of the requested line.‹Calculate and Rep ort Cache Hit R atio:Ensure that the cache hit ratiois calculated and rep orted accurately after each trace le is executed, based onthe recorded hits and misses.‹Output Statis tics in Silent and Normal Mo des:Conrm that the statis-tics are correctly displayed in b oth silent and normal mo des, ensuring thatsilent mo de only displays a summary, while normal mo de provides detailedlogging.66

Simulated Cache Test Plan (Team 4) ECE 585Test Scenarios and Pro cedures1. Verify Tracking of Cache Reads and Writes1.1 Pro cess Read and Write RequestsPro cedure:1. Execute a trace le that includes a mix of read and write requests to the cache.2. Track the numb er of read and write op erations p erformed on the cache.3. At the end of the trace, compare the tracked counts with the exp ected valuesbased on the trace le content.Exp ected Outcome:The counts for cache reads and writes should match the numb er of read and writerequests in the trace le, conrming accurate op eration tracking.2. Conrm Cache Hit and Miss Count Accuracy2.1 Track Hits and Misses During Cache AccessPro cedure:1. Execute a trace le containing multiple requests, with some resulting in cachehits and others in misses.2. For each request, log whether it results in a cache hit or miss.3. At the end of the trace, verify that the hit and miss counts match the exp ectedoutcomes based on the cache's initial state and the sequence of accesses.Exp ected Outcome:The cache hit and miss counts should b e accurately rep orted, matching the exp ectedresults based on the trace le and cache b ehavior.67

Simulated Cache Test Plan (Team 4) ECE 5853. Calculate and Rep ort Cache Hit Ratio3.1 Compute and Display Cache H it RatioPro cedure:1. Run a trace le and record the total numb er of cache hits and misses.2. Calculate the hit ratio as:hit ratio =numb er of hitsnumb er of hits + numb er of misses3. Conrm that the displayed hit ratio at the end of the trace is accurate, basedon the recorded hits and miss es .Exp ected Outcome:The displayed hit ratio should match the calculated value, ensuring that the sys-tem correctly tracks and rep orts cache p erformance.68

Simulated Cache Test Plan (Team 4) ECE 5854. Output Statistics in Silent and Normal Mo des4.1 Verify Output in Silent Mo dePro cedure:1. Run a trace le in silent mo de.2. Conrm that the only output displayed is the summary of cache statistics:total reads, writes , hits, misses, and hit ratio.Exp ected Outcome:Silent mo de should display a concise summary of the cache statistics without anydetailed op eration logs, ensuring that it only provides the necessary summary infor-mation.4.2 Verify Output in Normal Mo dePro cedure:1. Run the same trace le in normal mo de.2. Conrm that, in addition to the summary statistics, the output includes de-tailed logs of each cache op eration, including bus op erations, sno op results, andmessages to the higher-level cache.Exp ected Outcome:Normal mo de should provide detailed op eration logs alongside the s ummary statis-tics, allowing for deep er analysis of cache b ehavior during execution.69

Simulated Cache Test Plan (Team 4) ECE 585Exp ected Results SummaryEach test scenario should conrm that:‹Read and Write Tracking:The cache correctly tracks the total numb er ofread and write op erations p erformed during the execution of the trace le.‹Hit and Miss Count Accuracy:The cache hit and miss counts are correctlyrep orted and reect the actual numb er of hits and misses based on the accesspatterns in the trace le.‹Correct Hit Ratio Calculation:The cache correctly calculates and rep ortsthe hit ratio based on the total numb er of hits and misses.‹Correct Output in Silent and Normal Mo des:Silent mo de outputs onlya summary of statistics, while normal mo de provides a detailed log of all cacheop erations.70