



MESI\_Two\_Level-dir.sm: action(a\_sendAck, "a", desc="Send ack to L2") {

MESI\_Two\_Level-dir.sm: action(d\_sendData, "d", desc="Send data to requestor") {

MESI\_Two\_Level-dir.sm: action(aa\_sendAck, "aa", desc="Send ack to L2") {

MESI\_Two\_Level-dir.sm: action(j\_popIncomingRequestQueue, "j", desc="Pop incoming request queue") {

MESI\_Two\_Level-dir.sm: action(k\_popIncomingResponseQueue, "k", desc="Pop incoming request queue") {

MESI\_Two\_Level-dir.sm: action(l\_popMemQueue, "q", desc="Pop off-chip request queue") {

MESI\_Two\_Level-dir.sm: action(kd\_wakeUpDependents, "kd", desc="wake-up dependents") {

MESI\_Two\_Level-dir.sm: action(qf\_queueMemoryFetchRequest, "qf", desc="Queue off-chip fetch request") {

MESI\_Two\_Level-dir.sm: action(qw\_queueMemoryWBRequest, "qw", desc="Queue off-chip writeback request") {

MESI\_Two\_Level-dir.sm: action(qf\_queueMemoryFetchRequestDMA, "qfd", desc="Queue off-chip fetch request") {

MESI\_Two\_Level-dir.sm: action(p\_popIncomingDMARequestQueue, "p", desc="Pop incoming DMA queue") {

MESI\_Two\_Level-dir.sm: action(dr\_sendDMAData, "dr", desc="Send Data to DMA controller from directory") {

MESI\_Two\_Level-dir.sm: action(qw\_queueMemoryWBRequest\_partial, "qwp",

MESI\_Two\_Level-dir.sm: action(da\_sendDMAAck, "da", desc="Send Ack to DMA controller") {

MESI\_Two\_Level-dir.sm: action(z\_stallAndWaitRequest, "z", desc="recycle request queue") {

MESI\_Two\_Level-dir.sm: action(zz\_recycleDMAQueue, "zz", desc="recycle DMA queue") {

MESI\_Two\_Level-dir.sm: action(inv\_sendCacheInvalidate, "inv", desc="Invalidate a cache block") {

MESI\_Two\_Level-dir.sm: action(drp\_sendDMAData, "drp", desc="Send Data to DMA controller from incoming PUTX") {

MESI\_Two\_Level-dir.sm: action(v\_allocateTBE, "v", desc="Allocate TBE") {

MESI\_Two\_Level-dir.sm: action(qw\_queueMemoryWBRequest\_partialTBE, "qwt",

MESI\_Two\_Level-dir.sm: action(w\_deallocateTBE, "w", desc="Deallocate TBE") {

MESI\_Two\_Level-dma.sm: action(s\_sendReadRequest, "s", desc="Send a DMA read request to memory") {

MESI\_Two\_Level-dma.sm: action(s\_sendWriteRequest, "\s", desc="Send a DMA write request to memory") {

MESI\_Two\_Level-dma.sm: action(a\_ackCallback, "a", desc="Notify dma controller that write request completed") {

MESI\_Two\_Level-dma.sm: action(d\_dataCallback, "d", desc="Write data to dma sequencer") {

MESI\_Two\_Level-dma.sm: action(t\_updateTBEData, "t", desc="Update TBE Data") {

MESI\_Two\_Level-dma.sm: action(v\_allocateTBE, "v", desc="Allocate TBE entry") {

MESI\_Two\_Level-dma.sm: action(w\_deallocateTBE, "w", desc="Deallocate TBE entry") {

MESI\_Two\_Level-dma.sm: action(p\_popRequestQueue, "p", desc="Pop request queue") {

MESI\_Two\_Level-dma.sm: action(p\_popResponseQueue, "\p", desc="Pop request queue") {

MESI\_Two\_Level-dma.sm: action(zz\_stallAndWaitRequestQueue, "zz", desc="...") {

MESI\_Two\_Level-dma.sm: action(wkad\_wakeUpAllDependents, "wkad", desc="wake-up all dependents") {

MESI\_Two\_Level-L1cache.sm: action(a\_issueGETS, "a", desc="Issue GETS") {

MESI\_Two\_Level-L1cache.sm: action(pa\_issuePfGETS, "pa", desc="Issue prefetch GETS") {

MESI\_Two\_Level-L1cache.sm: action(ai\_issueGETINSTR, "ai", desc="Issue GETINSTR") {

MESI\_Two\_Level-L1cache.sm: action(pai\_issuePfGETINSTR, "pai",desc="Issue GETINSTR for prefetch request")

MESI\_Two\_Level-L1cache.sm: action(b\_issueGETX, "b", desc="Issue GETX") {

MESI\_Two\_Level-L1cache.sm: action(pb\_issuePfGETX, "pb", desc="Issue prefetch GETX") {

MESI\_Two\_Level-L1cache.sm: action(c\_issueUPGRADE, "c", desc="Issue GETX") {

MESI\_Two\_Level-L1cache.sm: action(d\_sendDataToRequestor, "d", desc="send data to requestor") {

MESI\_Two\_Level-L1cache.sm: action(d2\_sendDataToL2, "d2", desc="send data to the L2 cache because of M downgrade") {

MESI\_Two\_Level-L1cache.sm: action(dt\_sendDataToRequestor\_fromTBE, "dt", desc="send data to requestor") {

MESI\_Two\_Level-L1cache.sm: action(d2t\_sendDataToL2\_fromTBE, "d2t", desc="send data to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(e\_sendAckToRequestor, "e", desc="send invalidate ack to requestor (could be L2 or L1)") {

MESI\_Two\_Level-L1cache.sm: action(f\_sendDataToL2, "f", desc="send data to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(ft\_sendDataToL2\_fromTBE, "ft", desc="send data to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(fi\_sendInvAck, "fi", desc="send data to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(forward\_eviction\_to\_cpu, "\cc", desc="sends eviction information to the processor") {

MESI\_Two\_Level-L1cache.sm: action(g\_issuePUTX, "g", desc="send data to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(j\_sendUnblock, "j", desc="send unblock to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(jj\_sendExclusiveUnblock, "\j", desc="send unblock to the L2 cache") {

MESI\_Two\_Level-L1cache.sm: action(dg\_invalidate\_sc, "dg",

MESI\_Two\_Level-L1cache.sm: action(h\_load\_hit, "hd",

MESI\_Two\_Level-L1cache.sm: action(h\_ifetch\_hit, "hi", desc="Notify sequencer the instruction fetch completed.")

MESI\_Two\_Level-L1cache.sm: action(hx\_load\_hit, "hx", desc="Notify sequencer the load completed.")

MESI\_Two\_Level-L1cache.sm: action(hh\_store\_hit, "\h", desc="Notify sequencer that store completed.")

MESI\_Two\_Level-L1cache.sm: action(hhx\_store\_hit, "\hx", desc="Notify sequencer that store completed.")

MESI\_Two\_Level-L1cache.sm: action(i\_allocateTBE, "i", desc="Allocate TBE (isPrefetch=0, number of invalidates=0)") {

MESI\_Two\_Level-L1cache.sm: action(k\_popMandatoryQueue, "k", desc="Pop mandatory queue.") {

MESI\_Two\_Level-L1cache.sm: action(l\_popRequestQueue, "l",

MESI\_Two\_Level-L1cache.sm: action(o\_popIncomingResponseQueue, "o",

MESI\_Two\_Level-L1cache.sm: action(s\_deallocateTBE, "s", desc="Deallocate TBE") {

MESI\_Two\_Level-L1cache.sm: action(u\_writeDataToL1Cache, "u", desc="Write data to cache") {

MESI\_Two\_Level-L1cache.sm: action(q\_updateAckCount, "q", desc="Update ack count") {

MESI\_Two\_Level-L1cache.sm: action(ff\_deallocateL1CacheBlock, "\f", desc="Deallocate L1 cache block. Sets the cache to not present, allowing a replacement in parallel with a fetch.") {

MESI\_Two\_Level-L1cache.sm: action(oo\_allocateL1DCacheBlock, "\o", desc="Set L1 D-cache tag equal to tag of block B.") {

MESI\_Two\_Level-L1cache.sm: action(pp\_allocateL1ICacheBlock, "\p", desc="Set L1 I-cache tag equal to tag of block B.") {

MESI\_Two\_Level-L1cache.sm: action(z\_stallAndWaitMandatoryQueue, "\z", desc="recycle L1 request queue") {

MESI\_Two\_Level-L1cache.sm: action(kd\_wakeUpDependents, "kd", desc="wake-up dependents") {

MESI\_Two\_Level-L1cache.sm: action(uu\_profileInstMiss, "\uim", desc="Profile the demand miss") {

MESI\_Two\_Level-L1cache.sm: action(uu\_profileInstHit, "\uih", desc="Profile the demand hit") {

MESI\_Two\_Level-L1cache.sm: action(uu\_profileDataMiss, "\udm", desc="Profile the demand miss") {

MESI\_Two\_Level-L1cache.sm: action(uu\_profileDataHit, "\udh", desc="Profile the demand hit") {

MESI\_Two\_Level-L1cache.sm: action(po\_observeMiss, "\po", desc="Inform the prefetcher about the miss") {

MESI\_Two\_Level-L1cache.sm: action(ppm\_observePfMiss, "\ppm",

MESI\_Two\_Level-L1cache.sm: action(pq\_popPrefetchQueue, "\pq", desc="Pop the prefetch request queue") {

MESI\_Two\_Level-L1cache.sm: action(mp\_markPrefetched, "mp", desc="Write data from response queue to cache") {

MESI\_Two\_Level-L2cache.sm: action(a\_issueFetchToMemory, "a", desc="fetch data from memory") {

MESI\_Two\_Level-L2cache.sm: action(b\_forwardRequestToExclusive, "b", desc="Forward request to the exclusive L1") {

MESI\_Two\_Level-L2cache.sm: action(c\_exclusiveReplacement, "c", desc="Send data to memory") {

MESI\_Two\_Level-L2cache.sm: action(c\_exclusiveCleanReplacement, "cc", desc="Send ack to memory for clean replacement") {

MESI\_Two\_Level-L2cache.sm: action(ct\_exclusiveReplacementFromTBE, "ct", desc="Send data to memory") {

MESI\_Two\_Level-L2cache.sm: action(d\_sendDataToRequestor, "d", desc="Send data from cache to reqeustor") {

MESI\_Two\_Level-L2cache.sm: action(dd\_sendExclusiveDataToRequestor, "dd", desc="Send data from cache to reqeustor") {

MESI\_Two\_Level-L2cache.sm: action(ds\_sendSharedDataToRequestor, "ds", desc="Send data from cache to reqeustor") {

MESI\_Two\_Level-L2cache.sm: action(e\_sendDataToGetSRequestors, "e", desc="Send data from cache to all GetS IDs") {

MESI\_Two\_Level-L2cache.sm: action(ex\_sendExclusiveDataToGetSRequestors, "ex", desc="Send data from cache to all GetS IDs") {

MESI\_Two\_Level-L2cache.sm: action(ee\_sendDataToGetXRequestor, "ee", desc="Send data from cache to GetX ID") {

MESI\_Two\_Level-L2cache.sm: action(f\_sendInvToSharers, "f", desc="invalidate sharers for L2 replacement") {

MESI\_Two\_Level-L2cache.sm: action(fw\_sendFwdInvToSharers, "fw", desc="invalidate sharers for request") {

MESI\_Two\_Level-L2cache.sm: action(fwm\_sendFwdInvToSharersMinusRequestor, "fwm", desc="invalidate sharers for request, requestor is sharer") {

MESI\_Two\_Level-L2cache.sm: action(i\_allocateTBE, "i", desc="Allocate TBE for request") {

MESI\_Two\_Level-L2cache.sm: action(s\_deallocateTBE, "s", desc="Deallocate external TBE") {

MESI\_Two\_Level-L2cache.sm: action(jj\_popL1RequestQueue, "\j", desc="Pop incoming L1 request queue") {

MESI\_Two\_Level-L2cache.sm: action(k\_popUnblockQueue, "k", desc="Pop incoming unblock queue") {

MESI\_Two\_Level-L2cache.sm: action(o\_popIncomingResponseQueue, "o", desc="Pop Incoming Response queue") {

MESI\_Two\_Level-L2cache.sm: action(m\_writeDataToCache, "m", desc="Write data from response queue to cache") {

MESI\_Two\_Level-L2cache.sm: action(mr\_writeDataToCacheFromRequest, "mr", desc="Write data from response queue to cache") {

MESI\_Two\_Level-L2cache.sm: action(q\_updateAck, "q", desc="update pending ack count") {

MESI\_Two\_Level-L2cache.sm: action(qq\_writeDataToTBE, "\qq", desc="Write data from response queue to TBE") {

MESI\_Two\_Level-L2cache.sm: action(ss\_recordGetSL1ID, "\s", desc="Record L1 GetS for load response") {

MESI\_Two\_Level-L2cache.sm: action(xx\_recordGetXL1ID, "\x", desc="Record L1 GetX for store response") {

MESI\_Two\_Level-L2cache.sm: action(set\_setMRU, "\set", desc="set the MRU entry") {

MESI\_Two\_Level-L2cache.sm: action(qq\_allocateL2CacheBlock, "\q", desc="Set L2 cache tag equal to tag of block B.") {

MESI\_Two\_Level-L2cache.sm: action(rr\_deallocateL2CacheBlock, "\r", desc="Deallocate L2 cache block. Sets the cache to not present, allowing a replacement in parallel with a fetch.") {

MESI\_Two\_Level-L2cache.sm: action(t\_sendWBAck, "t", desc="Send writeback ACK") {

MESI\_Two\_Level-L2cache.sm: action(ts\_sendInvAckToUpgrader, "ts", desc="Send ACK to upgrader") {

MESI\_Two\_Level-L2cache.sm: action(uu\_profileMiss, "\um", desc="Profile the demand miss") {

MESI\_Two\_Level-L2cache.sm: action(uu\_profileHit, "\uh", desc="Profile the demand hit") {

MESI\_Two\_Level-L2cache.sm: action(nn\_addSharer, "\n", desc="Add L1 sharer to list") {

MESI\_Two\_Level-L2cache.sm: action(nnu\_addSharerFromUnblock, "\nu", desc="Add L1 sharer to list") {

MESI\_Two\_Level-L2cache.sm: action(kk\_removeRequestSharer, "\k", desc="Remove L1 Request sharer from list") {

MESI\_Two\_Level-L2cache.sm: action(ll\_clearSharers, "\l", desc="Remove all L1 sharers from list") {

MESI\_Two\_Level-L2cache.sm: action(mm\_markExclusive, "\m", desc="set the exclusive owner") {

MESI\_Two\_Level-L2cache.sm: action(mmu\_markExclusiveFromUnblock, "\mu", desc="set the exclusive owner") {

MESI\_Two\_Level-L2cache.sm: action(zz\_stallAndWaitL1RequestQueue, "zz", desc="recycle L1 request queue") {

MESI\_Two\_Level-L2cache.sm: action(zn\_recycleResponseNetwork, "zn", desc="recycle memory request") {

MESI\_Two\_Level-L2cache.sm: action(kd\_wakeUpDependents, "kd", desc="wake-up dependents") {