Memory Hierarchy and Coherence

ESESC Tutorial

Speakers: Gabriel Southern

Department of Computer Engineering,
University of California, Santa Cruz
http://masc.soe.ucsc.edu

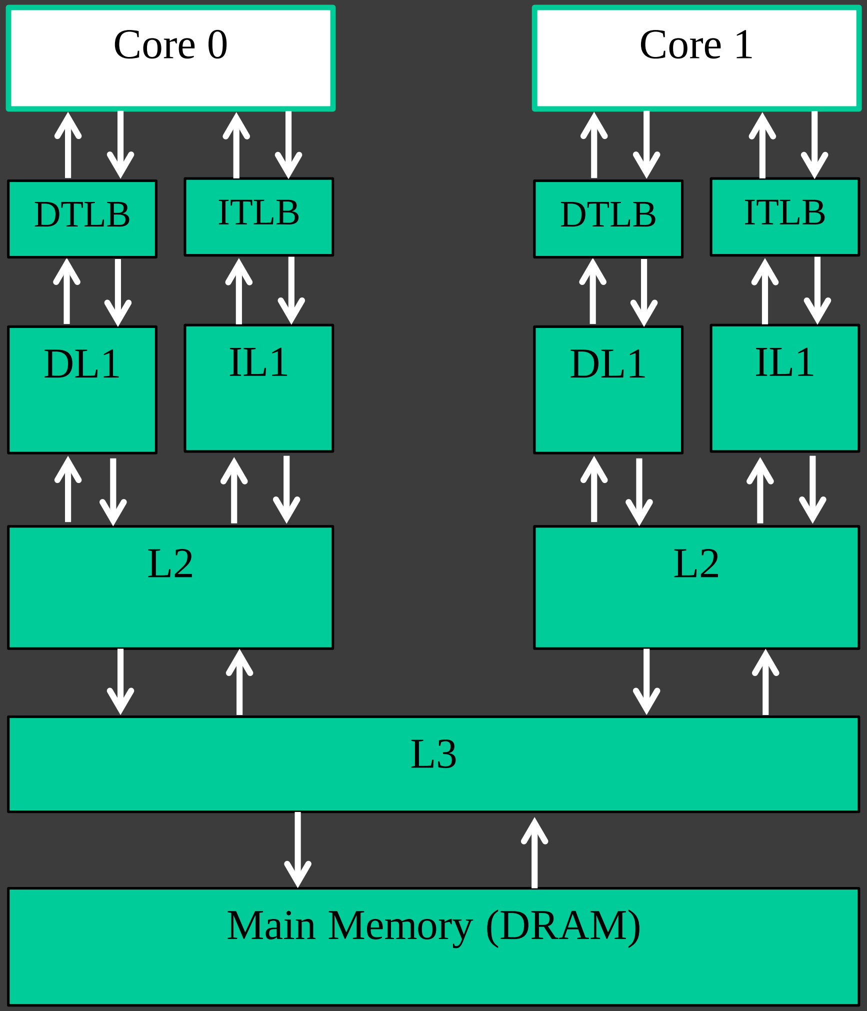
Introduction

- You will learn:
  - Memory hierarchy and cache coherence

Outline

- Introduction
- Configuration Options
- Demo: Change Configuration
- Life of a Memory Request
- Demo: Modify L2 cache behavior

Sample Configuration



Core Configuration Options

Setting	Example	Purpose
`stForwardDelay`	`3`	load store forwarding delay
`maxLoads`	`48`	load queue size
`maxStores`	`32`	store queue size
`DL1`	`"PerCore_DTLB PTLB"`	L1 data cache setting
`IL1`	`"PerCore_ITLB ITLB"`	L1 instruction cache setting
`MemoryReplay`	`false`	disable memory replay
`enableICache`	`true`	enable/disable instruction cache
`enableDCache`	`true`	enable/disable data cache
`noMemSpec`	`false`	disable memory speculation
`StoreSetSize`	`8192`	store set size

Cache Configuration Options (1)

Setting	Example	Purpose
`deviceType`	`'cache'`	for initialization
`inclusive`	`true`	force inclusion (exclusive not supported)
`directory`	`false`	use directory coherence
`blockName`	`"dcache"`	used for floorplan name in thermal
`numBanks`	`1`	number of banks
`maxRequests`	`16`	max requests in flight
`size`	`32*1024`	size in bytes
`assoc`	`8`	associativity
`skew`	`false`	model skew cache

Cache Configuration Options (2)

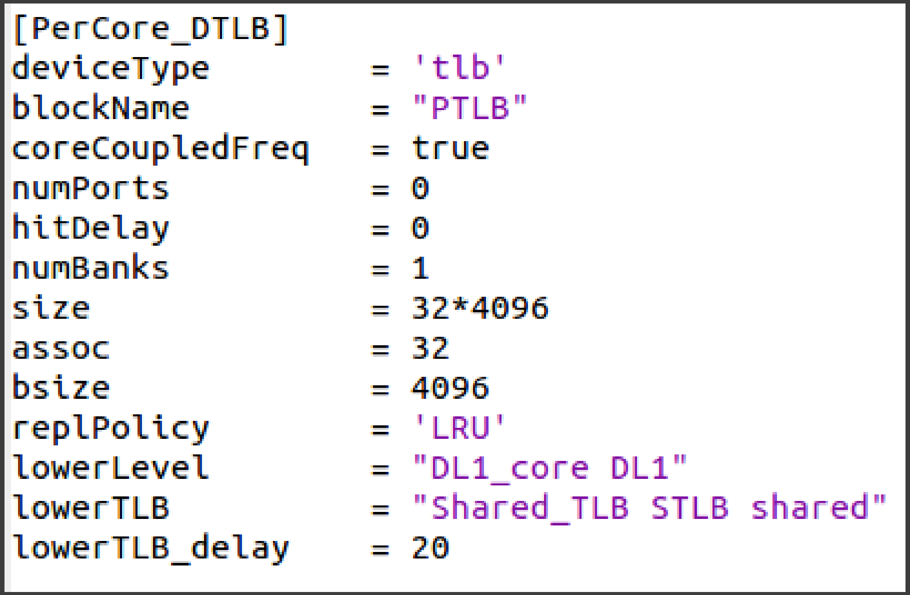
Setting	Example	Purpose
`bsize`	`64`	cache block size in bytes
`replPolicy`	`'LRU'`	replacement policy
`bkNumPorts`	`1`	for modeling bandwidth
`bkPortOccp`	`1`	for modeling bandwidth
`hitDelay`	`3`	hit time in cycles
`missDelay`	`2`	miss time + next level hit/miss
`MSHR`	`"DL1_MSHR"`	MSHR configuration section
`lowerLevel`	`"PrivL2 L2 sharedby 1"`	next level in cache hierarchy
`fillBuffSize`	`4`	for power model
`pfetchBuffSize`	`16`	for power model
`wbBuffSize`	`16`	for power model

Device Type Option

Option	Object	Notes
cache	CCache	Cache structure
nicecache	NICECache	Lowest level of memory hierarchy
stridePrefetcher	StridePrefetcher	Under development
markovPrefetcher	MarkovPrefetcher	Under development
taggedPrefetcher	TaggedPrefetcher	Under development
bus	Bus	Bus interconnect
tlb	TLB	TLB modeled using cache
splitter	Splitter	
memxbar	MemXBar	
unmemxbar	UnMemXBar	
memcontroller	MemController	Model DRAM
void	-	Stops traversing memory objects

Translation Lookaside Buffer (TLB)

- Model delay not address translation
  - Use cache with 4096 byte lines
  - Fully associative
- Simulates three level page walk
- Virtually indexed / physically checked
- Fixed delay for L2 TLB access



```
[PerCore_DTLB]
deviceType          = 'tlb'
blockName           = "PTLB"
coreCoupledFreq     = true
numPorts            = 0
hitDelay            = 0
numBanks            = 1
size                = 32*4096
assoc               = 32
bsize               = 4096
replPolicy          = 'LRU'
lowerLevel          = "DL1_core DL1"
lowerTLB            = "Shared_TLB STLB shared"
lowerTLB_delay      = 20
```

Miss Status Handling Register (MSHR)

Setting	Example	Purpose
`type`	`"full"`	Type of MSHR
`size`	`16`	Number of entries
`nSubEntries`	`16`	Number of accesses to shared MSHR entry

Type Option	
`full`	**tracks dependencies between addresses**
`blocking`	not currently supported with coherence
`none`	not currently supported with coherence

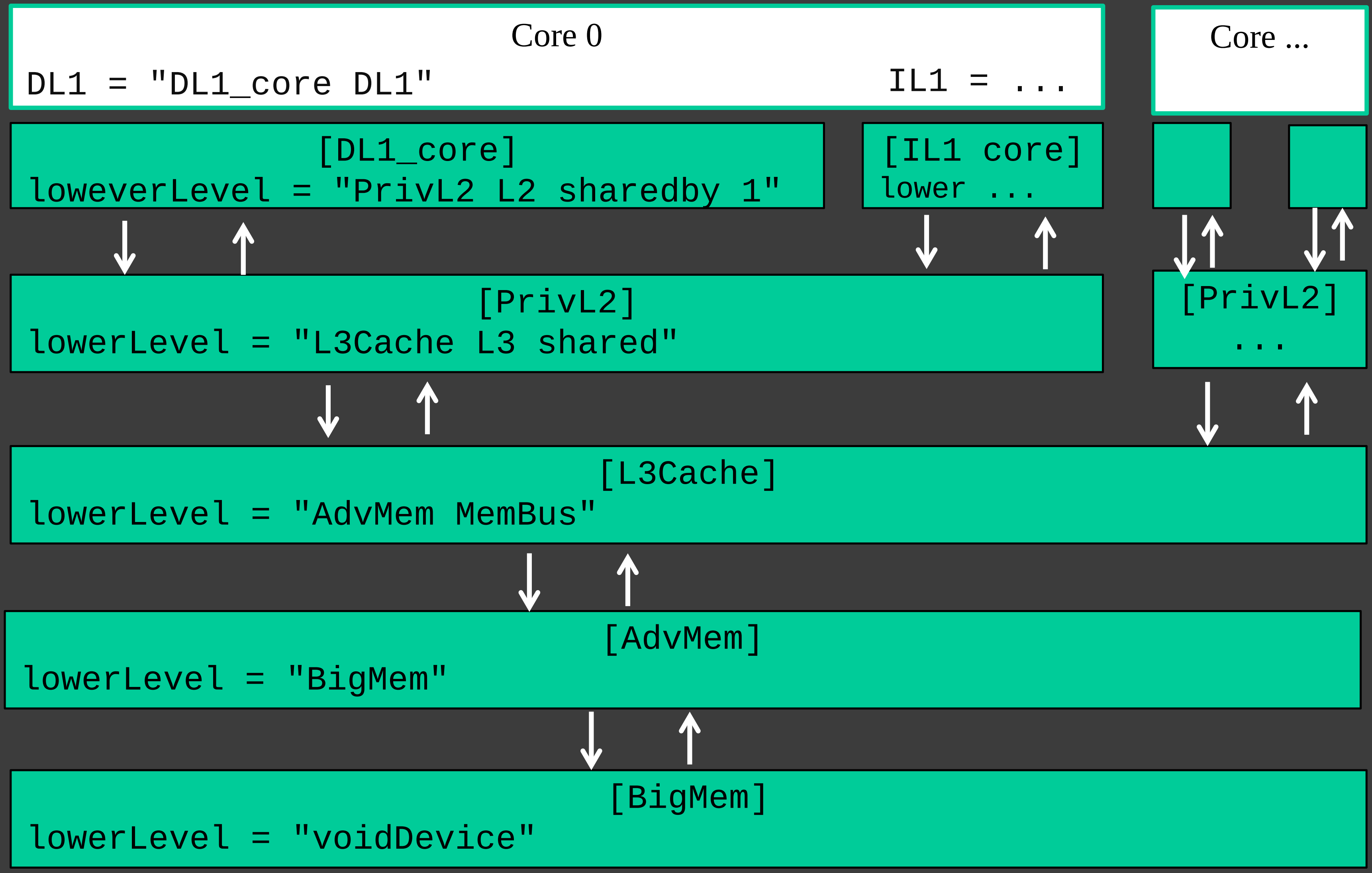
Memory Controller

- Models DRAM access delay
- Implements FR-FCFS (First-Ready First-Come-First-Served) algorithm

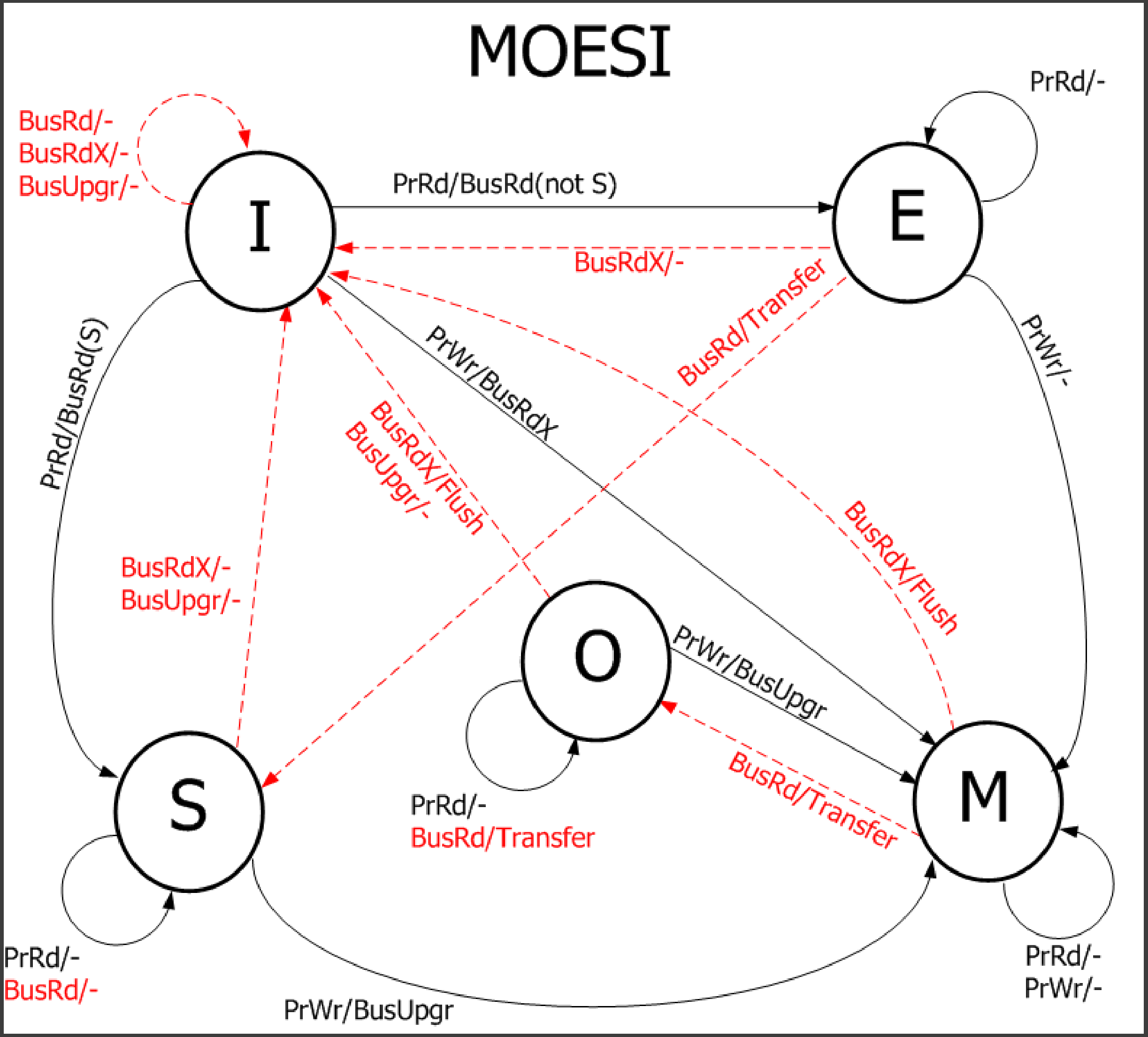
Memory Controller Settings

Setting	Example	Notes
deviceType	'memcontroller'	type is memory controller
busWidth	64	width in bits
numPorts	1	should always be 1
portOccp	11	model memory bandwidth
delay	3	delay in addition to other componets
NumBanks	256	number of DRAM banks
NumRows	8192	number of DRAM rows per bank
NumColumns	1024	number of DRAM columns per bank
ColumnSize	256	data bits per column
PreChargeLatency	52	percharge latency in cycles
RowAccessLatency	52	row access latency in cycles
ColumnAccessLatency	52	column access latency in cycles
memReqestBufferSize	32	max buffered requests

Configuring Cache Hierarchy



MOESI Cache Coherence

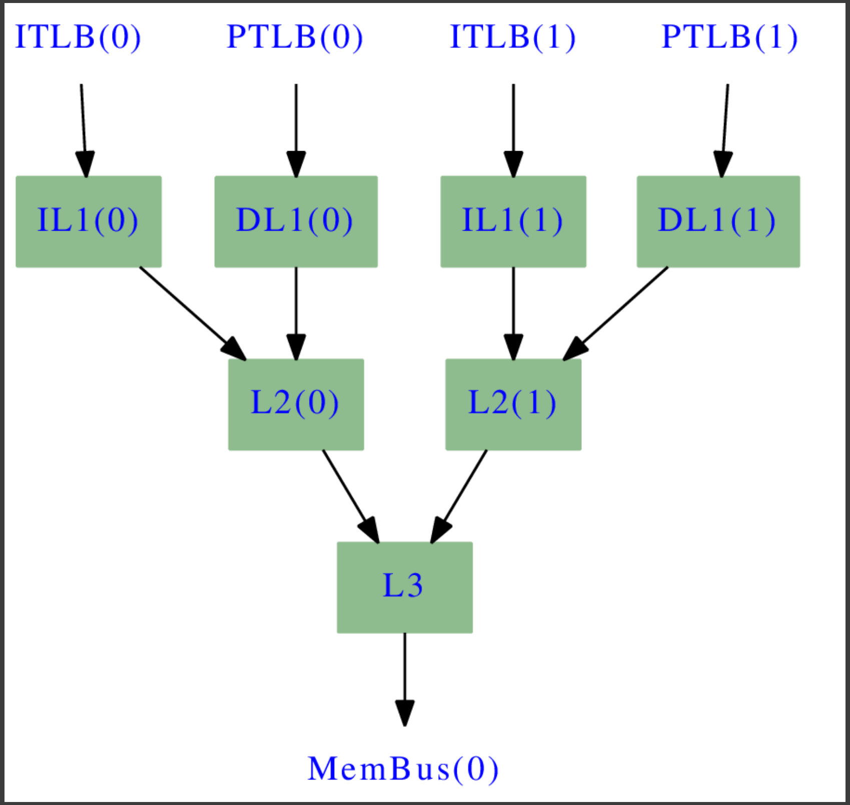


Demo: Configure Cache Size

- Make change to configuration and rerun a benchmark

Demo: View Cache Configuration

- `memory-arch.dot` file created when ESESC initialized
- Use graphviz to convert to graphical format

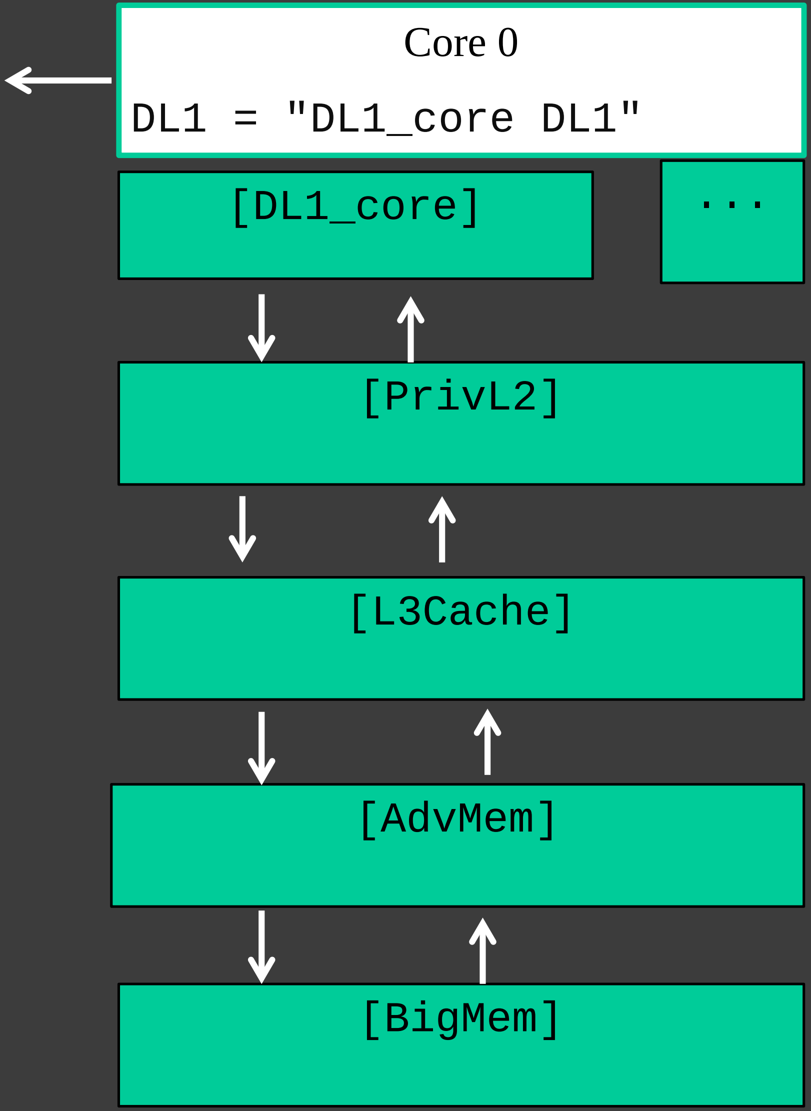


Request Scheduling

- ESESC uses callbacks schedule events
  - callback scheduled in a specific cycle
  - advance global clock and service callbacks

Life of a MemRequest

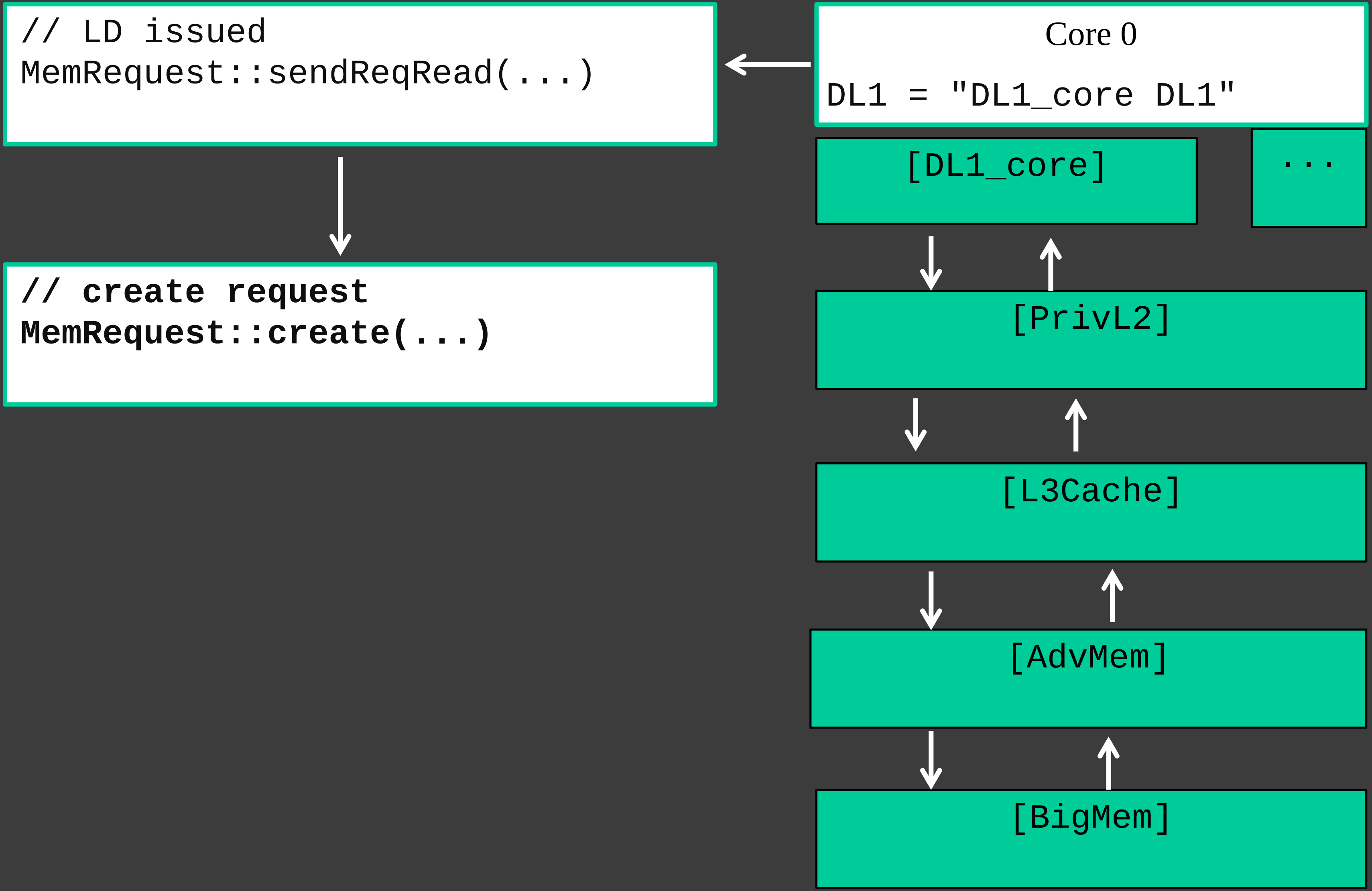
```
// LD issued
MemRequest::sendReqRead(...)
```



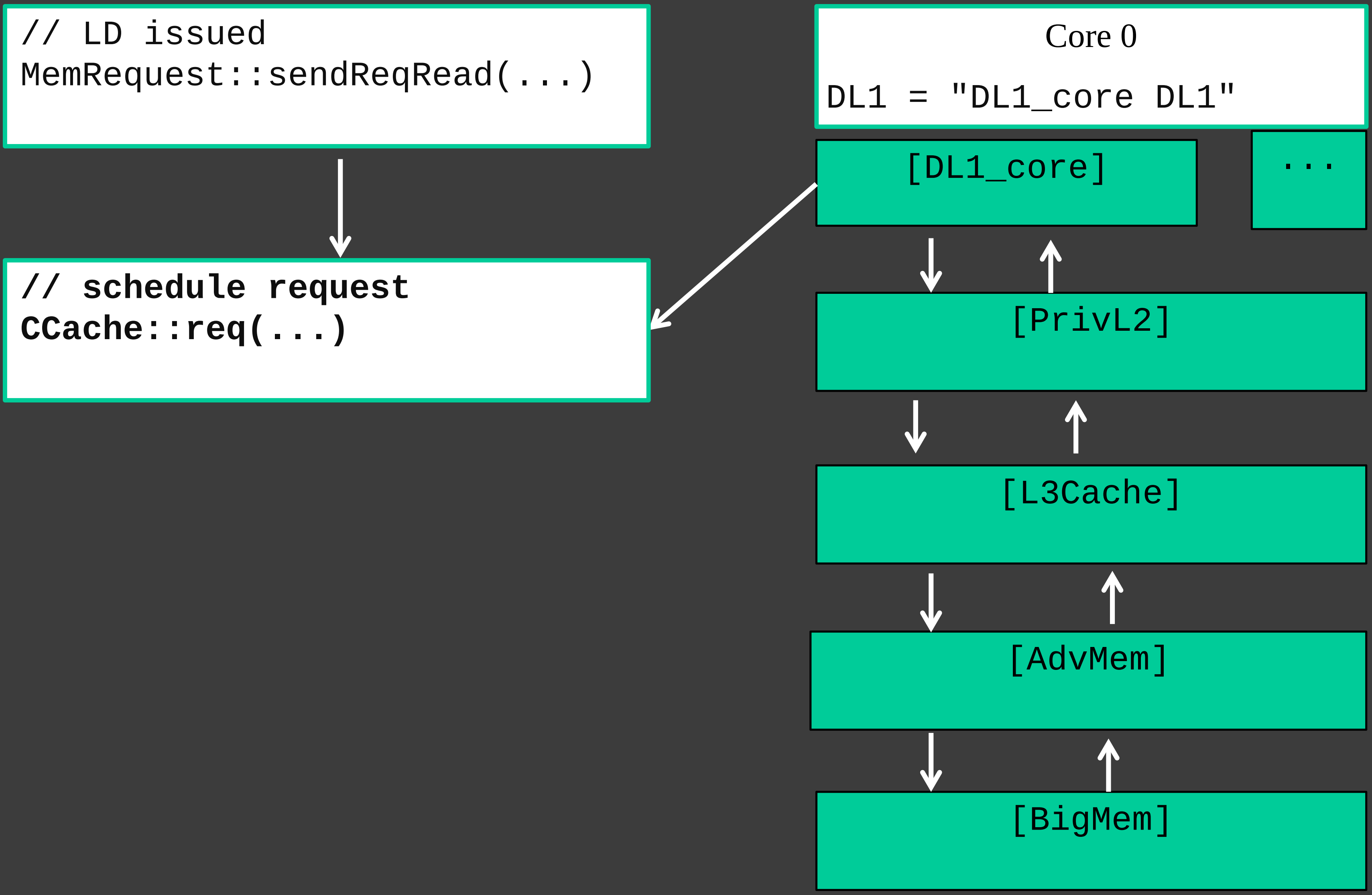
Life of a MemRequest (cont)

```
// LD issued
MemRequest::sendReqRead(...)
```

`// create request`
`MemRequest::create(...)`



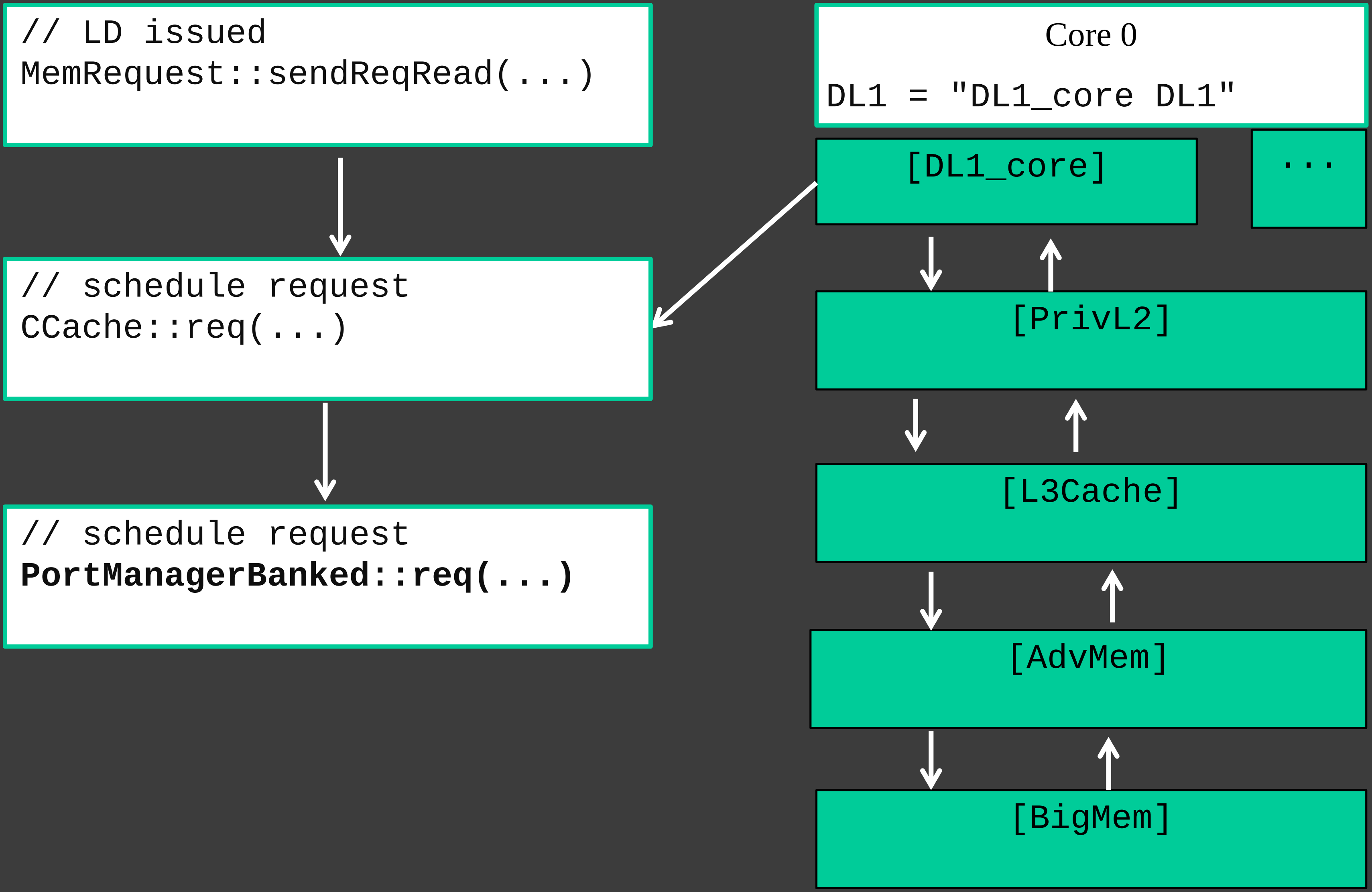
Life of a MemRequest (cont)



```
// LD issued
MemRequest::sendReqRead(...)
```

```
// schedule request
CCache::req(...)
```

Life of a MemRequest (cont)

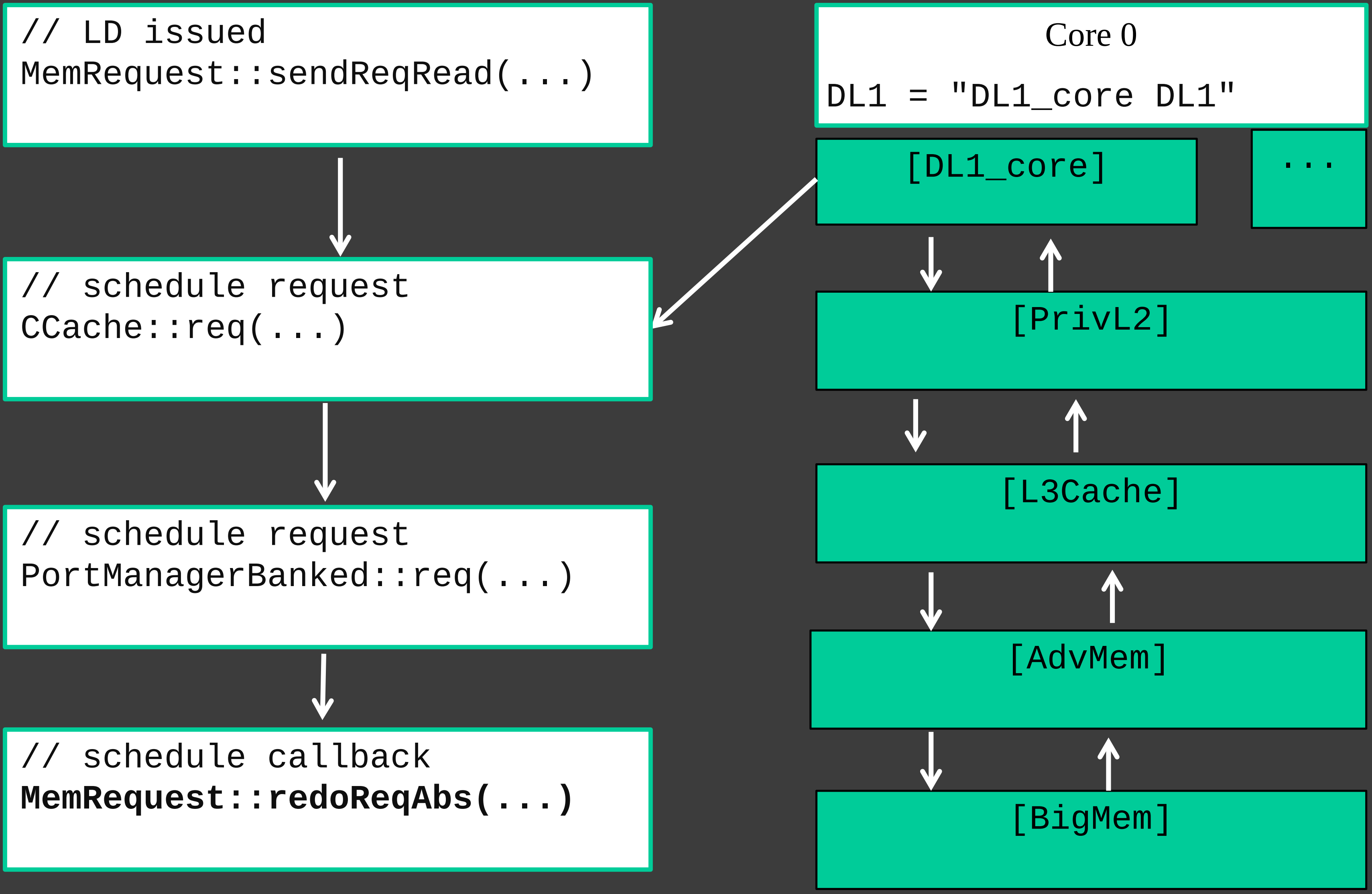


```
// LD issued
MemRequest::sendReqRead(...)
```

```
// schedule request
CCache::req(...)
```

```
// schedule request
PortManagerBanked::req(...)
```

Life of a MemRequest (cont)



Memory Object Interface

Request Interface (Down)

```
void req(MemRequest *req);
void setStateAck(MemRequest *req);
void disp(MemRequest *req);

void doReq(MemRequest *req);
void doSetStateAck(MemRequest *req);
void doDisp(MemRequest *req);
```

Response Interface (Up)

```
void reqAck(MemRequest *req);
void setState(MemRequest *req);

void doReqAck(MemRequest *req);
void doSetState(MemRequest *req);
bool isBusy(AddrType addr) const;
```

Demo: Add module

- Create a module that increases the hit time for odd addresses