
CHAPTER 1 CSL Memory Controller

All rights reserved
Copyright ©2006 Fastpath Logic, Inc.
Copying in any form without the expressed written
permission of Fastpath Logic, Inc is prohibited

1.1 Definitions

TABLE 1.1 Definitions

DMA	Direct Memory Access
SRAM	Static Random Access Memory
DRAM	Dynamic Random Access Memory
bw	band-width
cluster	A group of units

1.2 CSL Memory Controller Overview

TABLE 1.2

Chapter 1.2.1 , “Overview”
Chapter 1.2.2 , “MCU”
Chapter , “”
Chapter , “”
Chapter 1.2.5 , “MC”
Chapter 1.2.6 , “Memory request packet”
Chapter 1.2.7 , “Memory formats”
Chapter 1.2.8 , “Address”
Chapter 1.2.9 , “System level view of MCU+MCC+MRRT+MC.”
Chapter 1.2.10 , “DRAM”
Chapter 1.2.11 , “Requirements”
Chapter 1.2.11 , “Requirements”
Chapter 1.2.13 , “Test”

TABLE 1.2

Chapter 1.2.14 , “Notes”
Chapter 1.2.15 , “Performance analysis”

1.2.1 Overview

Operations

- random read (single rd)
- random write (single write)
- rectangular region read
- rectangular region write

Specify a rectangular region in a surface.

Unit level memory bandwidth requirements during execution.

Each unit on the chip has a certain memory bandwidth requirement. The memory bandwidth requirement is the amount of memory cycles consumed or average during a block transfer:

- max / peak operation
- average operation
- min operation

100% memory utilization is possible. If a client reads a word every other cycle (1 out of 2 cycles) then the client utilizes 50% of the available memory bandwidth. At any given point in the enabled set of units on the chip a demand memory bandwidth of 100% cannot be exceeded.

100% memory bandwidth is equal to the number of cycles / second * memory word size excluding overhead cycles.

Account for memory :

- precharge
- refresh
- other latencies

on chip shared memory

off chip shared memory

Each unit has config regs with address to rd / wr which are configured by SW.

The address rd / wr counters track which addresses to currently read / write.

de sap

Pixel frame, memory, bandwidth requirements.

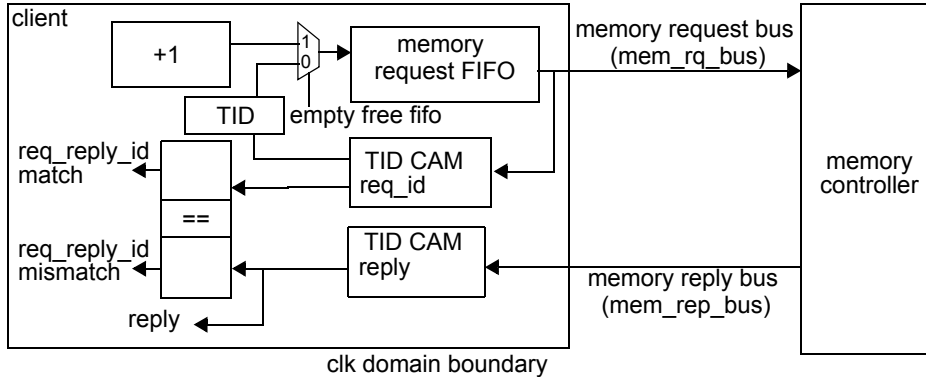
The pixel frame memory bw req vary throughout the frame depending on the frame type and which unit is currently processing the frame.

It turns out that the various units do not have a uniform bandwidth requirement over the entire frame time. Frame access time is the amount of time necessary to transfer the frame from memory to the local unit. So there are times in which memory controller is delivering 70 to 80% efficiency, but then it will drop to 50% because there is not enough bandwidth demand from the clients.

1.2.2 MCU

1.2.2.1 MCU TID logic

FIGURE 1.1 Memory client architecture

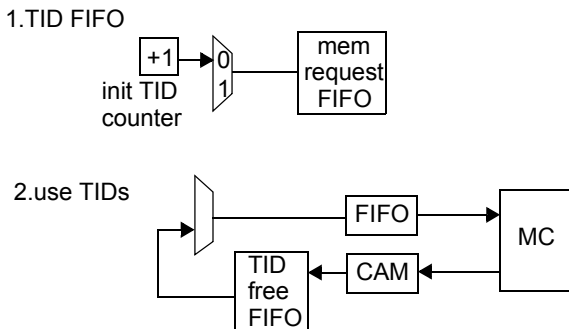


Every request sent by the request FIFO has a request id (req_id) associated with a memory request transaction. Each time a memory request is sent to the memory controller the associated req_id is pushed on the memory req_id FIFO.

Memory request FIFO contains mem reg packets (mrp): {TID, unit_id, wr/rd, data, addr}.

Hold requests which are not high priority (use programmable timer to hold request). Merge requests which are smaller than the memory mem_reg_bus and which have the same address

FIGURE 1.2



For each memory request:

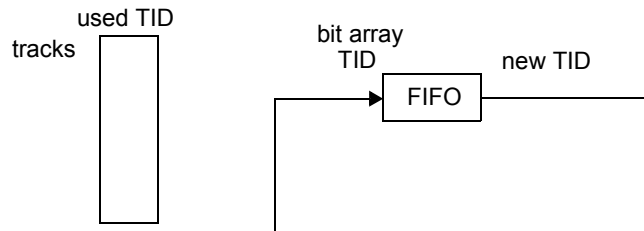
- initialize TIDs in FIFO

- send requests
- use TIDs in free FIFO

Send TIDs until the counter reaches max TID. Max TID is the size of the fifo.

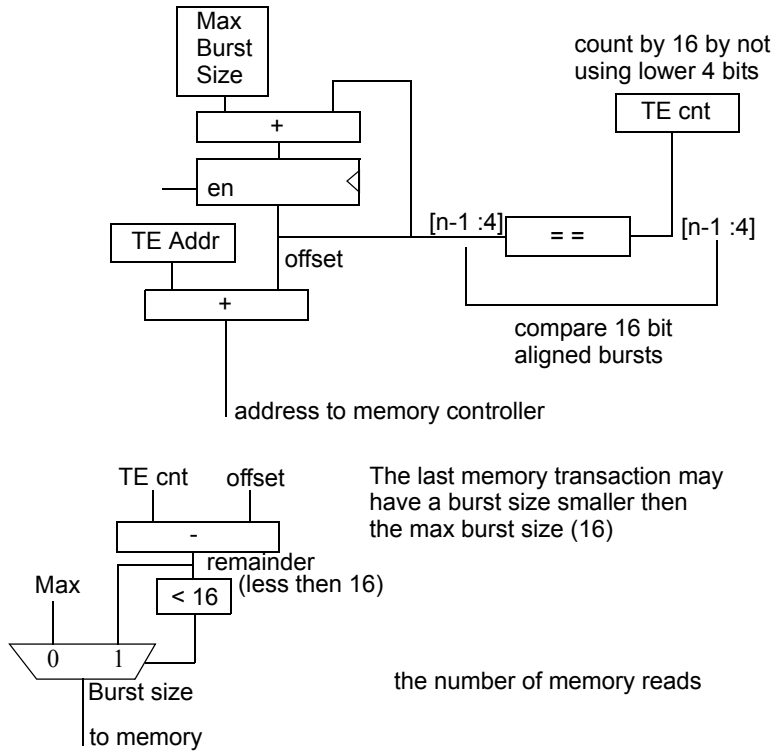
Use CAM to handle out of order transactions. When there is a CAM match remove the entry from the CAM and add it to the free FIFO.

FIGURE 1.3



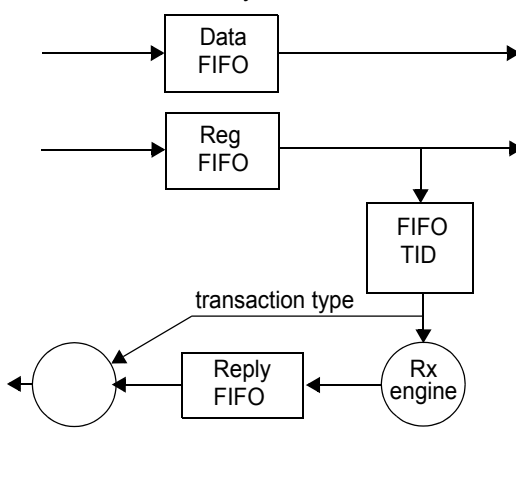
1.2.2.2 MCU DMA

FIGURE 1.4 Burst address generation logic



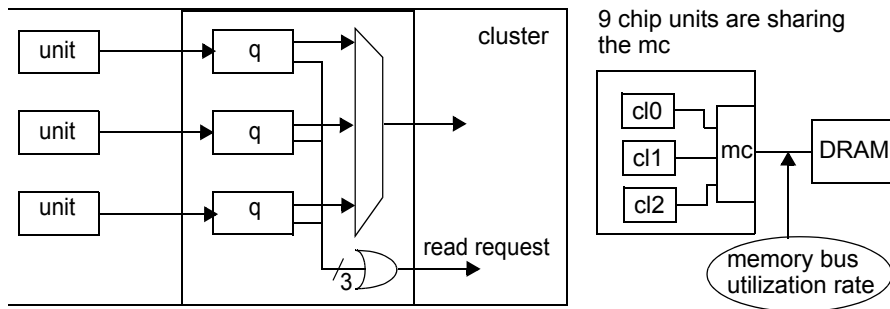
1.2.2.3 MCU TID engine

FIGURE 1.5 Unit memory controller



1.2.3 MCC

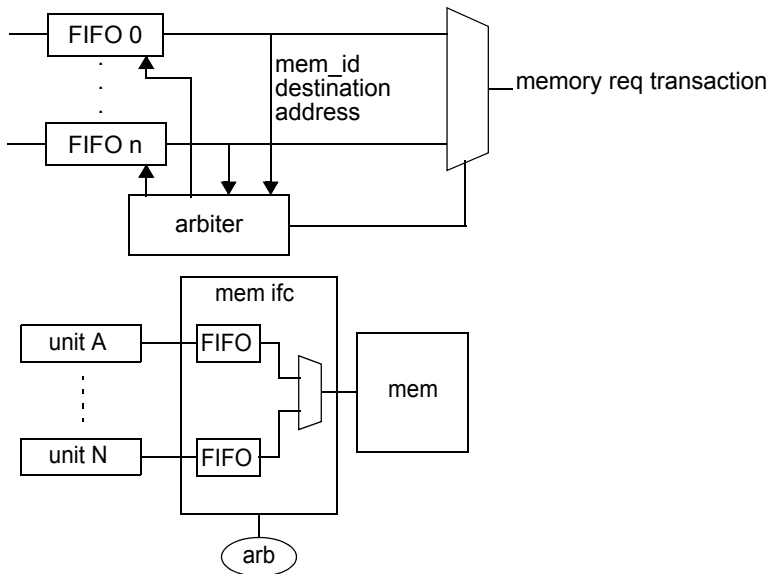
FIGURE 1.6 Cluster memory controller



Group clients which use the same type of operations and have the same bandwidth priority.

1.2.3.1 MCC arbiter

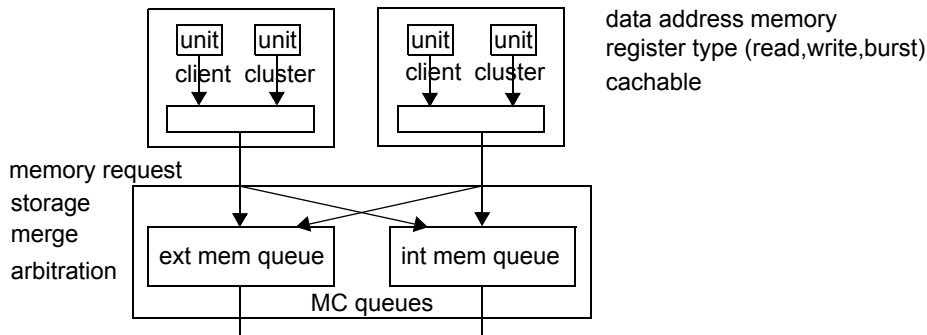
FIGURE 1.7 Cluster level memory controller arbitration logic



1.2.4 MRRT

Request/reply ? the MRRT is hierarchical. Clients are grouped together.

FIGURE 1.8 Memory clients



The intermediate MRRT nodes perform the following operations:

- accept new client transactions
- sort and store the new clients transactions in a queue for each client
- select the next memory transaction from the client queues (arbitration)
- form and MRRT transaction packet
- sends the MRRT packet to the MC

1.2.4.1 Memory request transaction packet format

Each of the memory clients sends the main memory controller a memory transaction request. The memory transaction request contains a destination bit which directs the transaction to either the internal memory controller (0) or external memory controller (1).

Dependency graph for client FIFO of regs for a bank.

TABLE 1.3 Memory request transaction packet format

Request_id	Memory controller
0	internal
1	external

1.2.5 MC

A memory controller (MC) accepts memory requests from clients (leaf level units) on a chip, sorts the requests into queues, selects the next request(s), converts the requests into memory bus transactions, returns any read data to the clients and manages any memory maintenance operations such as DRAM RAS/CAS. The MC can interface to different types of memory such as flash, SRAM and DRAM.

On chip memory controllers receives requests from clients. Since there may be many clients on a chip, the clients are connected to a memory.

FIGURE 1.9 Memory controller and memories

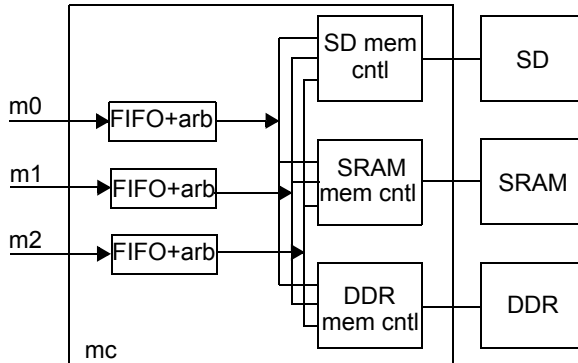
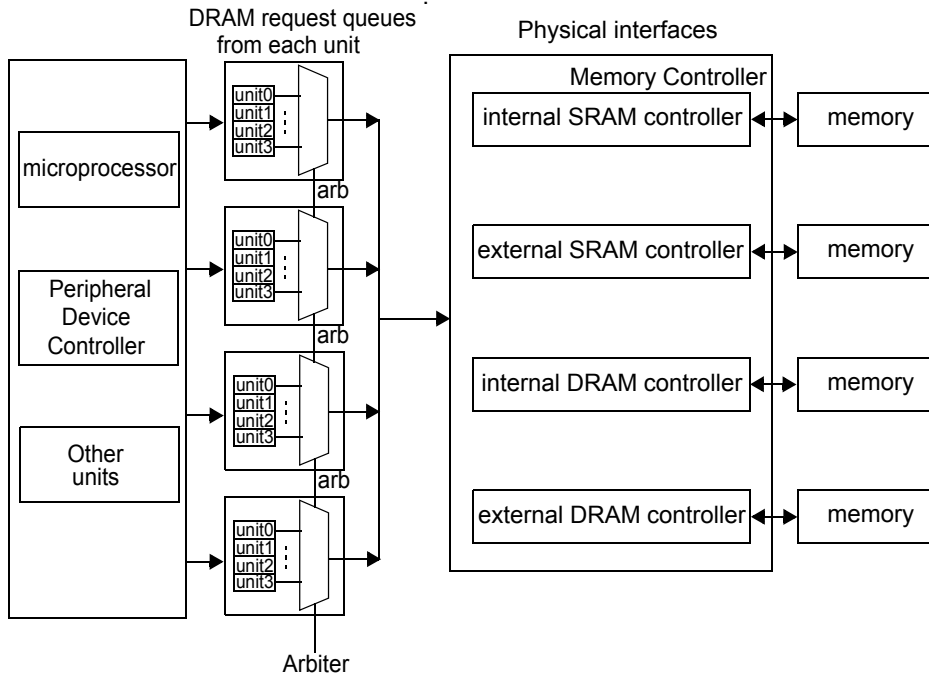
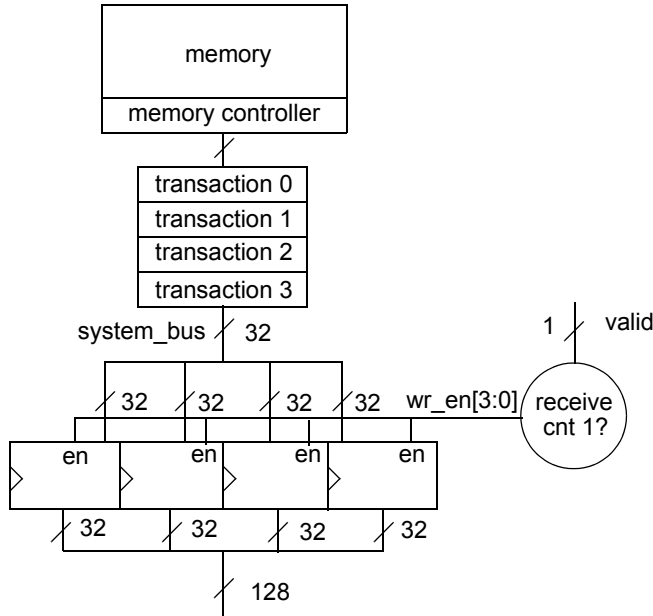


FIGURE 1.10 Memory controllers

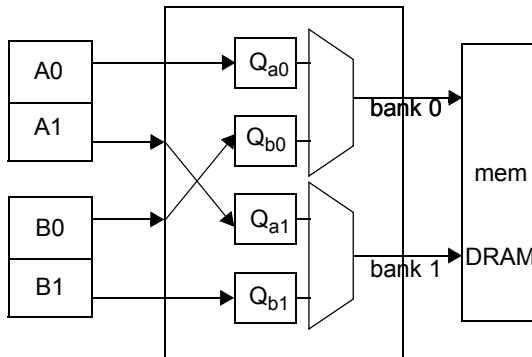


Each external/internal memory which is connected to the MC has it's own physical interface, set by memory request queues (mrq). Use arbiter to select the next transaction from the mrq.

FIGURE 1.11 Memory element with 4 separate 32-bit write to 1x128 bit

This type of hardware construct can be used when the system bus splits 128 bits into 4x32 bits. The sender transmits 4x32 bits on the system bus and the receiver converts the 4x32 into 1x128 bits.

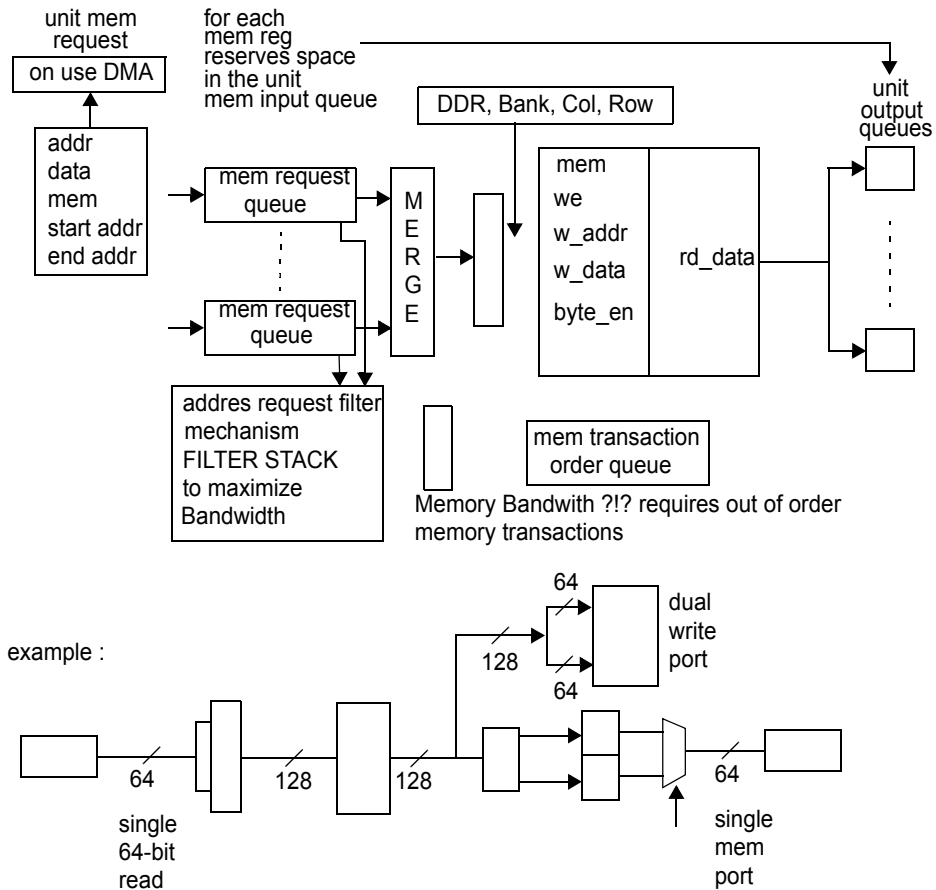
Don't split requests to memories if the reg goes to different memory s - maintain strict ordering. No out of order requests.

FIGURE 1.12 Memory controller input queues

Each request unit can send 2 request per cycle.

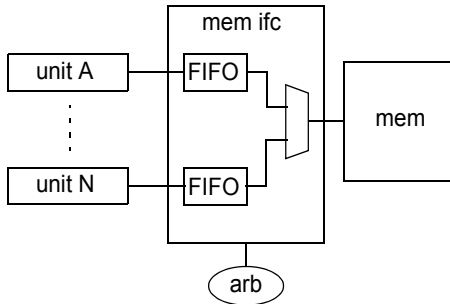
The MC queues the incoming requests in separate FIFOs there is one FIFO for each unit/ memory channel.

FIGURE 1.13 Memory Controller logical view



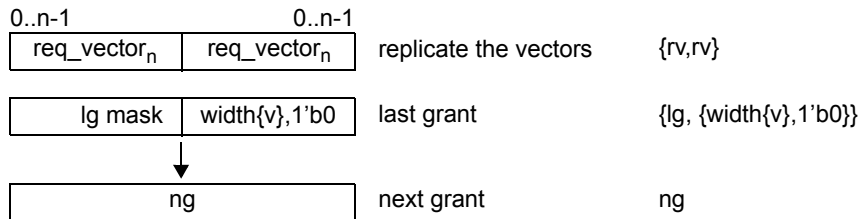
1.2.5.1 MC arbiter

FIGURE 1.14 Memory controller with arbiter selection



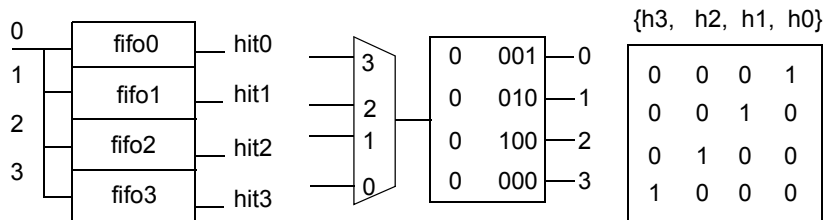
The arbiter decides which unit to select in the output mux.
 The arbiter inputs are the fifo status bits and the age information for each entry in the fifo.
 The arbiter outputs control the mux.

FIGURE 1.15 Priority arbitration logic circuit !!find the original pictures



A request vector is a bit vector where each bit corresponds to a unit asking an arbitration request. The bit is 1 if the unit is requesting access to the resource. Assume that only one unit is granted access per cycle the grant vector contains a 1 in the bit position that corresponds to the unit which was granted last cycle. Using a simple round robin arbitration scheme we grant access to units in priority order after the position of the last unit granted access.

FIGURE 1.16



Mux select is one hot encoded and is generated from:
 wire [31:0] mmv= rv2 [32:lastgrant-1:last_grant];

```

one position Find first One (mrv);
last_grant=grant;
grant = one_position>> last_grant;

```

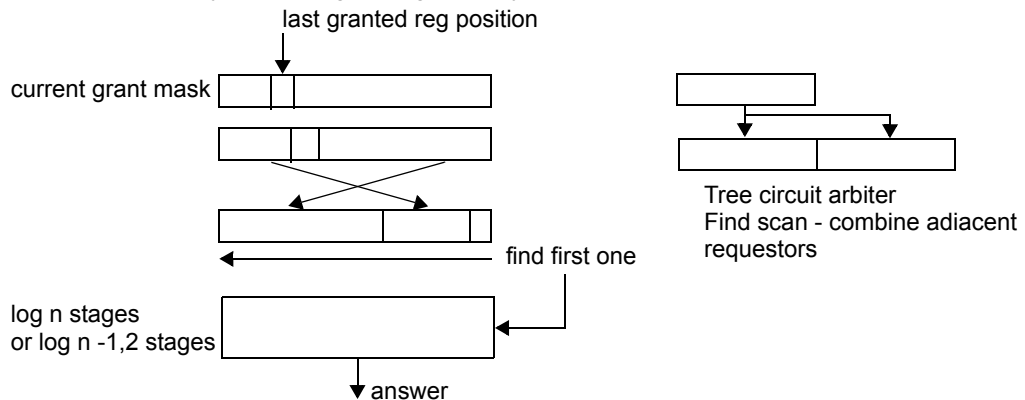
1.2.5.2 MC queue arbiter

```

horizontal / vertical flip
<size> <size> -> merge into 2x<size>_req address
addr  addr+1
Arbiter
last request number

```

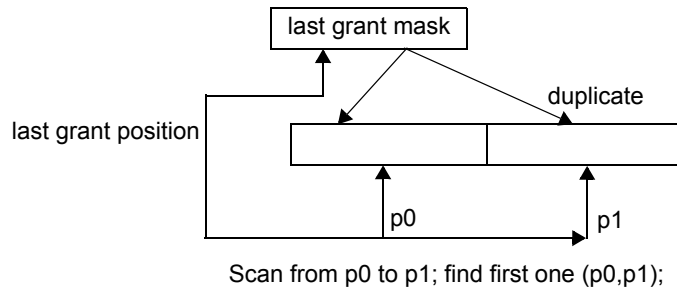
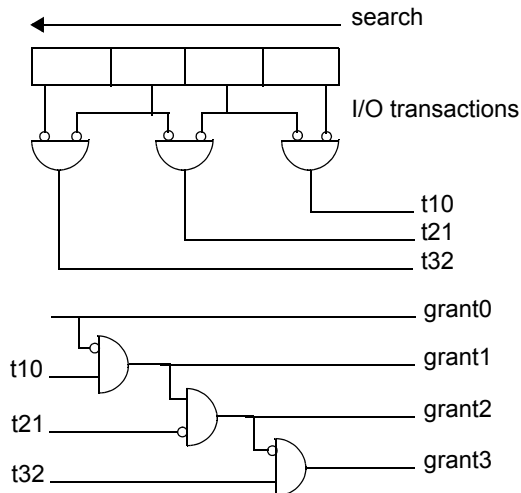
FIGURE 1.17 Memory controller grant logic priority mechanism !!find hand drawn picture !!!!! remake



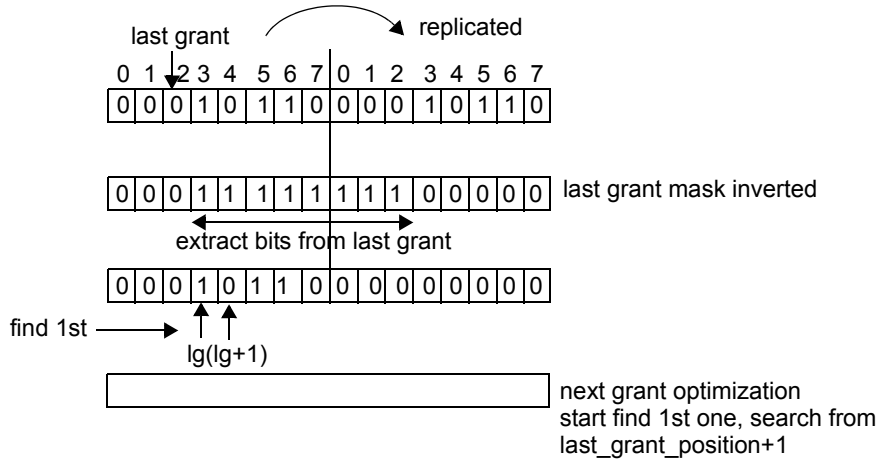
```

cvm={ (0|1)+ }
ex cvm? = {1,1?}
lnrg n = last unit grant number <n>
lnrg_mask = (1<<lnrg n)-1;
2x lnrg_mask={lnrg_mask,lnrg_mask};
cvm masked = lnrg_mask & ?? (cvm? <<lnrg n)
cvm masked[x+lnrg n-1: lnrg n??]
width of request
gvmt_vnm=find_first_1 (cvm masked);
optimization
use module
request mask [n-1:0];

```

FIGURE 1.18 Memory Controller design**FIGURE 1.19** Alternative priority encoder arbitration

Use propagate generate logic to speed up logic.

1.2.5.3 MC grant**FIGURE 1.20**

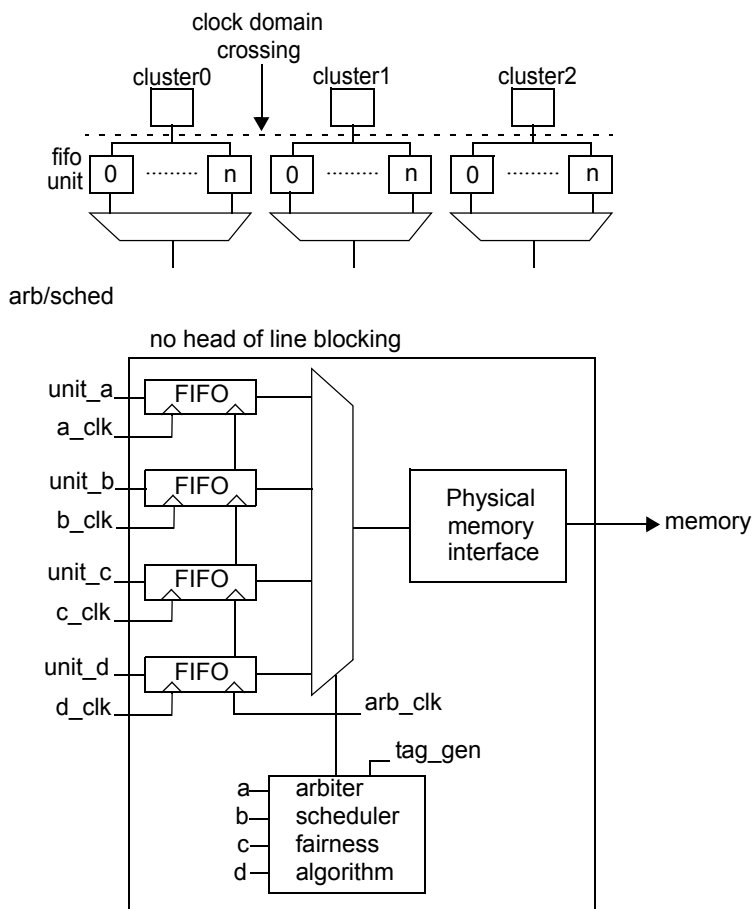
(rv) request vector [31:0]

(lg) last grant [4:0] 5 bits

rr2 [63:0]={rr,rr};

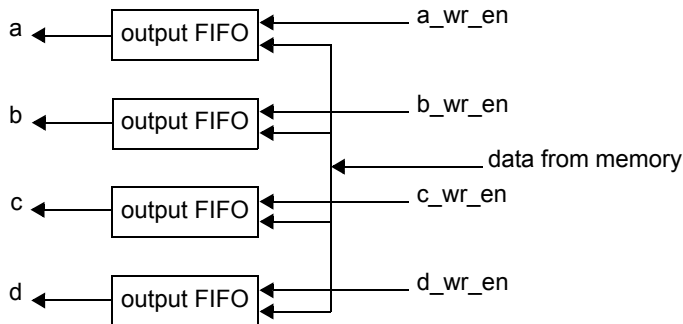
1.2.5.4 MC HOL blocking elimination

FIGURE 1.21 Eliminating head of line blocking



1.2.5.5 MC output FIFO

FIGURE 1.22 MC output fifo



1.2.5.6 MC system

MC traffic cases:

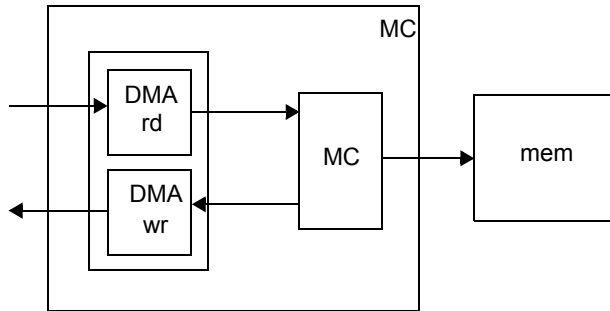
TABLE 1.4 Valid combination of active units

Unit	max bw	min bw	Valid unit combinations									
A	5	0	X	X	X	X						
B	5	5					X	X	X			
C	1	0		X		X		X		X	X	
D	4	0			X	X			X		X	X

A memory controller interface to the internal/external memory has a maximum bandwidth. Each unit has a min bandwidth requirement. The total bw of all units connected to the mem controller cannot exceed the max bw of MC. Diff combinations of units can be connected to the MC. We show in the example in Table 1.4 the valid combinations of units which may be connected to the MC to support different usage cases.

1.2.5.7 MC system DMA

FIGURE 1.23 Memory controller with DMA read and write engines



1.2.5.8 MC tile address engine

FIGURE 1.24 Tile

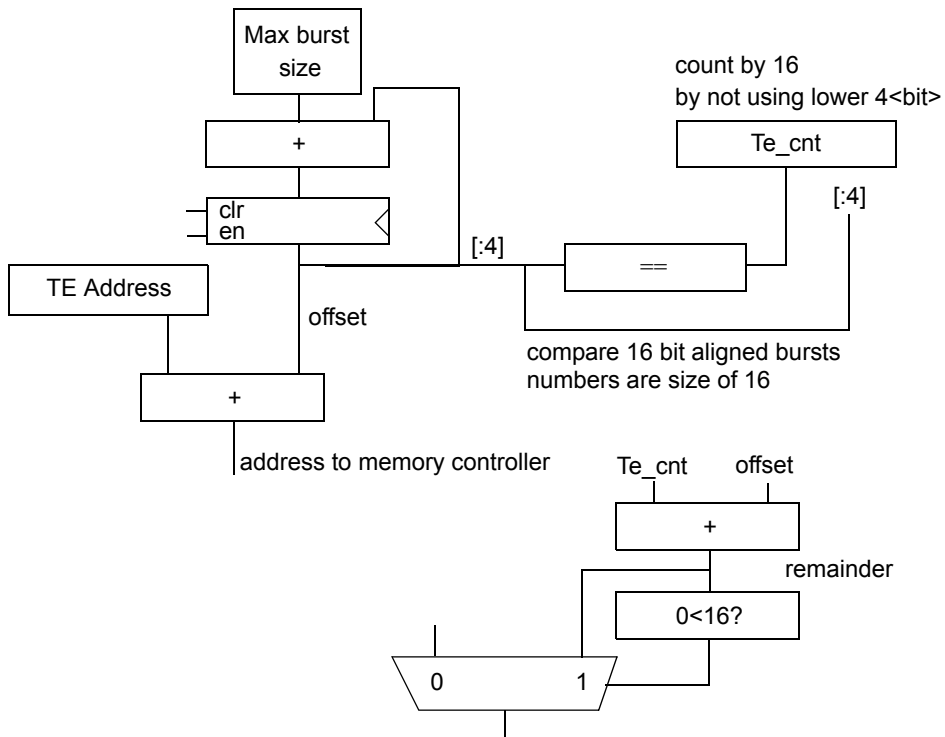
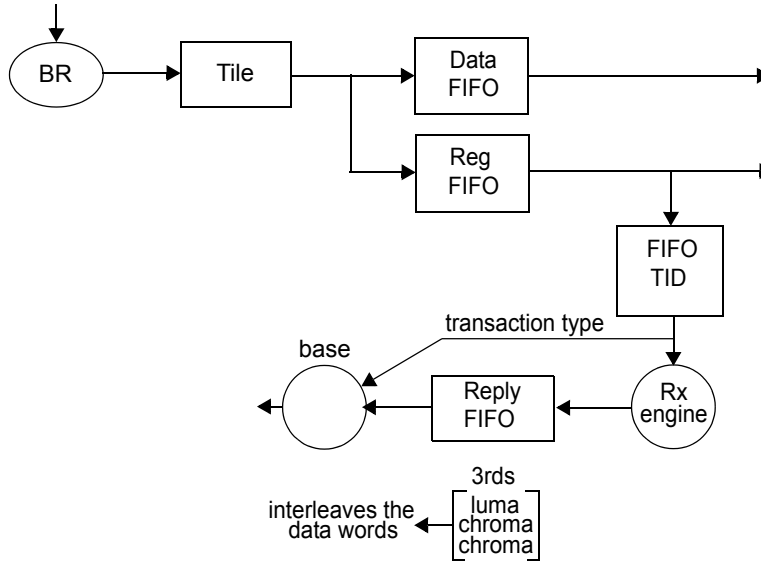


FIGURE 1.25 Tiling address logic !!find the original picture

convert the block acces
into individual word acces



horizontal size
line stride
modes

1.2.5.9 MC tile

FIGURE 1.26 Memory Controller Tiling

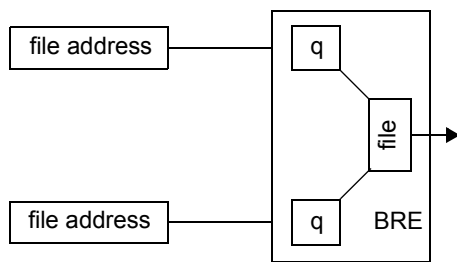
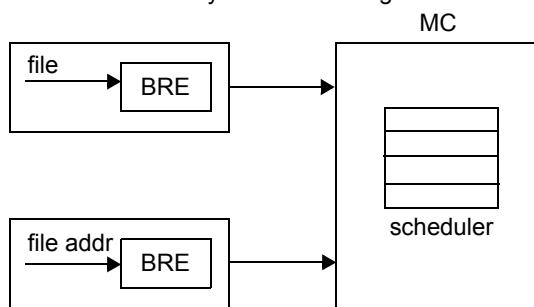
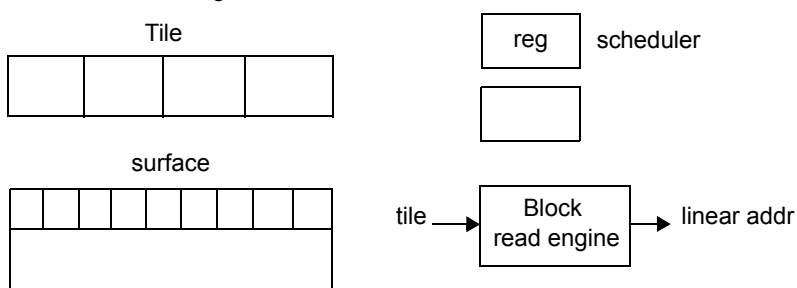
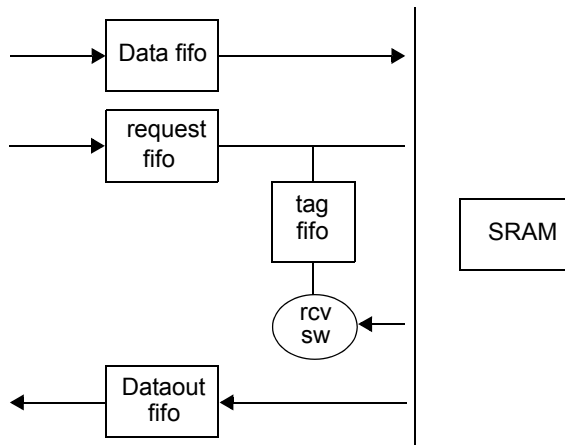


FIGURE 1.27 MC tiling



1.2.5.10 MC request/reply logic**FIGURE 1.28****1.2.5.11 MC features**

MC features:

- no head of line blocking
- split transaction from MC to MEM
- multiple outstanding requests from one channel
- multiple requests from MC to MEM in the same cycle
- input / output queue sizing based on clk frequency differences
- input / output queue sizing based on traffic patterns
- buffer pool support pointer + bkl length list

Different types of units:

- low bandwidth units
- high bandwidth units
- guaranteed bandwidth units
- high priority interrupts

Different priorities for different usage models.

Building hierarchies of units to communicate with the MC merge memory requests.

scheduler

fairness

debit / credit

Arbiter schedule:

min bandwidth requirement
max bandwidth requirement
memory transaction order :

- in order
- out of order - need a tag

performance counters

Schedulers maximize the memory utilization / bandwidth.

Need bypass queue.

Need (high/low priority) queue's in the clients.

Need to handle cache operations.

Optional :

- Tiling support
- Block operations (rectangle)

Goal - high bandwidth utilization

MC Filter Stack:

1. Bandwidth - min
2. Bandwidth - max
3. Aging
4. Fairness
5. Rate control
6. Priority Bypass
7. Current memory state (DDR2 bck, cd, rom)

there is a priority order in which the filters are considered when selecting the next state.

The parameters for 1-7 should be programmable.

1.2.5.12 MC physical interface

input from VM

output interfaces to :

- FLASH
- ROM
- SRAM
- DRAM
- DDR2
- DDR3
- DDR4

1.2.5.13 MC specifications

MC Bank number

Memory order requirements

latency return

CPU transaction ID's

The client memory space is specified. The memory word width is specified. The base address and number of word is specified

client memory spec

word_width:

base_address:

mem_of_words:

bits in memory protocol

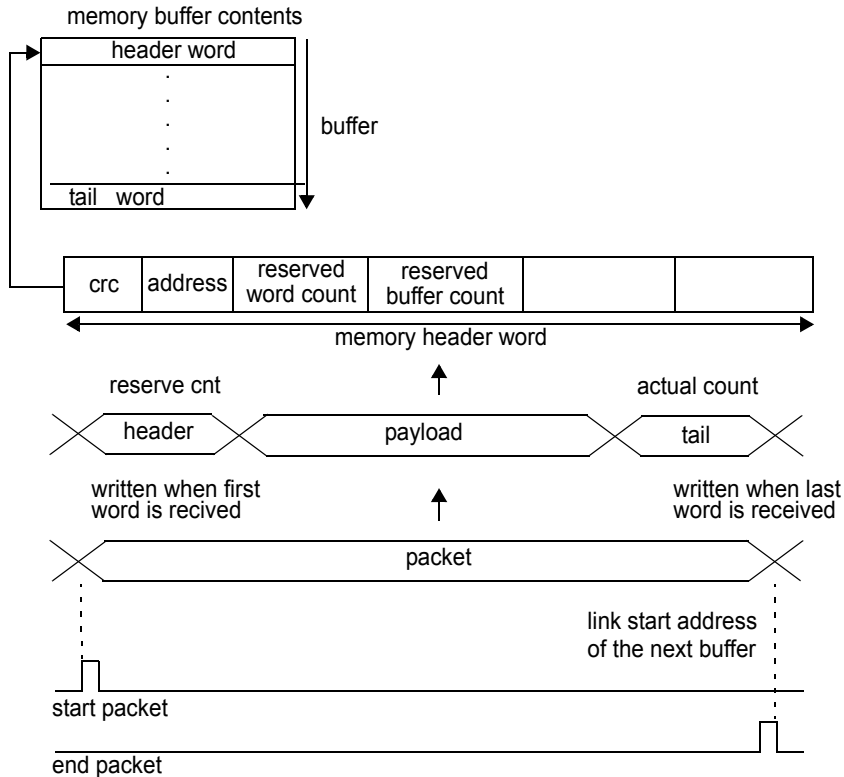
- free TID FIFO
- requested TID CAM
- request TID FIFO

1.2.6 Memory request packet

MRP's use the following format :

sender_id/destination_memory_id/transaction_id/gos_type/priority_level/?_load(length of ?load is 32-bit words)/crc.

FIGURE 1.29



On chip point to point memory bus signaling protocol
 credit / debit based
 ack / nack / wait / read

1.2.7 Memory formats

Make requests for tiled memory. Requires changing the block reads which span two 128 bit words horizontally be split into 2 requests of 128 bit columns.

The tile

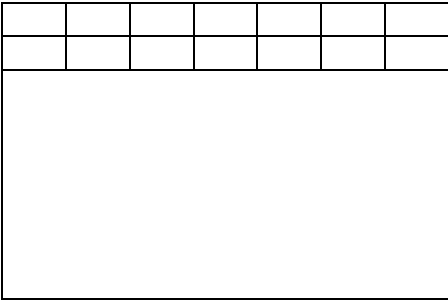
01
23

Which maps to a screen

13131313131313131313131313131313
02020202020202020202020202020202
13131313131313131313131313131313
02020202020202020202020202020202
13131313131313131313131313131313
02020202020202020202020202020202
13131313131313131313131313131313
02020202020202020202020202020202
13131313131313131313131313131313
02020202020202020202020202020202
13131313131313131313131313131313
02020202020202020202020202020202

Tiles in a surface

FIGURE 1.30 Tiles in a surface



x = bytes 14 bits addr
y = lines 12 bits addr

Tiled addr = base_addr +
 row y[11:4] * 16 * pitch +
 column x[13:4] * 256

 within y offset y[3:0] * 16 +
 the tile x offset x[3:0]
 pitch is 14 - 6 = 8 bits need an 8x8 multiplication

FIGURE 1.31 Pairing data in different memory segments

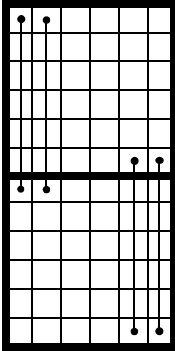
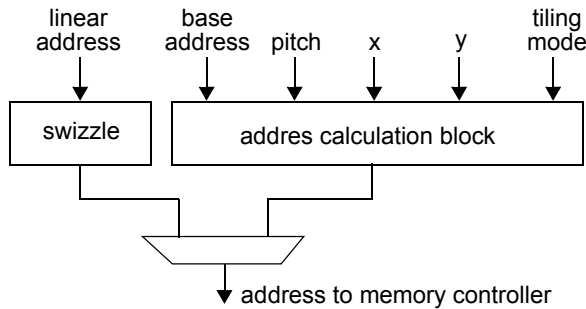
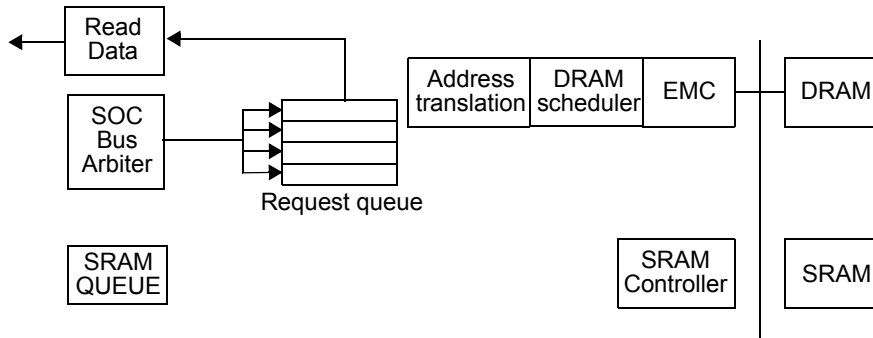


FIGURE 1.32 Tiled memory address calculation



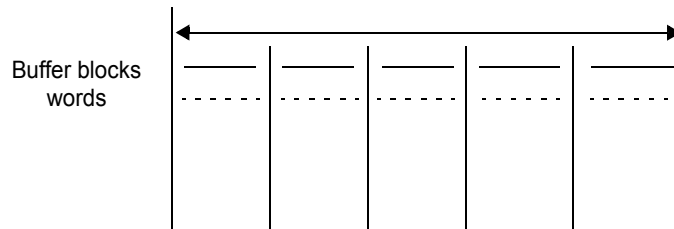
The swizzling process (performed by the swizzle block) is the conversion of references based on name or position (relative) to direct references. Mip Map levels start on 16 bytes boundary

FIGURE 1.33 Tiling different addresses - regions are tiled differently



Tiling different addresses regions are tiled differently

FIGURE 1.34 Memory organization



buffer_size_blks : buffer size measured in blocks
 buffer_size_wds : buffer size measured in words
 blk_size : block size in words
 blk_nr : number of blocks
 word_size

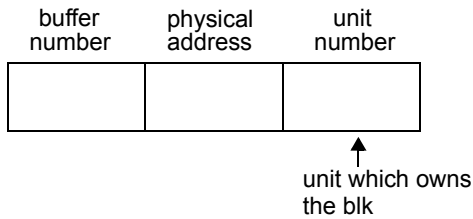
Memory Address Translation.

Memory address translation from a linear address to a bank / column / row address

DRAM / SRAM queuing + arbitration

1.2.8 Address

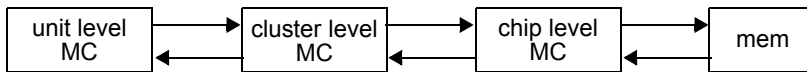
FIGURE 1.35 Buffer address location



V.M. - Virtual Memory
Need a simple fast VM

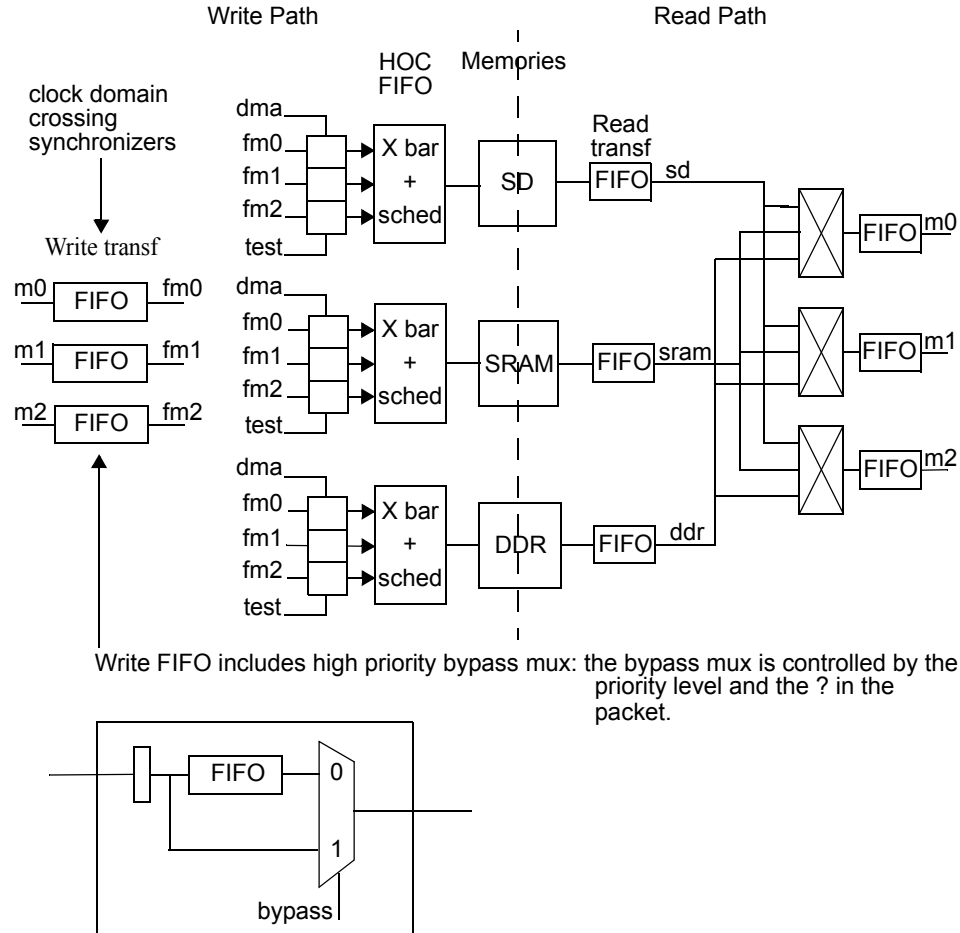
1.2.9 System level view of MCU+MCC+MRRT+MC.

FIGURE 1.36 MCU+MCC+MC+memory



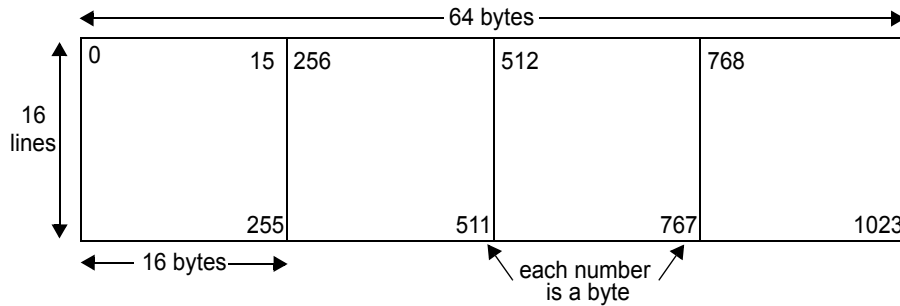
The units, MRRT, MC and internal/external memories can all be in different clock domains. Synchronizers (asynch FIFOs) are used to synchronize the memory request transactions across clock domains.

FIGURE 1.37 Chip level logical view



1.2.10 DRAM

DRAM is organized as a set of pages. Subsequent accesses within the same page are fast. Accesses to another page are slow. Pixels are clustered horizontally and vertically.

FIGURE 1.38 DRAM Page of 1Kb in size (1024bytes)

Tiling pitch has to be a multiple of screen divided

- DRAM memory operation
 - memory refresh
 - read operations(read from same bank as previous operation and read from different bank as previous operation)
 - write operations(same as reading from the same bank and different bank)

- SRAM memory architecture

SRAM (Static Random Access Memory) is a type of volatile semiconductor memory which retains its contents as long as power is applied. Four transistors are required to store each bit in a SRAM. Two additional transistors serve as access control to a bit storage cell for read and write operations.

- DRAM memory architecture

SDRAM (Dynamic Random Access Memory) is a type of volatile semiconductor memory which, unlike SRAM, needs to periodically refresh its contents. This has to do with the fact that DRAM uses a capacitor to store each bit, and real-world capacitors "leak" electrons in time, thus the information eventually fades unless the capacitor charge is not periodically refreshed. An additional transistor is required for each storage cell to provide read/write access.


- memory bank
- memory addressing logic
 - SRAM
 - DRAM
- memory transaction req packet

SOC Memory System Performance Analysis

SOC memory system performance varies according to:

- number of clients

- size of queue's in clients
 - number of memory banks
 - number of memory input / outputs
 - size of memory input / outputs
- clk rates of memory clients and mem cnt

1.open page  2.rd/wr page  3.close page 

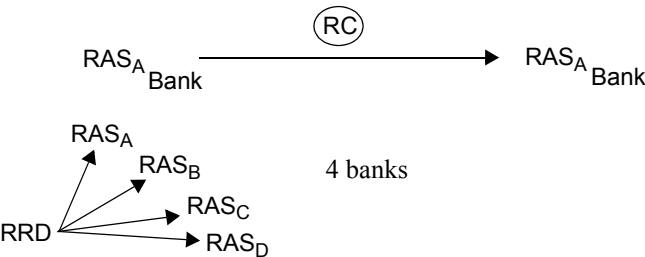
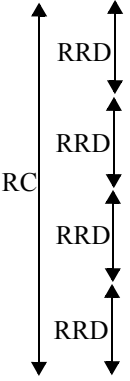
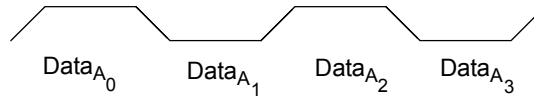


TABLE 1.5 Maximize performance and power

	Cycle	Command	Dataout
	0	RAS _A	
	1		
	2	RAS _B	
	3	CAS _A (ap)	
	4	RAS _C	
	5	CAS _B (ap)	Data _A
	6	RAS _D	Data _A
	7		Data _B
	8	RAS _A	

4 transfers
Burst_length(BL)=4

FIGURE 1.39



Make a graph of the memory and dependencies

Use counters to age counters

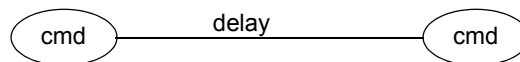
Use counters to track events.

RAS -> CAS

CAS tap -> RAS

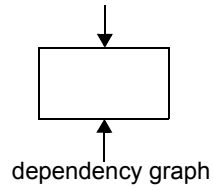
Design: stream of requests, send

FIGURE 1.40



Denali file format for describing memory characteristics.

FIGURE 1.41

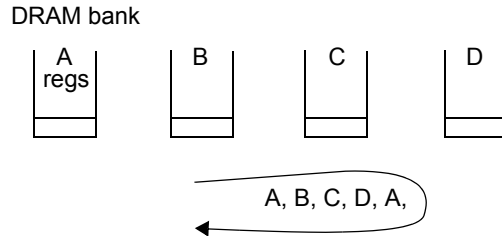


out_of_order
allows more bank regs to be at head of queue

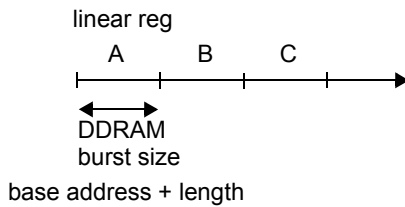
0 RAS_A
1
2
3 CAS_A
4
5 CAS_A
6
7 CAS_A
8
9 CAS_A + Auto pre change
10

MC Arb
goal ≥ 1 reg in each bank at all times

FIGURE 1.42



Up another level - how to map client regs to DRAM banks?



1.2.11 Requirements

1.2.12 Simulate

Various SOC (System-on-chip) units cannot have a uniform bandwidth requirement during the multi-media applications. The memory controller can deliver 70% to 80 % efficiency, but then it will drop to 50% because there is not enough bandwidth demand for the current set of client requests.

Memory allocation

hardware new | free mechanism
hardware garbage collection

unit requests mem allocation
mem cnt allocates memory and sends back buffer number to unit
unit writes to buffer

Units have associated memory formats
Different memory formats

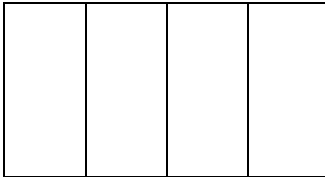
10/9/09

Confidential Copyright © 2007 Fastpath Logic, Inc. Copying in any form without the expressed written permission of Fastpath Logic, Inc. is prohibited

Different clock Domains
 Page Bit
 Parallel Banking
 Texture irregular stride pattern
 linear burst
 Scan line

FIGURE 1.43 Texels 16x16

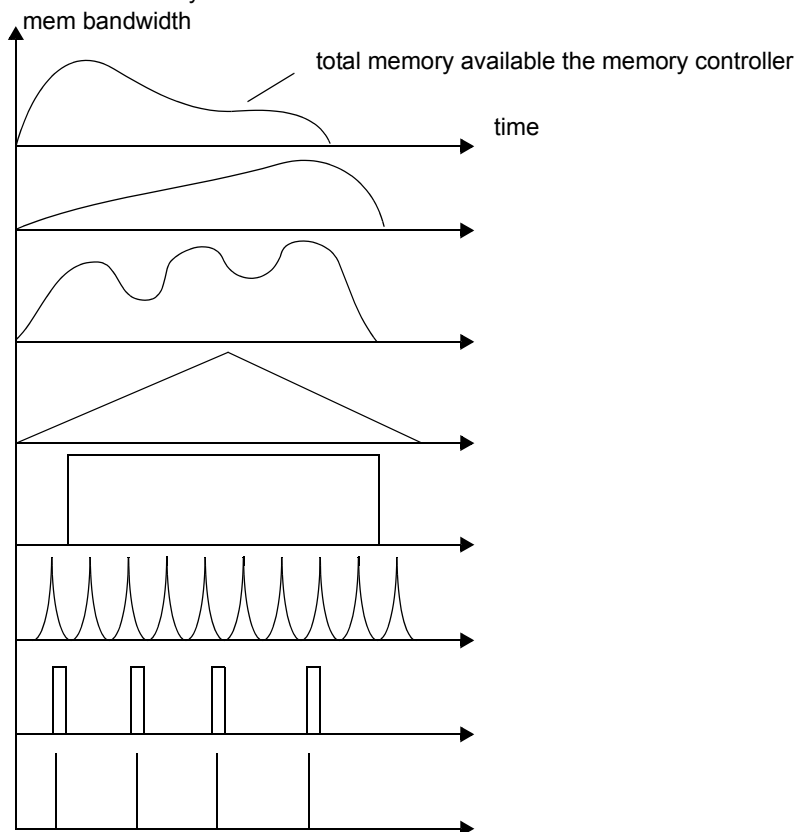
each block is a bank



17x17
 1/2 pel shift only
 touch 2 banks
 checker board memory

Memory traffic is the bandwidth access pattern between the MC and the memories. Figure 1.40 contains graphs which represent different memory traffic cases.

Gen memory tracer:

FIGURE 1.44 Memory Traffic Cases

Data can be preloaded into memory then we need to know the address that have been written.
Texture downloads

FIGURE 1.45

memory at the end of the test may not be the same

C model arbiter diff event timing, diff event addresses may not be the same in the C and RTL.
Memory footprint and transaction may not be the same.
Transaction payload, padded, split.

1.2.13 Test

MC testing the interaction between a client and the memory controller. Vary the memory response
time client -> MC memory
put a random() function between them
The amount of time that the MC takes to respond to a memory request will be varied

Problems with SOC memory hierarchies:
Flow control-mem client can make memory req and then a down stream consumer can stall the
memory client.

1.2.14 Notes

Memory Controller

base

A0

A1

A2

A3

base

B0

B1

B2

B3

1. random access

2. deterministic

1.2.15 Performance analysis

Transaction order type:

- in order
- out of order

Linear order processing memory hierarchy:

- on chip cache
- on chip SRAM
- on chip DRAM
- off chip

Unit which is consuming data - can't tolerate out of order data - a cache line fill can be out of order.

FIR filter - must be in order.

- Memory client
 - DMA engine
 - memory address calculation unit
- Memory request cluster
- Memory traffic
- Memory map
 - physical memory map
 - logical/architectural memory map
- Memory controller
 - memory address calculation unit
 - types
 - internal SRAM controler
 - external DRAM
 - internal SRAM
 - external DRAM
 - input request queue
 - input arbiter
 - ouput reply queue
 - output req logic
 - addres transaltion unit
 - dram scheduler
 - phiyscal controllers
- Memory organization
- Linear order data processing memory hierarchy
 - onchip cache
 - onchip SRAM
 - onchip DRAM
 - offchip SRAM
 - offchip DRAM
- Request que
- Arbitration logic

- Grant logic
- Ensuring fairness/scheduling algorithms
- Priority select logic
- Minimum bw req/guarantee bw
- Memory req format/fields in mem req packet header
- Memory tiling
- Mapping screen pixels to memory addresses
- SRAM memory operation

FIGURE 1.46

