Computer Architecture: Fundamentals, Tradeoffs, Challenges

Chapter 6: Physical Memory

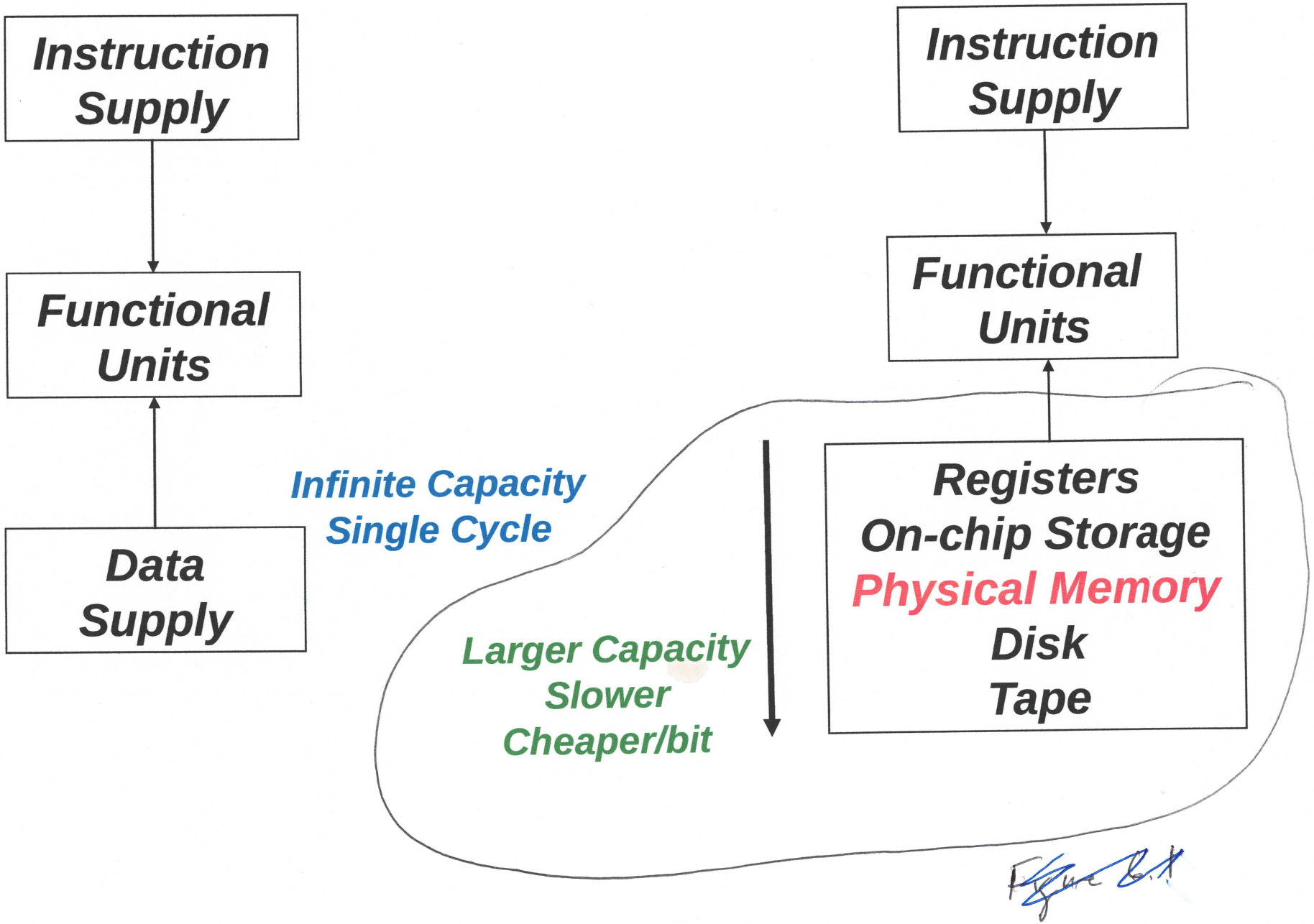
Yale Patt
The University of Texas at Austin

Austin, Texas
Fall, 2022

Outline

- The Storage hierarchy
  - Structures: Registers, L1/L2/L3…Cache, Memory, Disk, Tape
  - Access: RAM, DASD, Sequential, CAM
- Two important concepts
  - Interleaving
  - Unaligned Access
- Device Technology: Mag. Cores, SRAM, DRAM, NVM
  - The DRAM chip
    - Multiple Banks
    - Row Buffer
- The Memory Controller
- Error Detection, Correction

The Storage Hierarchy



	L1	L2	L3	Memory	DISK	TAPE
CAPACITY	32KB	128KB and up	1MB and up	8MB - 32GB and TB	16GB to TB	INFINITE
Access Time LATENCY	2 cycles	8 - 32 cycles	16 - 64 cycles	can be 50 msec	2.5 msec	INFINITE

Figure 6.1 STORAGE HIERARCHY

Figure 6.2. Aligned and Unaligned Information

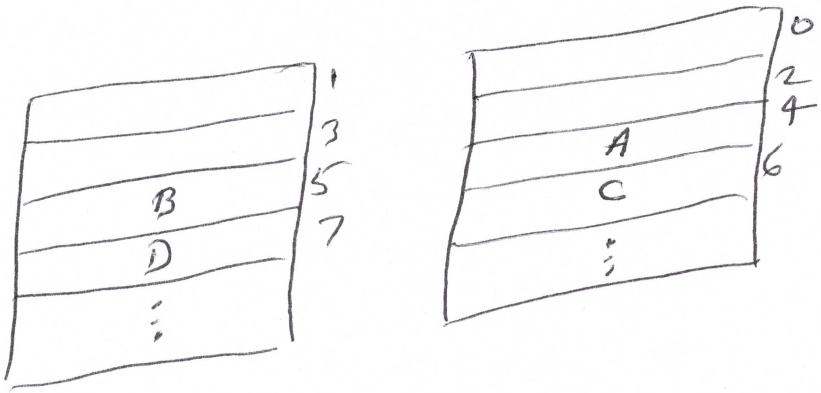
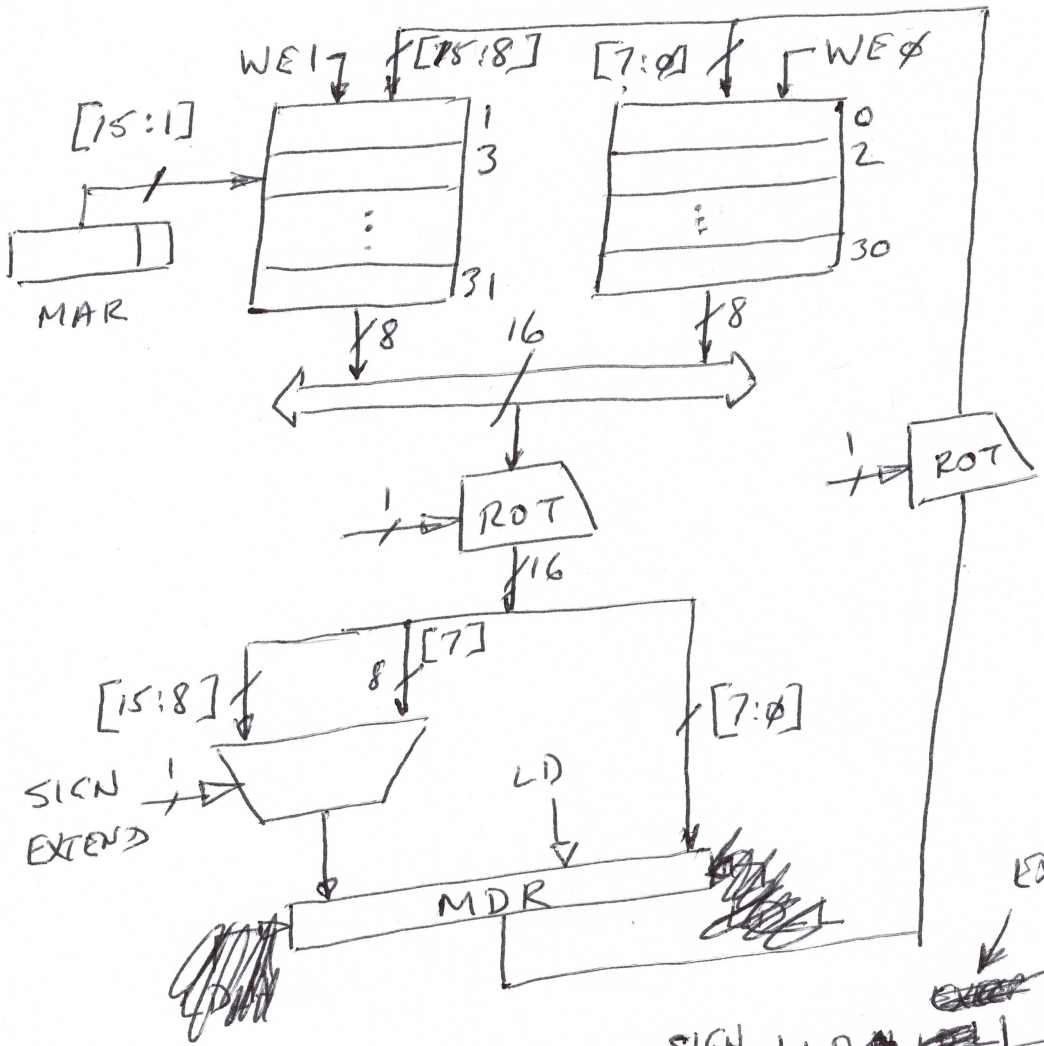


FIGURE 6.8

Aligned Access

ALIGNED ACCESS

64 BYTES OF MEMORY
(32 BYTE CHIPS)



LD/ST	MAR[0]	W/B	ROT	SIGN EXTEND	LD	EXCEP	WE1	WE0
LD	0	W	0	0	1	0	0	0
LD	0	B	0	1	1	0	0	0
LD	1	W	X	X	0	1	0	0
LD	1	B	1	1	1	0	0	0
LD								
ST	0	W	0	X	0	0	1	1
ST	0	B	0	X	0	0	0	1
ST	1	W	X	X	0	1	0	0
ST	1	B	1	X	0	0	1	0
ST								

ACCESS (UNALIGNED)

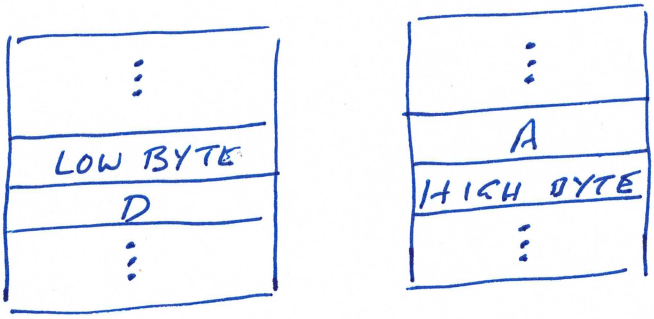
Figure 6.4

LD/ST	MAR[0]	W/B	1ST/2ND	ROT	LD_H	SEXT	LD_L	WE1	WE0
LD	1	W	1	1	*	X	1	0	0
LD	1	W	2	1	1	0	0	0	0
ST	1	W	1	1	0	X	0	1	0
ST	1	W	2	1	0	X	0	0	1

INC MAR →

INC MAR →

LOAD



AFTER (1) MDR | A | LOW BYTE |

AFTER (2) MDR | HIGH BYTE | LOW BYTE |

STORE

MDR | HIGH BYTE | LOW BYTE |

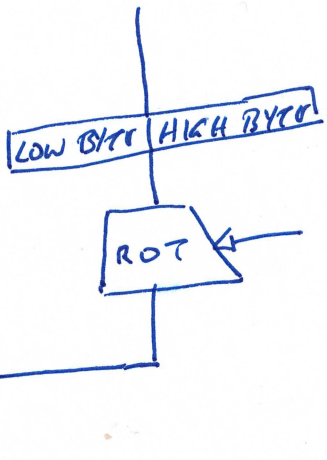
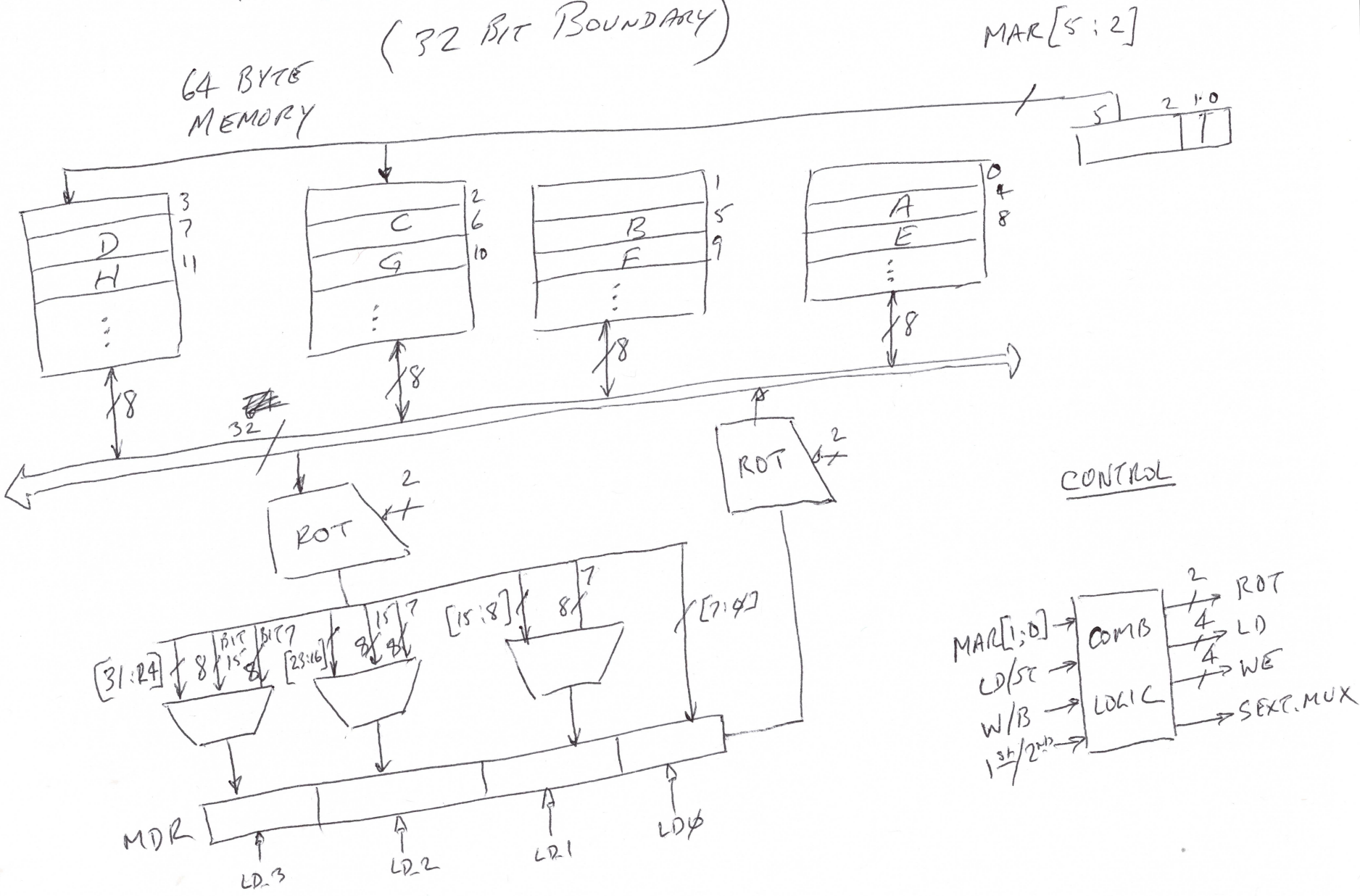


FIG. 6.5 UNALIGNED ACCESS (32 BIT BOUNDARY)



Interleaving

- 2-way interleaved (i.e., 2 banks)
- 64 bytes of memory, using 16 byte chips
- 16 bit bus supplied by one of the two banks

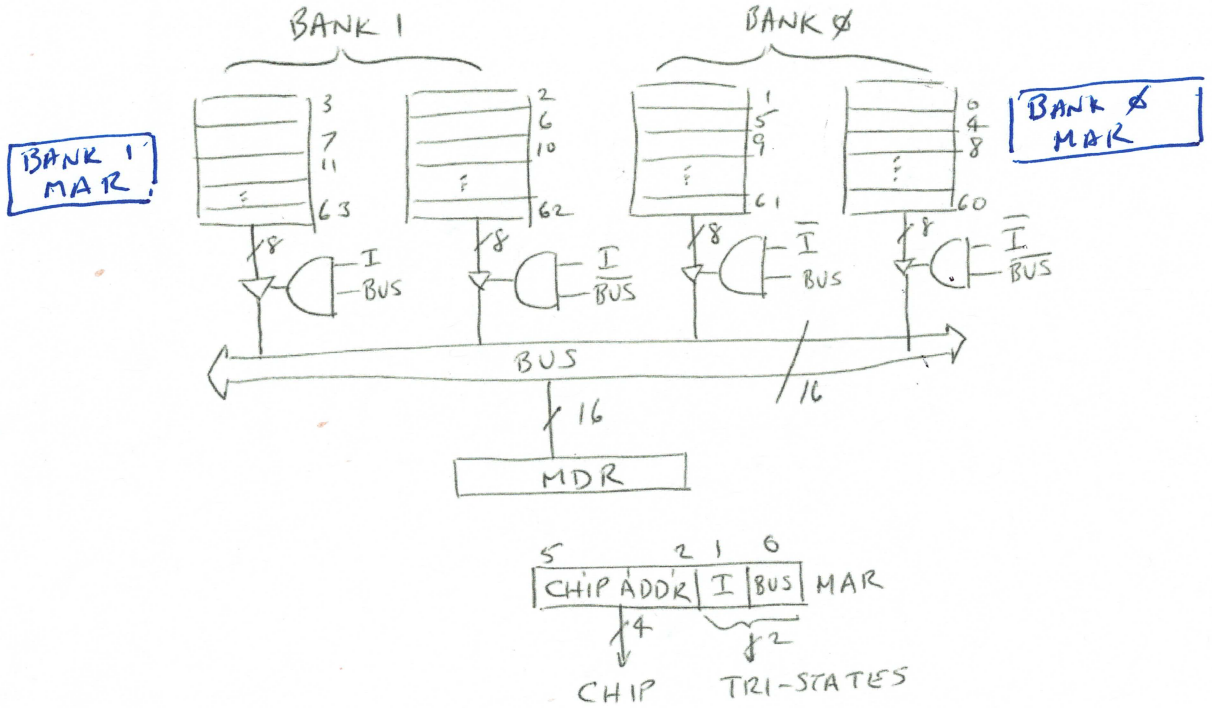


FIGURE 6.7. ~~ANOTHER EXAMPLE OF interleaving~~ Interleaving, an important mechanism for performing vector instructs.

4 WAY INTERLEAVES

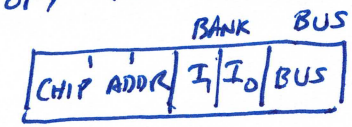
ACCESS TAKES TEN CYCLES

STRIDE = 1

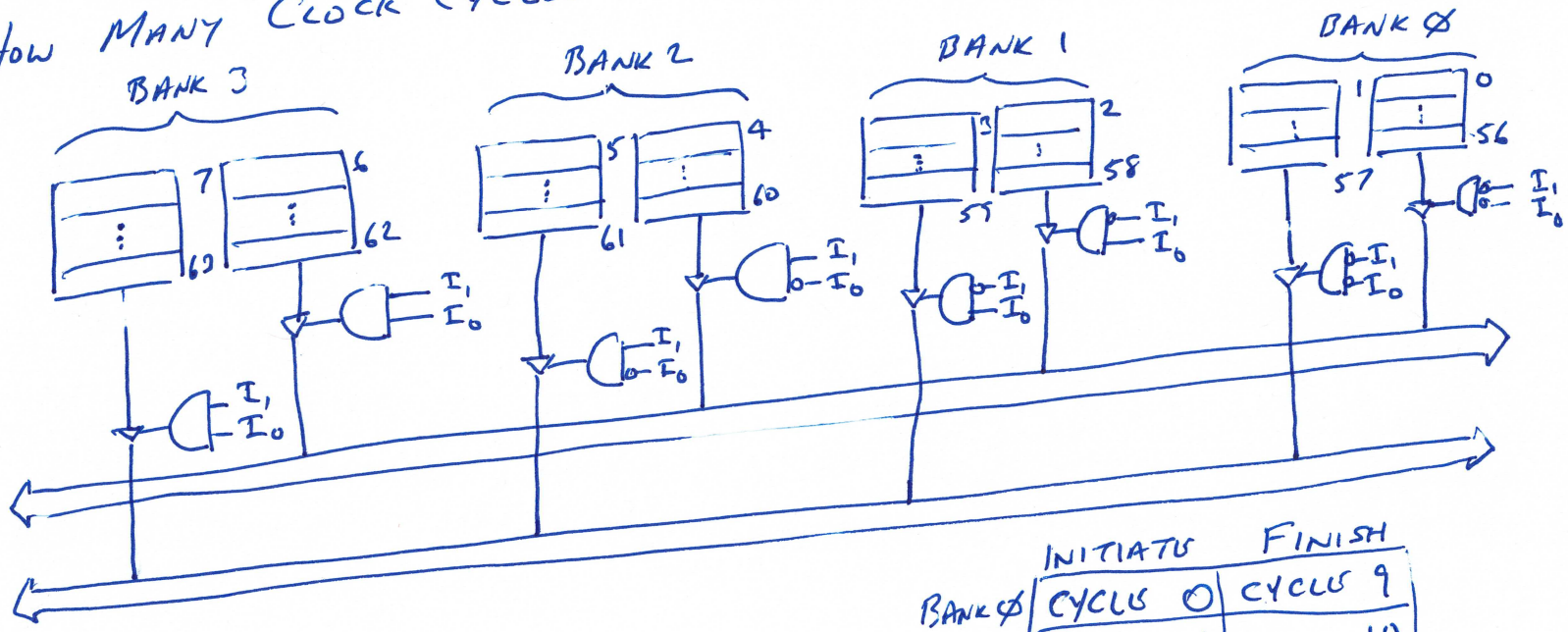
LENGTH = 6

SIZE = 16 BITS

MEMORY IS 64 BYTES



HOW MANY CLOCK CYCLES TO EXECUTE | VLD V1, A |



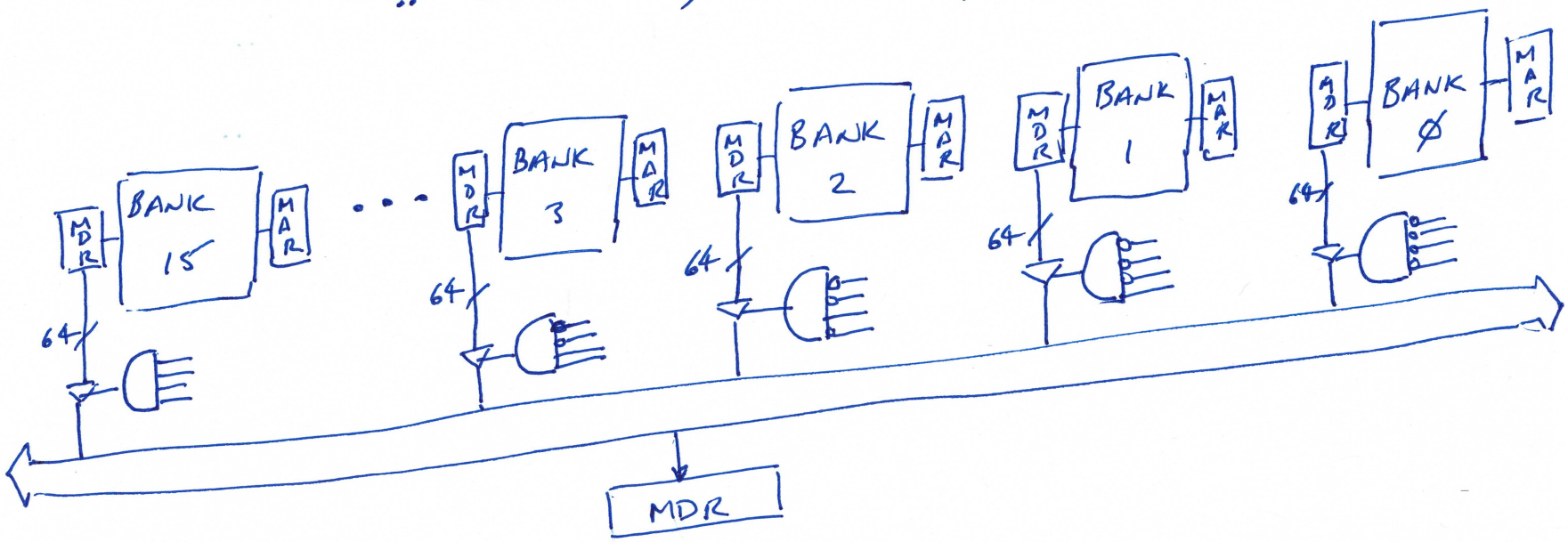
SIX ACCESSES →

	INITIATE CYCLE 0	FINISH CYCLE 9
BANK Ø	1	10
BANK 1	2	11
BANK 2	3	12
BANK 3	10	19
BANK 0	11	20
BANK 1		

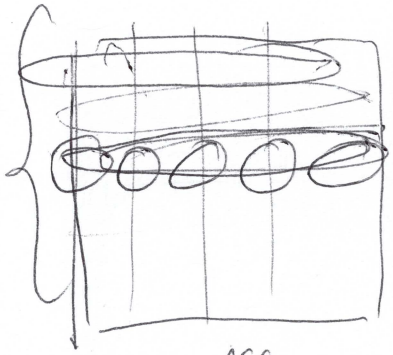
Figure 6.8. Memory Interleaving on CRAY 1

Access time = 11 clockcycles

∴ Interleaving is 16 way (16 Banks



The Devices and their Tradeoffs



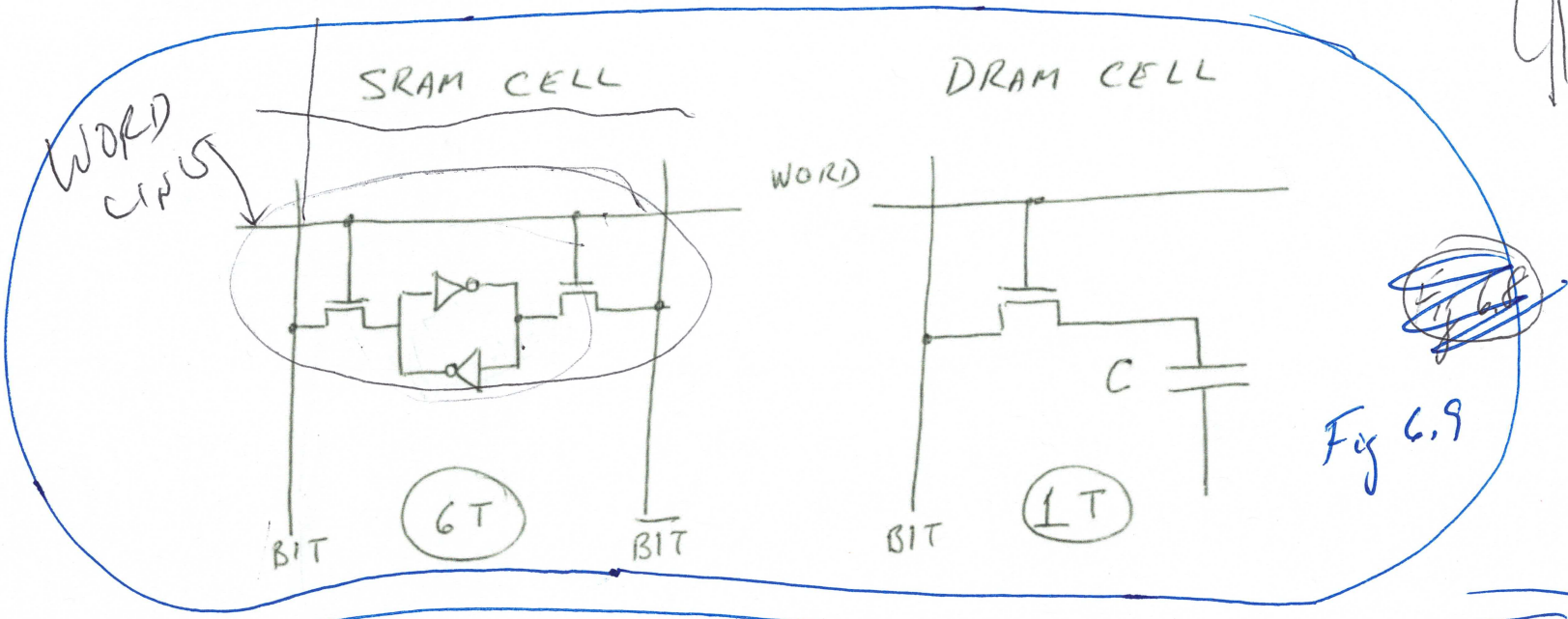
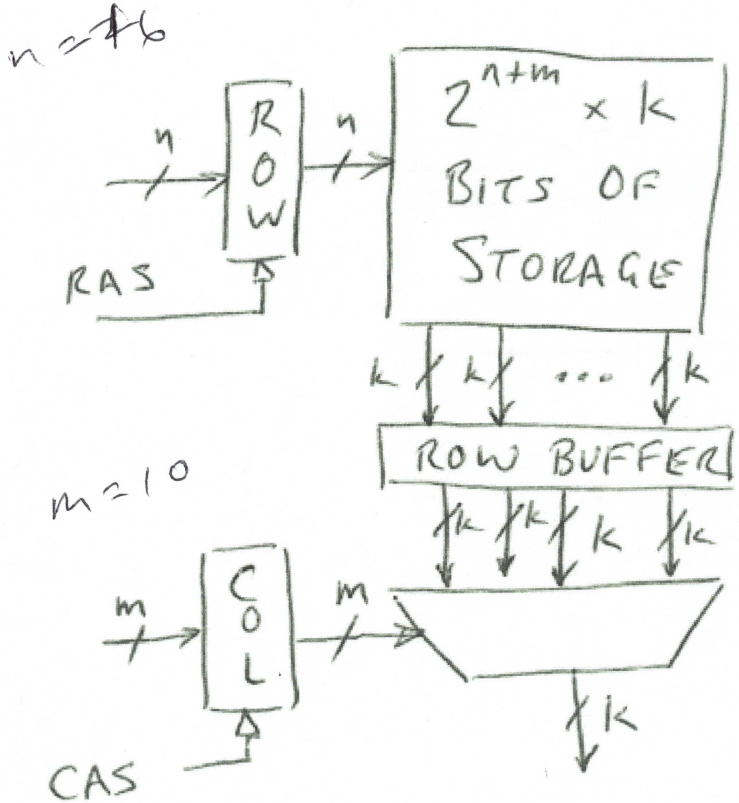


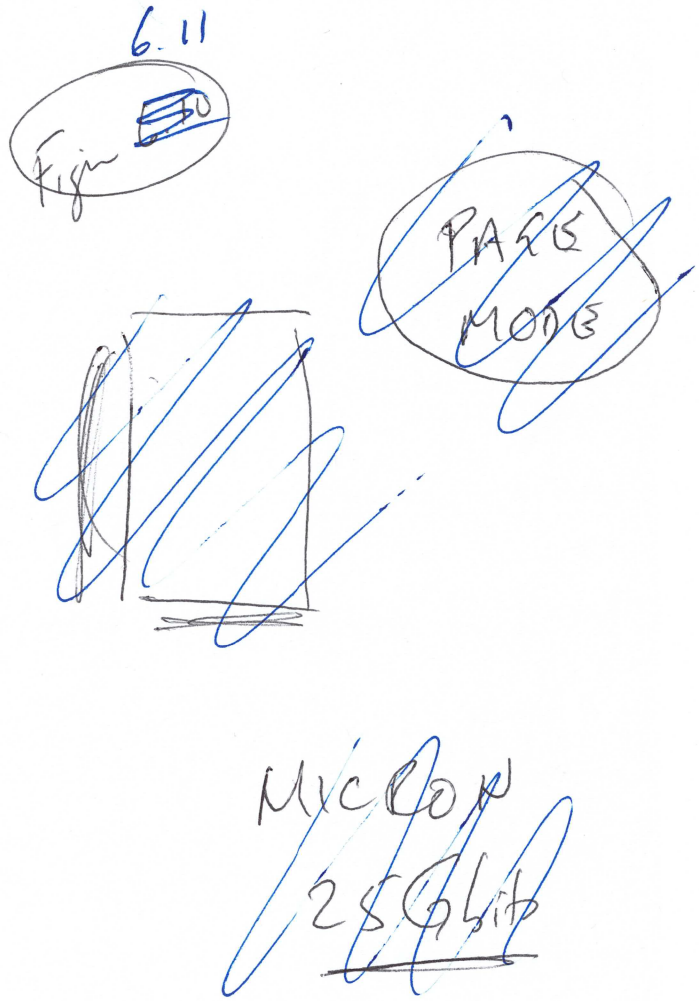
Fig 6.9

	SRAM	DRAM	NVM
Latency:	Low	High	Highest
Density:	Low	High	Highest
Persistence:	~~Static~~ No	~~Dynamic~~ No	~~Non-vol~~ Yes
Refresh:	No	Yes	No

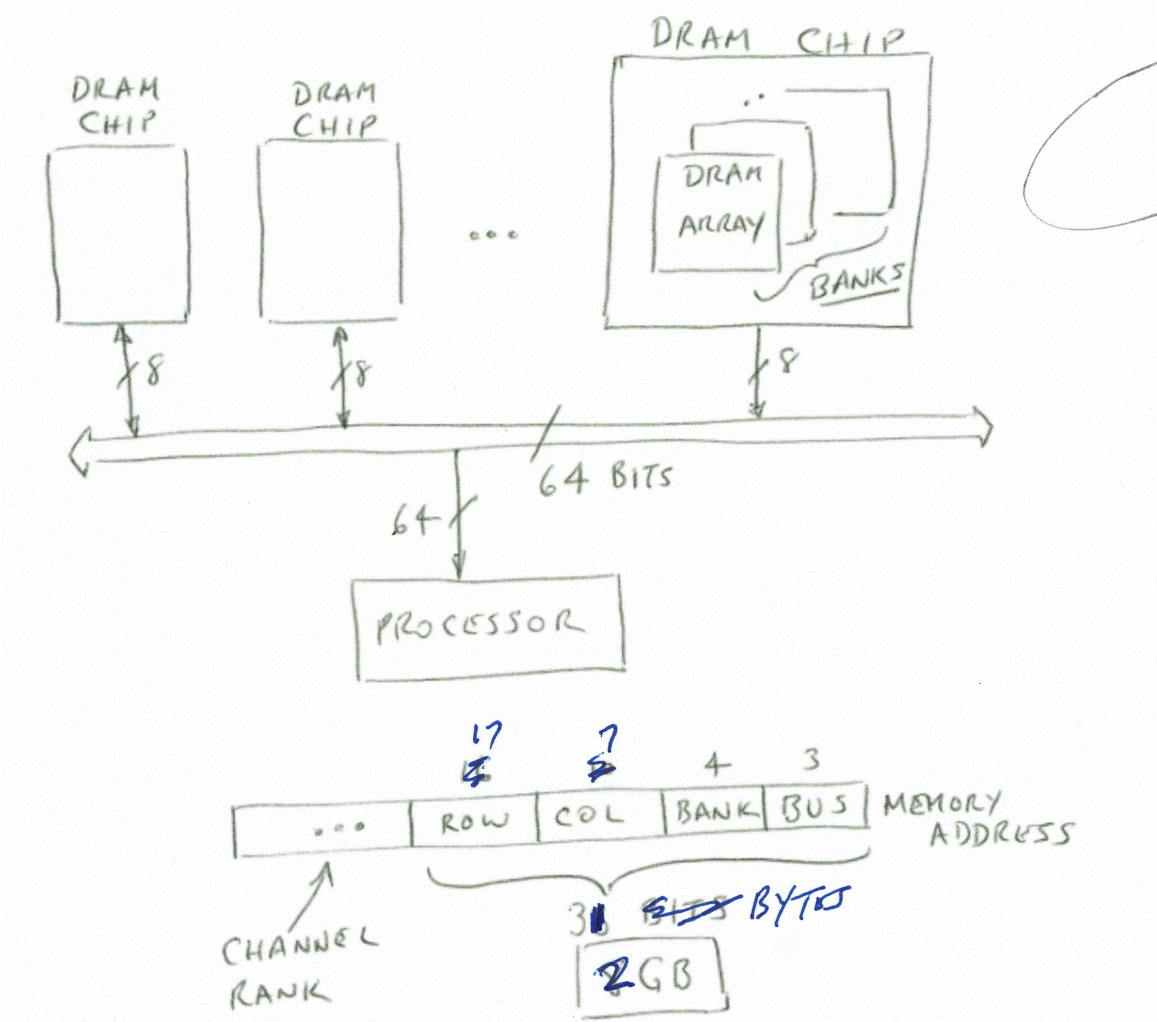
Figure 6.10

The DRAM Array





DRAM Memory


DRAM CHIP
DRAM CHIP
DRAM CHIP
DRAM ARRAY
BANKS
8
8
8
64 BITS
64
PROCESSOR
17
4
7
4
3
...
ROW
COL
BANK
BUS
MEMORY ADDRESS
CHANNEL
RANK
BYTES
2GB

The Memory Controller

- *Determines which access to initiate*
  - *Bank information*
  - *Row buffer open/closed, last access R/W*
  - *Demand vs Prefetch*
- *One per channel*
- *Between the core and the DRAMs*

Error Detection/Correction

- **Parity**
  - Detects single bit errors
  - Errors must be statistically independent
- **ECC**
  - When detecting is not good enough
  - Corrects single bit errors
  - Errors must be statistically independent
- **Checksum**
  - For large numbers of bits transmitted
  - Errors are not statistically independent

Parity

- *Simplest mechanism*
- *Detects single bit errors if statistically independent*
- *Typically, for 8 bits of data, we transfer 9 bits*
- *The 9th bit is the XOR of the 8 information bits*
  - *Guarantees that the number of 1's transferred is even*
  - *At destination, count them. If odd, an error has occurred!*
  - *Retransmit!*

FIGURE 6.13 ERROR DETECTION

EXAMPLE: 8 BITS OF DATA TO BE TRANSMITTED
1 BIT IS NEEDED FOR DETECTION

9 BITS ARE TRANSMITTED — AN EVEN NO. OF 1s.

EX. 1. 8 BITS OF DATA 10101010. 9TH BIT IS 0
EX. 2. 8 BITS OF DATA 11100000 9TH BIT IS 1

ECC

- *Errror Correcting Codes (when detecting is not enough*
- *Allows the correct information to be reconstructed*
- *We show by an example:*
  - *We want to transfer n bits (let n=8 in this example))*
  - *We specify n+logn+1 bits (i.e., 8+3+1 bits) as follows, where Di is a data bit, and Pi is one of the extra logn+1 bits.*

Bit:	12	11	10	09	08	07	06	05	04	03	02	01
	D7	D6	D5	D4	P8	D3	D2	D1	P4	D0	P2	P1

Note the bit number of each bit (e.g., D4 is Bit 9, in binary 1001):

D7	D6	D5	D4	P8	D3	D2	D1	P4	D0	P2	P1
1	1	1	1	1	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	0	0	0
0	1	1	0	0	1	1	0	0	1	1	0
0	1	0	1	0	1	0	1	0	1	0	1

ECC (continued)

- *Continuing…*
  - *We form four parity (i.e., XOR) functions, one for each row, XORing the bits in each row that has a 1 in its entry.*
    For example, P8 = XOR (D7, D6, D5, D4)
    For example, P4 = XOR (D7, D3, D2, D1)
  - *At the destination, the four parity functions are examined*
  - *If any gave an odd number of 1s, it must have been caused by the bit that transmitted in error.*
  - *We identify that bit by its "bit number," and correct it!*
    e.g., if D4 flipped, it would cause parity errors for P8 and P1, but not P4 or P2. P8(1),P4(0),P2(0),P1(1) identifies 1001, the bit number for D4, so we can correct it.

FIGURE 6, 14. ERROR CORRECTION

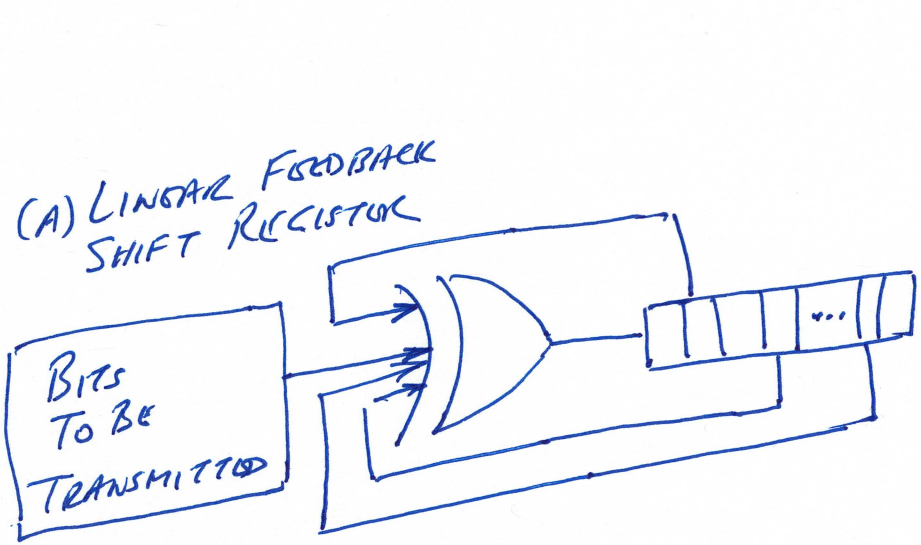
EXAMPLE: 8 BITS OF DATA TO BE TRANSFORMED
4 PARITY BITS ARE NEEDED FOR CORRECTION

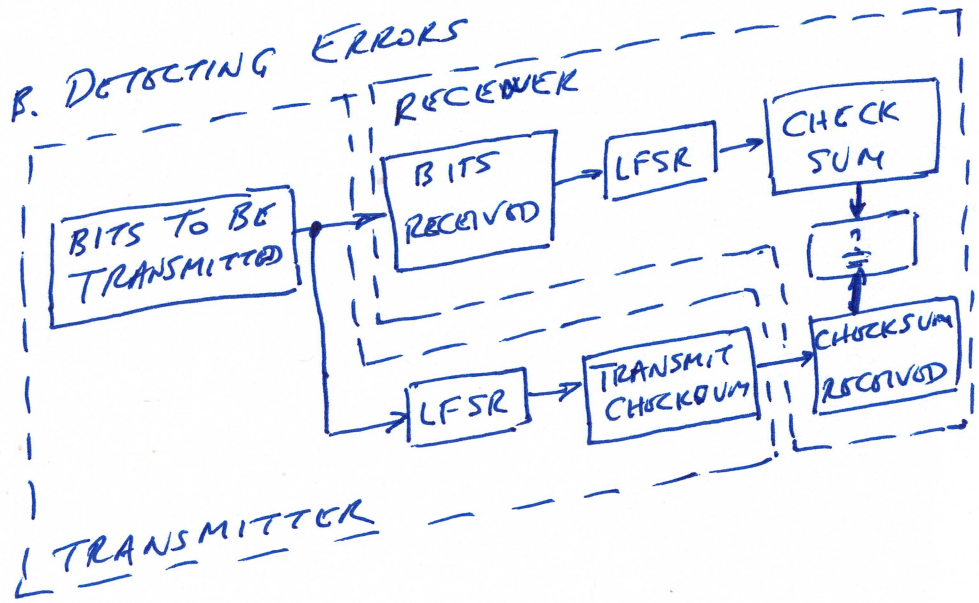
BIT:	12	11	10	9	8	7	6	5	4	3	2	1
	D7	D6	D5	D4	P8	D3	D2	D1	P4	D0	P2	P1
PARITY 8	*	*	*	*	*				*	*	*	*
PARITY 4	*					*	*				*	*
PARITY 2		*	*			*	*		*		*	
PARITY 1		*		*		*		*		*		*

Checksum

- ***When the probability of error is not statistically independent***
- ***and there is likely to be a burst of bits in error***
- ***Original scheme: use a linear feedback shift register***
  - ***Input bit-serial the information to be transferred***
  - ***Output the bits from the shift register***
  - ***After the input has been output, output the content of LFSR***
  - ***At the destination, repeat the process***
  - ***If an error occurred, it will show up in the LFSR***

FIGURE 6.15 CHECKSUM FOR DETECTING BURST ERRORS





Todah!