FIFO Buffer

- "Elastic" storage between two subsystems

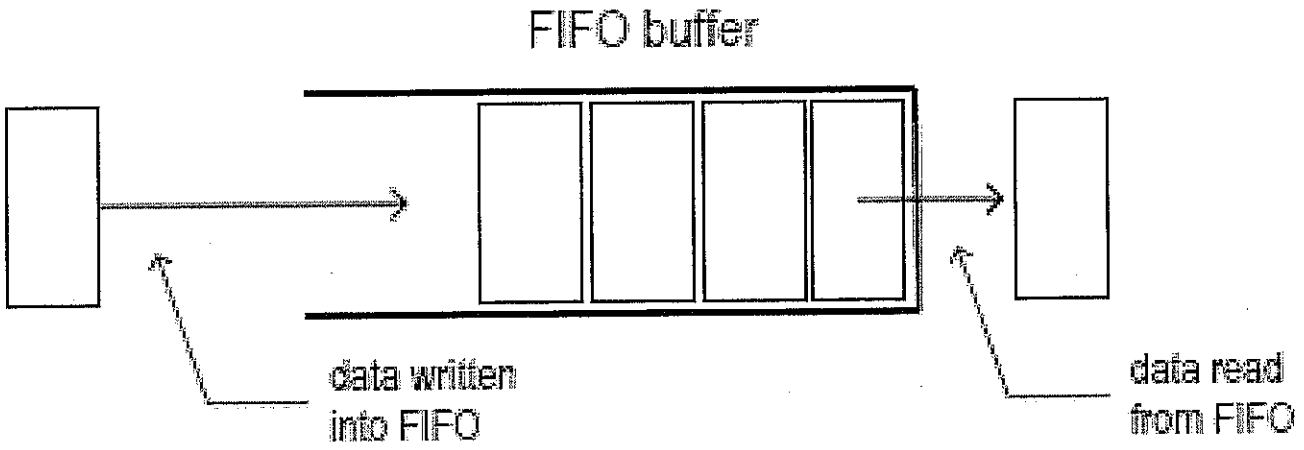
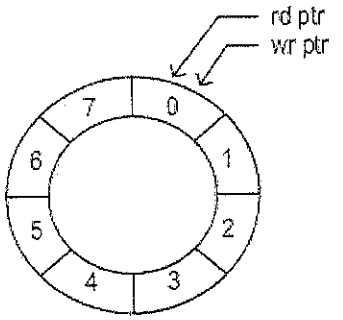
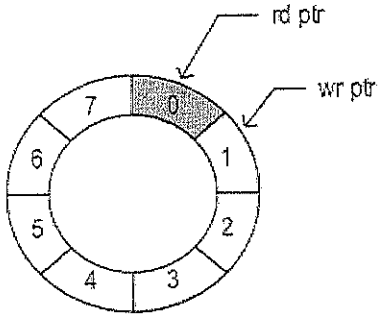


Figure 9.11 Conceptual diagram of a FIFO buffer.

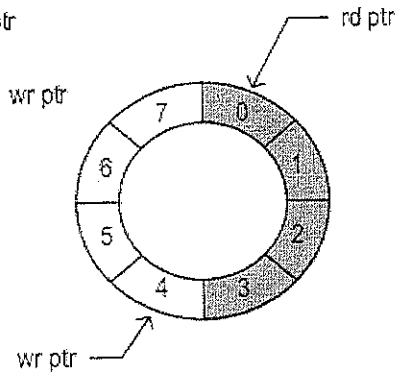
- Circular queue implementation
- Use two pointers and a "generic storage"
  - Write pointer: point to the empty slot before the head of the queue
  - Read pointer: point to the tail of the queue



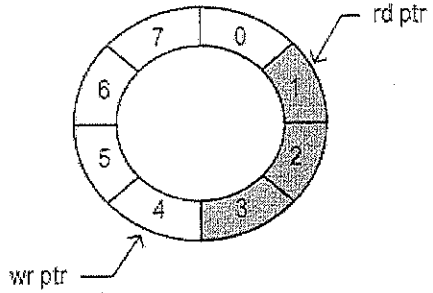
(a). initial (empty)



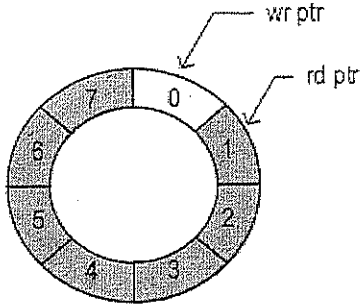
(b). after a write



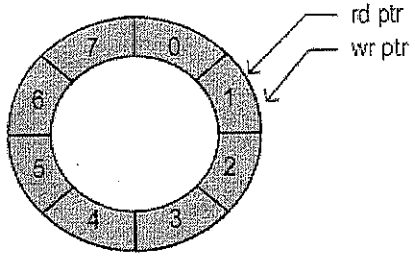
(c). 3 more writes



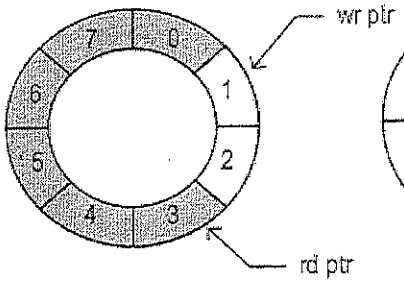
(d). after a read



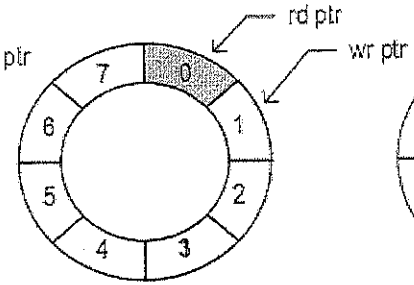
(e). 4 more writes



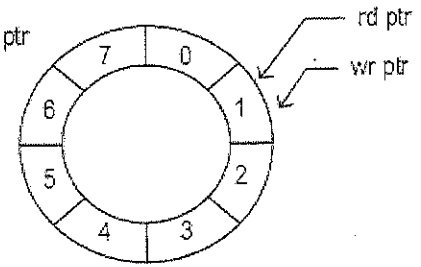
(f). 1 more write (full)



(g). 2 reads



(h). 5 more reads



(i). 1 more read (empty)

FIG 9.12

- FIFO controller
  - Read and write pointers: 2 counters
  - Status circuit:
    - Difficult
    - Design 1: Augmented binary counter
    - Design 2: with status FFs
  - LSFR as counter

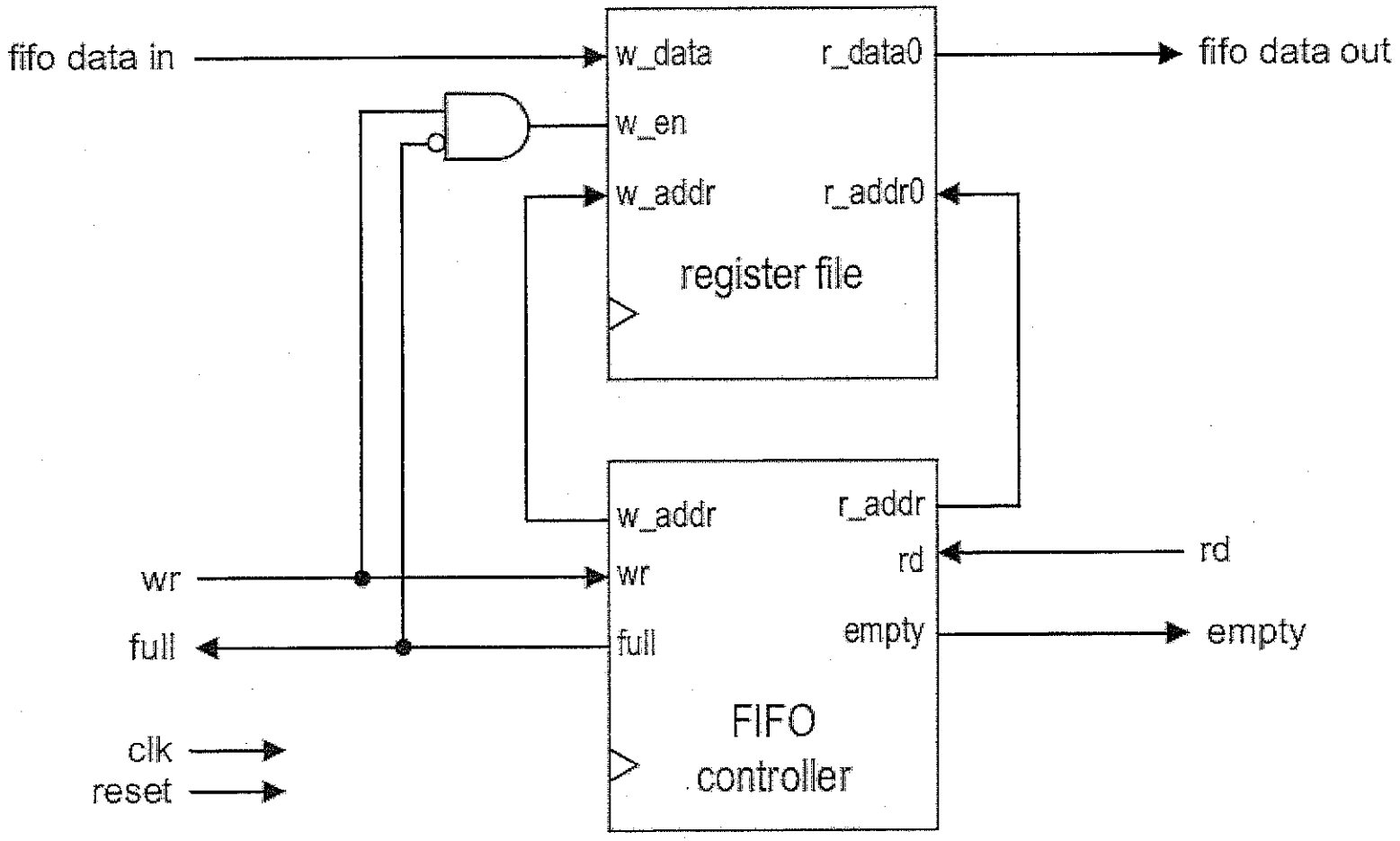

fifo data in
w_data
r_data0
fifo data out
w_en
w_addr
r_addr0
register file
w_addr
r_addr
rd
rd
wr
wr
full
full
empty
empty
FIFO
controller
clk
reset

fig 9.13

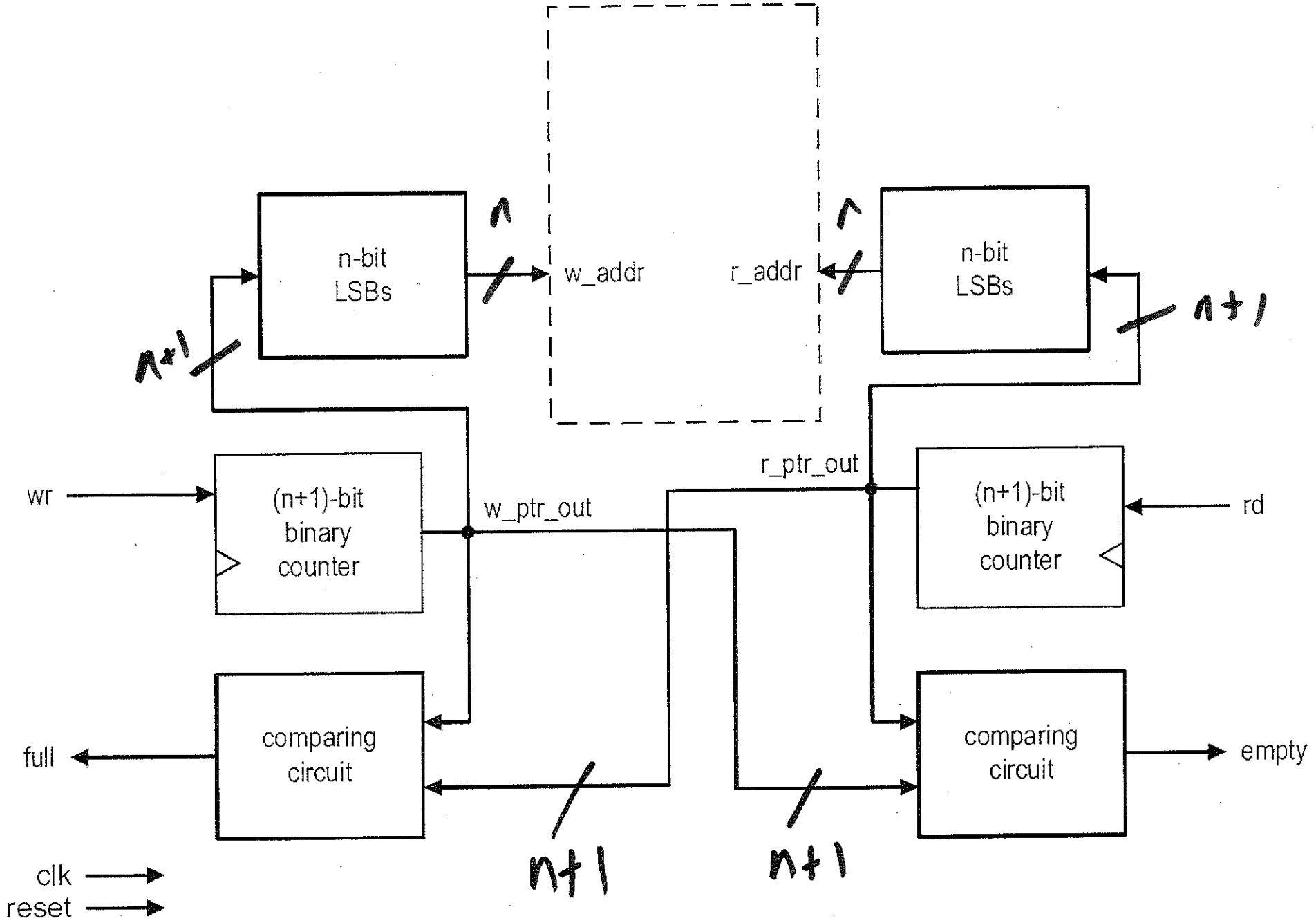


Figure 9.14 Detailed block diagram of an augmented-binary-counter FIFO control circuit.

Single Clock Domain (Sync)

- Augmented binary counter:
  - increase the counter by 1 bits
  - Use LSBs for as register address
  - Use MSB to distinguish full or empty

Write pointer	Read pointer	Operation	Status	MSB
0 000	0 000	initialization	empty	equal
0 111	0 000	after 7 writes		
1 000	0 000	after 1 write	full	DIFF
1 000	0 100	after 4 reads		
1 100	0 100	after 4 writes	full	DIFF
1 100	1 011	after 7 reads		
1 100	1 100	after 1 read	empty	equal
0 011	1 100	after 7 writes		
0 100	1 100	after 1 write	full	DIFF
0 100	0 100	after 8 reads	empty	EQUAL

Table 9.2

- 2 extra status FFs
  - Full_erg/empty_reg memorize the current staus
  - Initialized as 0 and 1
  - Modified according to wr and rd signals:
    - 00: no change
    - 11: advance read pointer/write pointer; full/empty no change
    - 10: advance write pointer; de-assert empty; assert full if needed (when write pointer=read pointer)
    - 01: advance read pointer; de-assert full; asserted empty if needed (when write pointer=read pointer)

- Non-binary counter for the pointer
  - Exact location does not matter as long as the write pointer and read pointer follow the same pattern
  - Other counters can be used for the second scheme
  - E.g, use LFSR

```
w_ptr_succ <=
   (w_ptr_reg(1) xor w_ptr_reg(0)) & w_ptr_reg(3 downto 1);
r_ptr_succ <=
   (r_ptr_reg(1) xor r_ptr_reg(0)) & r_ptr_reg(3 downto 1);
```