



---

# Data Transmission





# Data Transmission



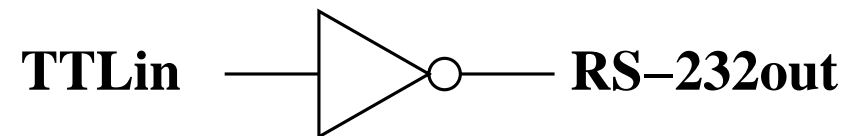
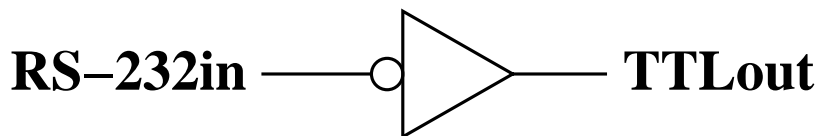
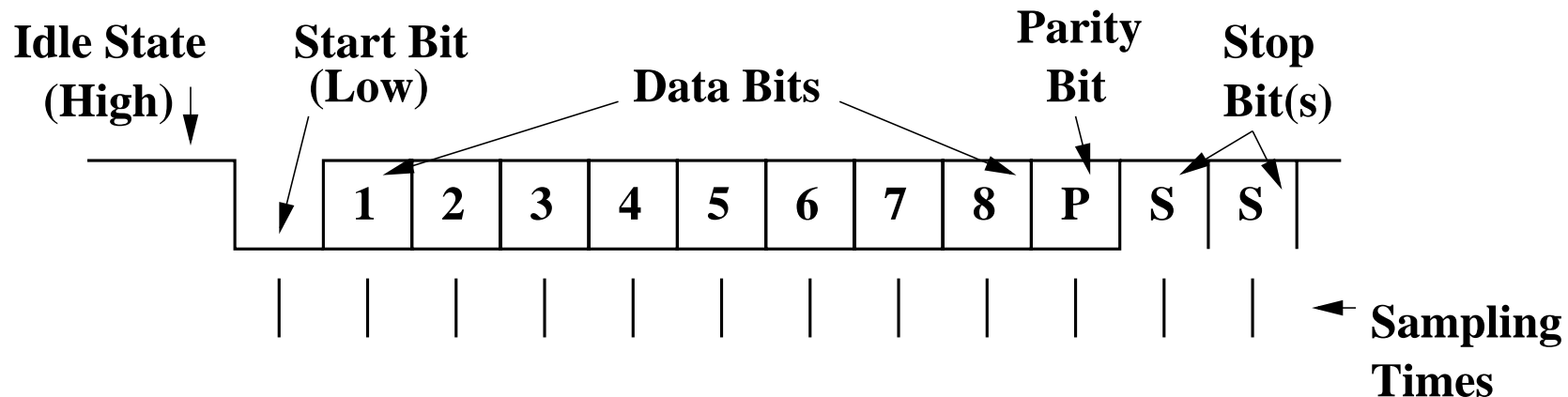
- **There are two basic methods.**
  - **Recover a clock from a serial transmission.**
    - **This requires a phase lock loop.**
    - **Examples:**
      - network data
      - disk
      - tape
      - GPS
      - Digital TV
    - **We will not do this in 6.111.**
      - FPGAs are hardwired to use a single clock.
  - **Synchronize incoming data to a local clock.**
    - **Serial data transmission**
      - **Uses only two wires (or radio and earth ground).**
      - **Slow – at most one bit per clock period**
    - **Parallel data transmission**
      - **Uses at least one wire per bit.**
      - **Fast – a word (n bits) at a time**
      - **Need to agree on control signals to know when data is stable**



# Serial Data Transmission

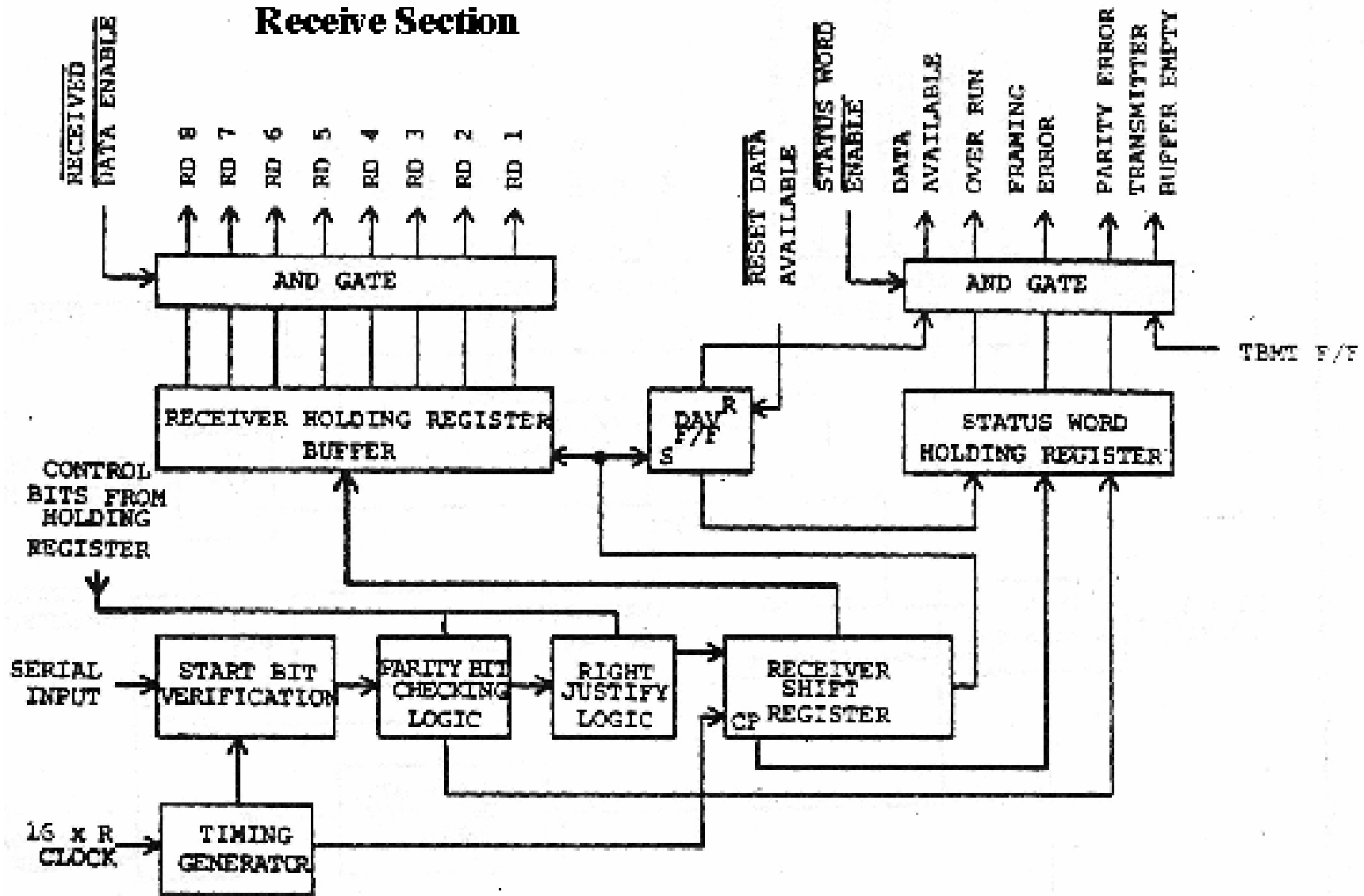


- **RS – 232 is a serial interface standard.**
  - RS – 232 levels are between
    - - 3v and - 15 v for a logic 1
    - + 3v and + 15 v for a logic 0
- **The TTL signal below has an “idle” state of a logic 1.**
  - MAXIM 233 has two RS – 232 to TTL and two TTL to RS – 232 level converters.

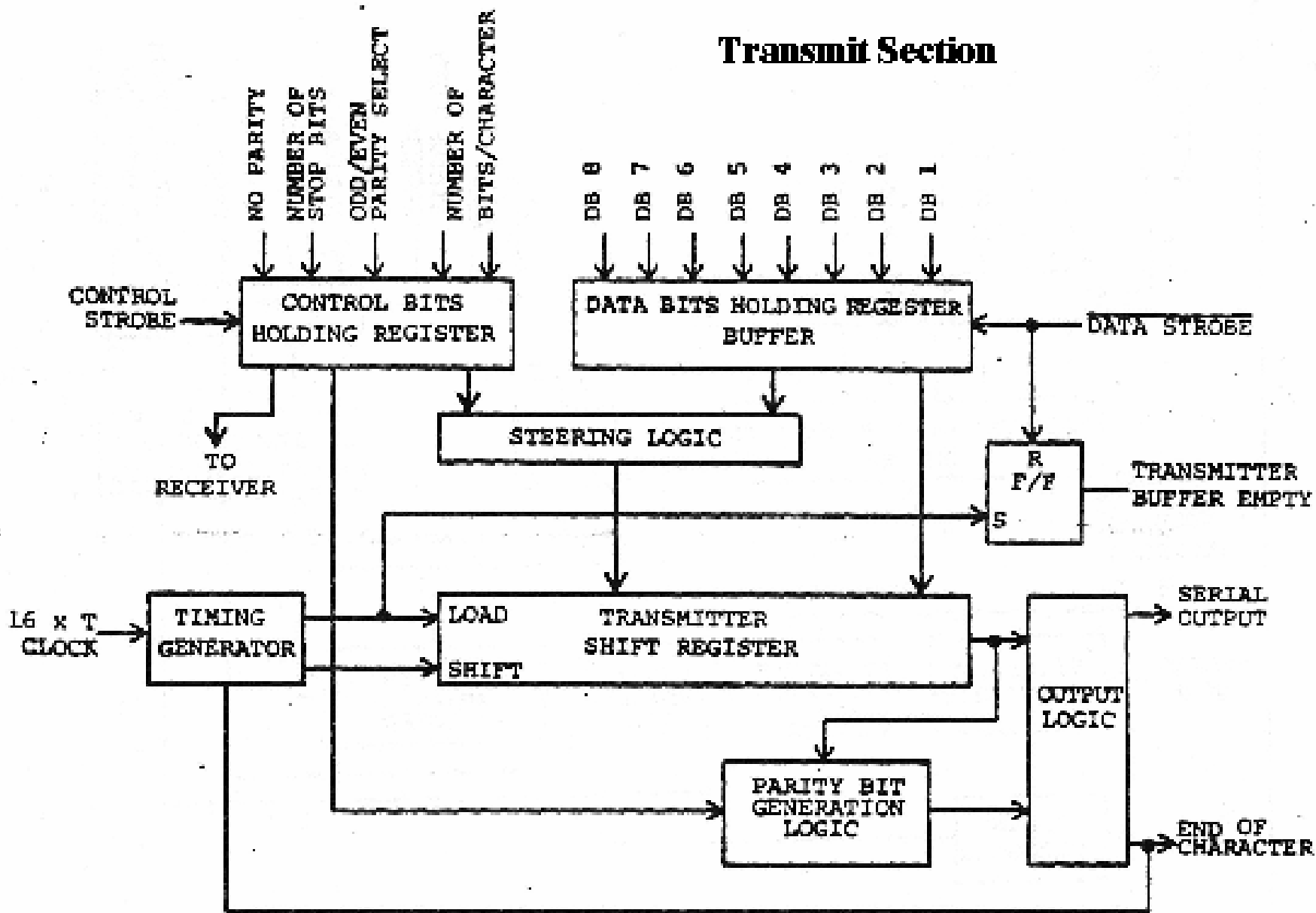


# AY 3 105D Receive Section

## Receive Section



# AY 3 105D Transmit Section

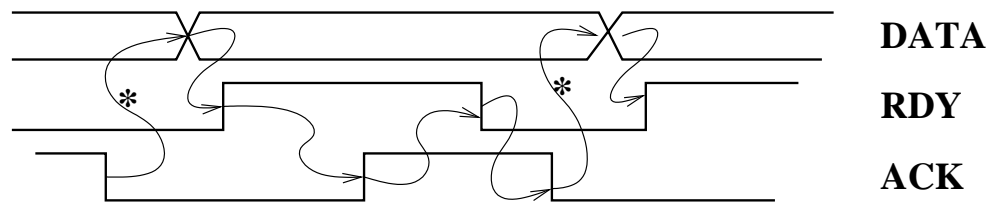




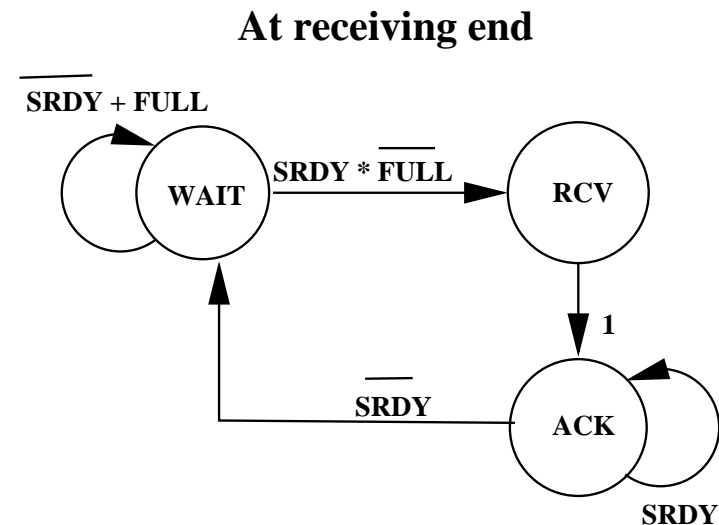
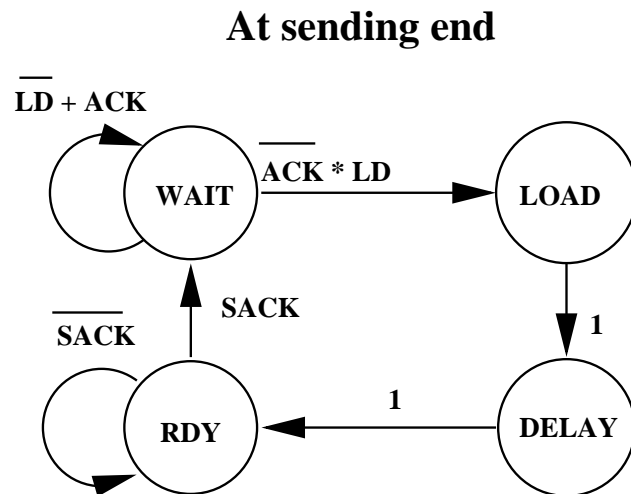
# Parallel Transmission



- Full Handshake – RDY and ACK return to zero.
  - Assumes data source and destination have unrelated clocks.
  - E.g., the two locations are on separate kits.
  - Why not send clock from one kit to another?



\* and we want to send more data, i.e., LD is true.





# Parallel Interface

