IEEE 802.4 Token-Bus

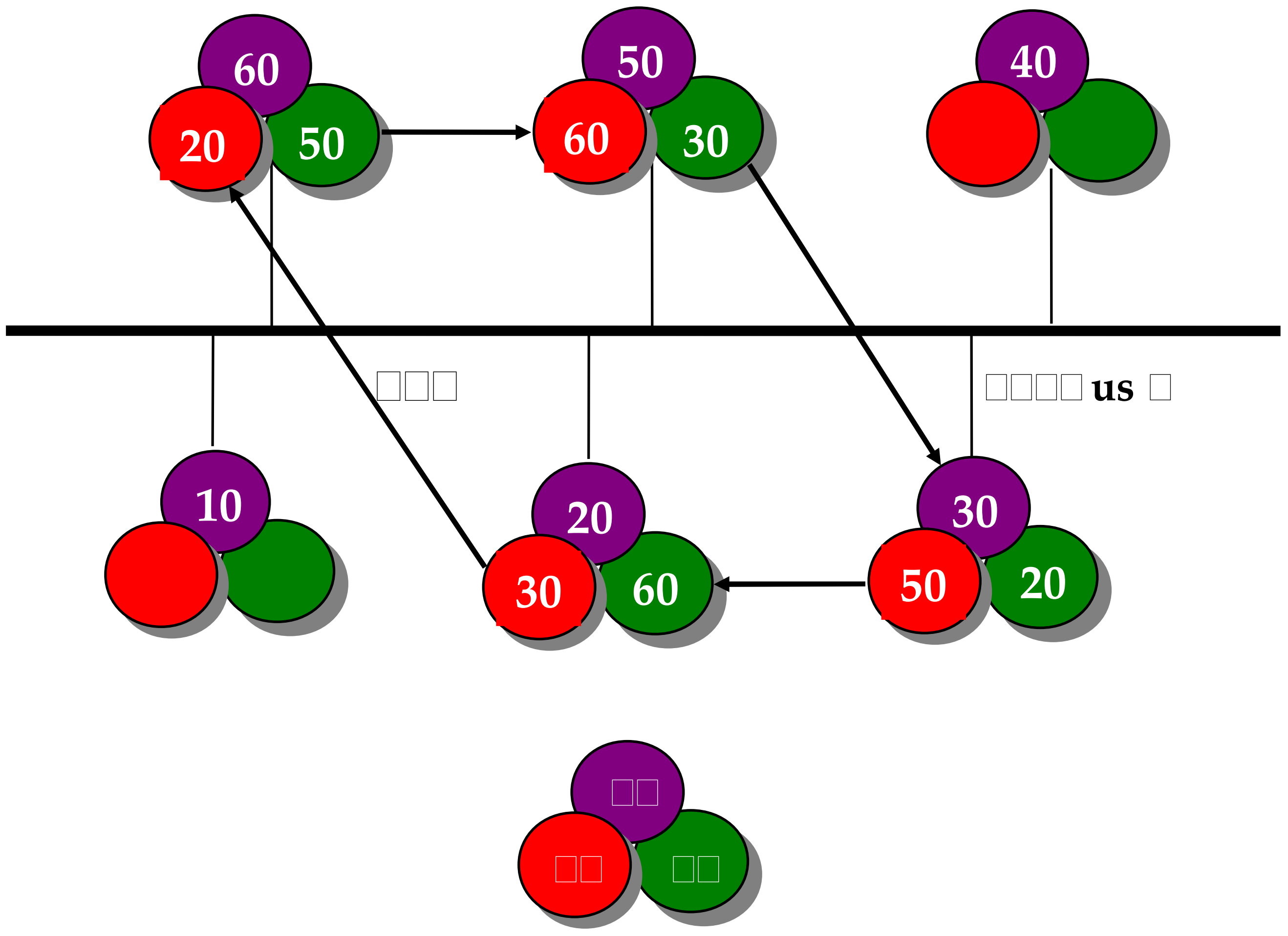
Token-Passing Bus Access Method

- **Physically, it is a Bus network. Logically, it is a Ring network**
- **Stations are organized as a circular doubly-linked list**
- **A distributed polling algorithm is used to avoid bus contention**
- **Token: Right of access**
- **Token Holder (The station receiving the token)**
  - **Transmit one or more MAC-frame**
  - **Poll other stations**
  - **Receive responses**

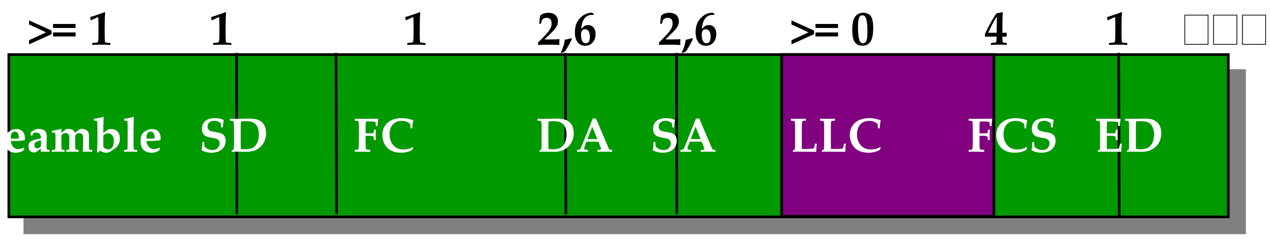
Token-Passing Bus Access Method

- **Non-Token Holder**
  - **Listen to the channel**
  - **Respond to a poll**
  - **Send Acknowledgement**
- **Logical Ring Maintenance**
  - **Ring Initialization**
  - **Addition to ring**
  - **Deletion from ring**
  - **Error Recovery**

Token-Bus Network Example

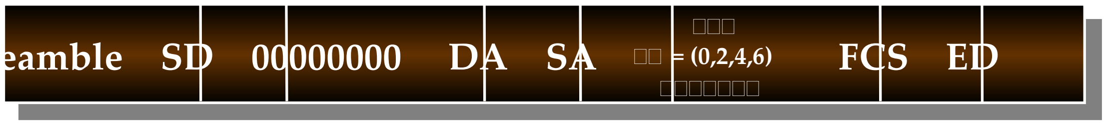


Frame Format



- Preamble: Establish bit synchronization and locate the first bit of the frame
- SD = Start Delimiter
- FC = Frame Control
- DA = Destination Address
- SA = Source Address
- FCS = Frame Check Sequence
- ED = End Delimiter
- Maximum Length (From SD to ED): 8191 Bytes

Specific IEEE 802.4 Frame Formats



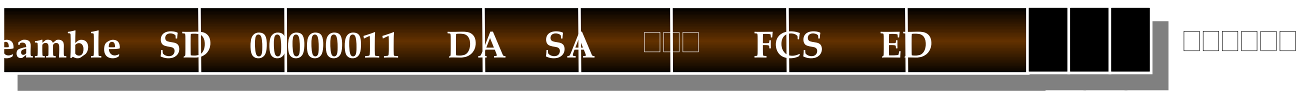
(a) □□ - □□ (Claim Token, CT). The frame has a data-unit whose value is arbitrary and whose length in octets (between addresses and FCS exclusive) is 0,2,4, or 6 times the system's slot-time also measured in octets.

(b) □□ - □□□□ -1 (Solicit-Successor-1, SS1). The frame has a DA= the contents of the station's NS register and a null data unit. One response window always follows this frame.

- Response Window: One slot time (Round-Trip propagation delay). The Token holder expects response or acknowledgement from other stations in a given number of Response Windows.
- Possible events in the response window
  - No response
  - One Response
  - Multiple Responses -- A garbled Response (Contention)

Specific IEEE 802.4 Frame Formats

(c) □□ - □□□□ -2 (Solicit-Successor-2, SS2). The frame has a DA= the contents of the station's NS or TS register and a null data unit. Two response windows always follow this frame.



(d) □ - □□□□ (Who-follows, W). The frame has a data-unit = the value of the station's NS register. The format and the length of the data-unit is the same as a source address. Three response windows always follow this frame. (This gives receivers two extra slot-time to make a comparison with an address other than TS.)

(e) □□ - □□ (Resolve-contention, R). The frame has a null data-unit. Four response windows always follow this frame.

Specific IEEE 802.4 Frame Formats

eamble	SD	00001000	DA	SA	FCS	ED

(f) □□ (Token, T). The frame has DA = the contents of the station's NS register, and has a null data-unit.

eamble	SD	00001100	DA	SA	□□□	FCS	ED

(g) □□ - □□□□ (Set-Successor, S). The frame has DA = the SA of the last frame received, and data-unit = the value of the station's NS or TS register. The format and the length of the data-unit is the same as a source address.

eamble	SD	01MMMPPP	DA	SA	LLC	FCS	ED

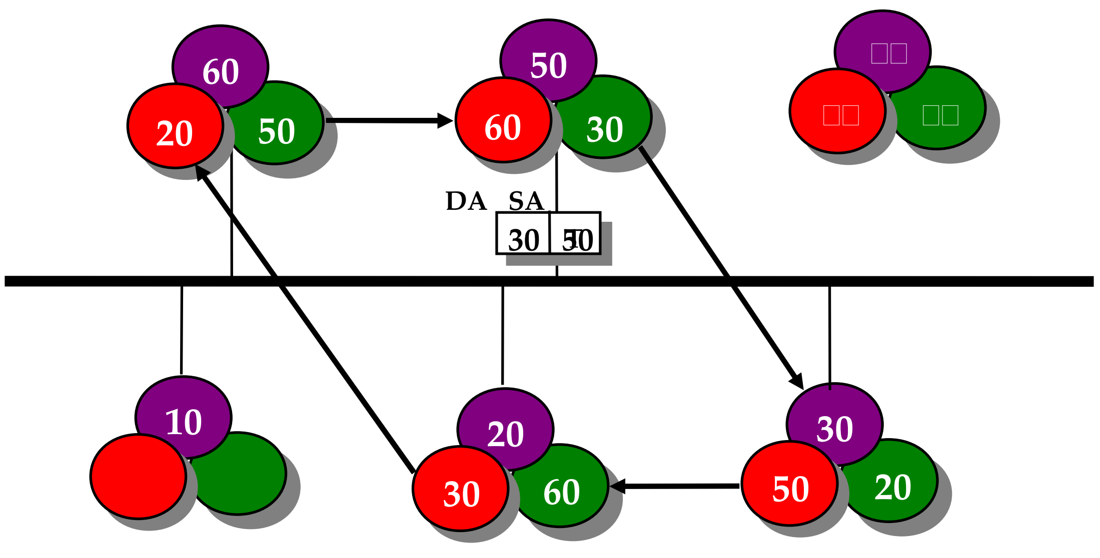
(h) LLC □□□□. LLC data frames have a DA and data-unit specified by a station's LLC sublayer. A frame of this type with a non-null data unit shall be passed to the receiving station's LLC sublayer.

Normal Token Passing Operation

- **In numerically descending order (60 -> 50 -> 30 -> 20 -> 60)**
- **Each station in the ring has knowledge of**
  - **TS: This Station's address (□□)**
  - **NS: Next Station's address (□□)**
  - **PS: Previous Station's address (□□)**
- **A station has a token holding timer to limit the time it can hold the token. This value is set at the system initialization time by the network management process.**
- **At the end of transmission, the token is passed to the next station.**
- **Once received the token, the station either starts to transmit or passes the token to the next station within one response window.**

Normal Token Passing Example

- The Logical ring consists of stations 60,50,30,20. Station 10 is not included in the logical ring.
- Station 50 passes a token to station 30 via the broadcast bus.
- All the stations in the bus can see the token, but only station 30 has the right to use the token.



Addition of a Station

- **Token holder has the responsibility of periodically (an inter-solicit-count timer) granting an opportunity for new stations to enter the logical ring before it passes the token.**
- **A Solicit-Successor-1 (SS1) control frame is issued with**
  - **DA = NS**
  - **SA = TS**
  - **Data = Null**
- **One response window is reserved for those stations desired to enter the logical ring and their address is between DA and SA. (If the address of the token-holder has the smallest address in the logical ring, then a Solicit-Successor-2 frame is issued)**

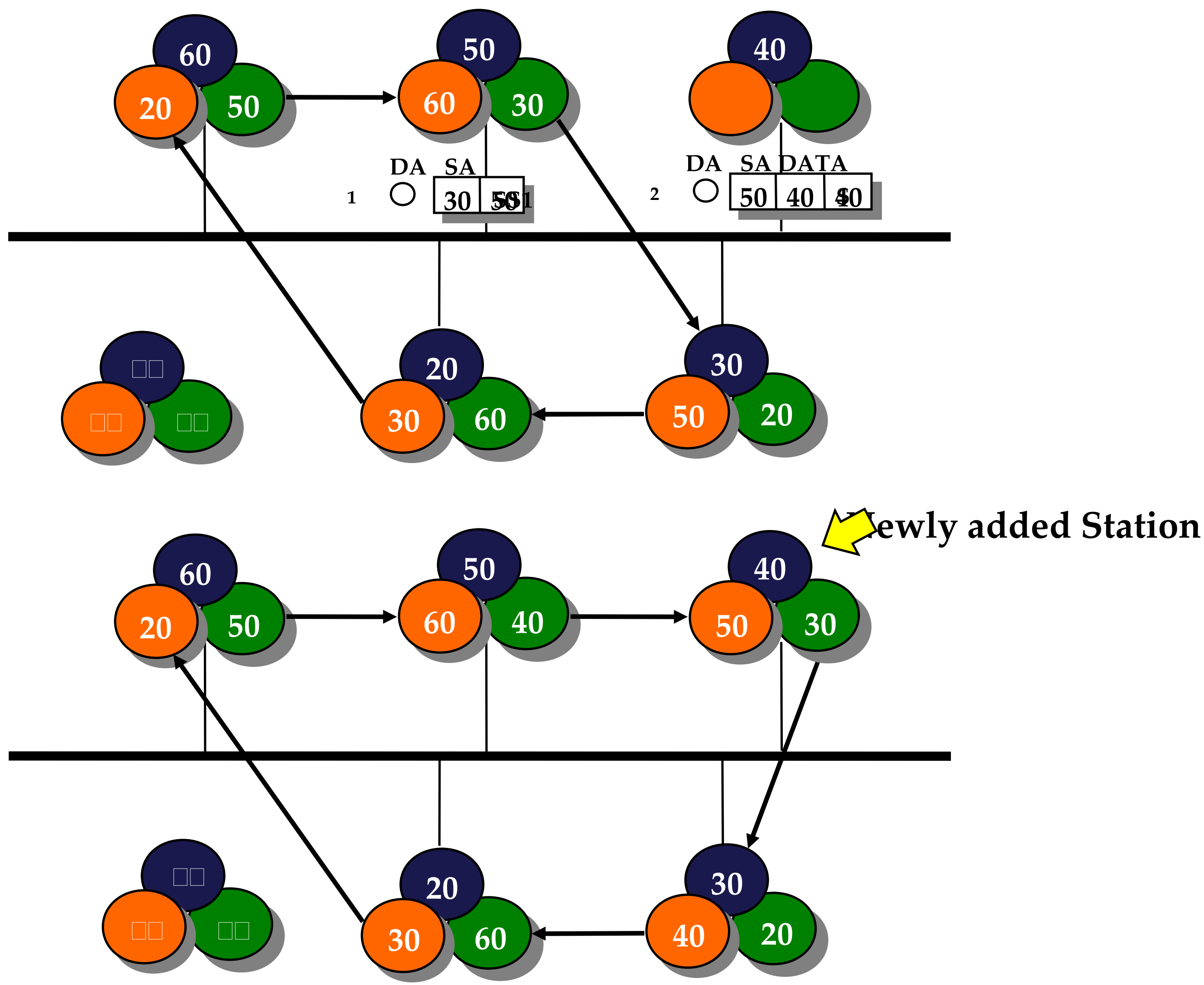
Addition of a Station

- **The station desired to enter the logical ring will respond with a Set-Successor frame with**
  - **DA = Token-holder address**
  - **SA = TS (Its address)**
  - **Data = TS**
- **The Token holder detects the event in the response window and takes appropriate actions:**
  - **No Response: Pass token to the next station**
  - **One Response: Pass token to the newly added station. The newly added station will update its NS value by recall the DA field of the previously received Solicit-Successor-1 frame.**

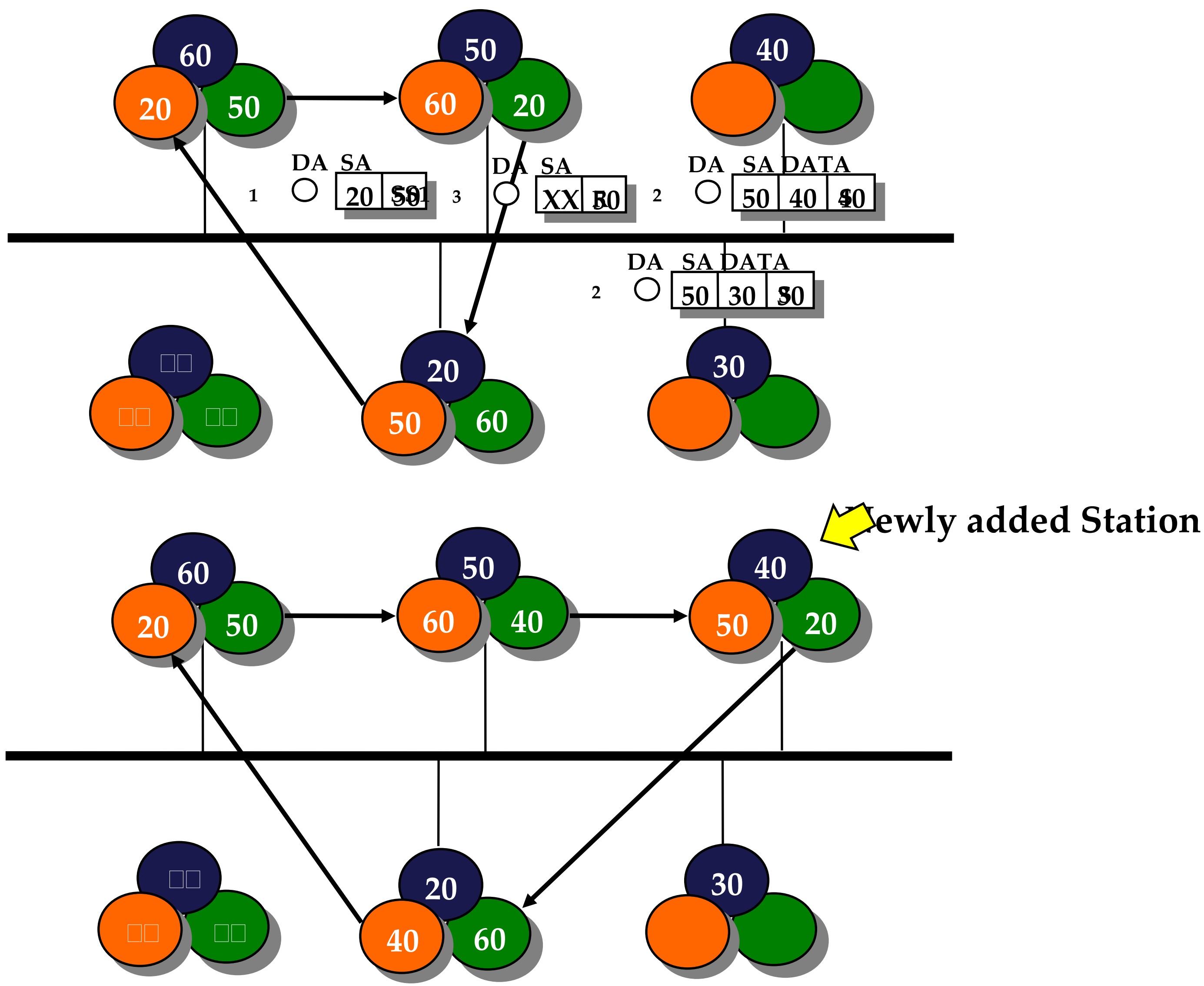
Addition of a Station (Continued)

- **Multiple Responses:**
  - A Resolve-Contention frame is issued by the token-holder with
    - ✓ DA = XX, SA = TS, Data = Null
  - A station desired to enter the logical ring will response with a Set-Successor frame as before at the K-th window, where K is determined by the value of the first two bits of its address. However, if the channel is detected busy before the K-th window, it will give up.
  - If no valid Set-Successor frame is received by the token-holder, the token-holder will issue another Resolve-Contention frame.
  - Now only those stations involved in the contention may try again. The value of K now is determined by the next two bits.
  - The above procedure is repeated until a valid Set-Successor frame is received by the token-holder. A new station is thus successfully added to the logical ring.

Addition Example 1 (without Contention)

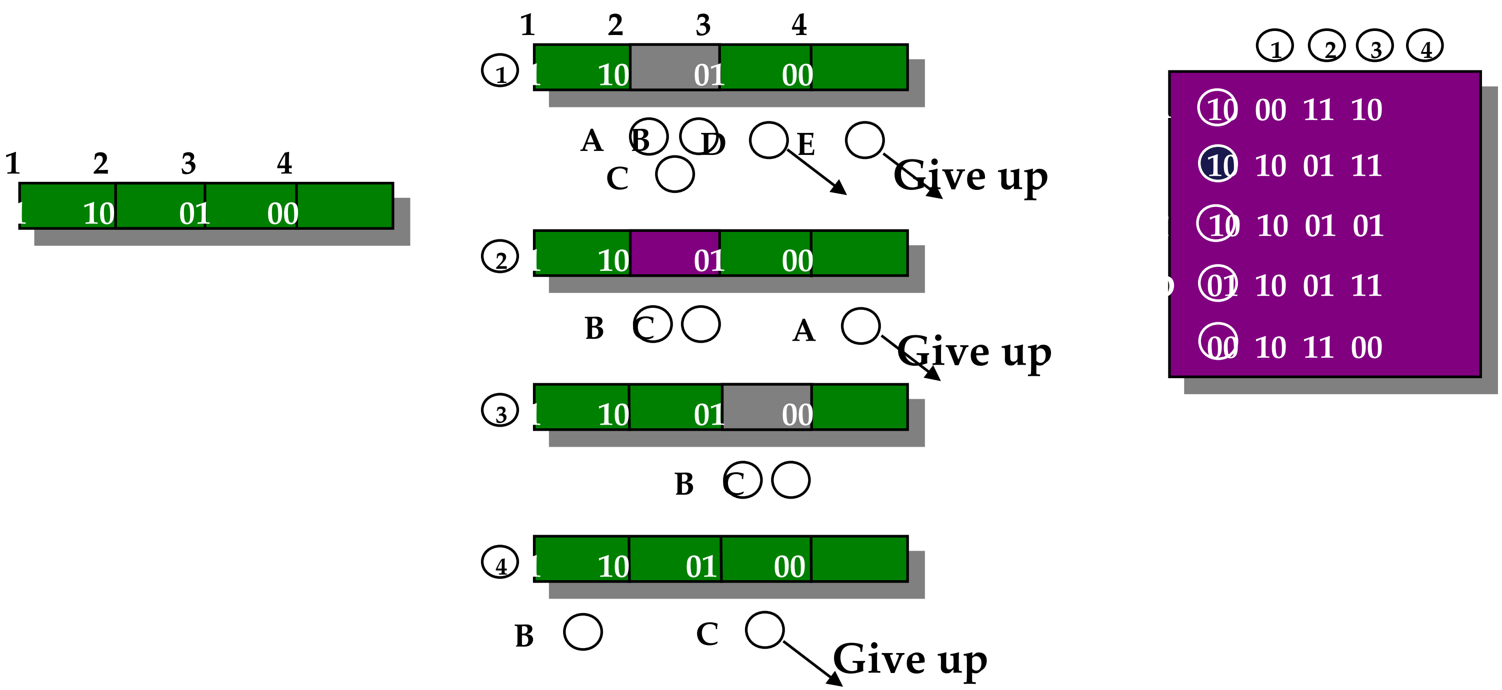


Addition Example 2 (with contention)



Resolve Contention Example

- Five stations: A,B,C,D,E. For simplicity, addresses are assumed to have only 8 bits.
- Four Resolve-Contention frames are required to find out the station with the largest address (B). Each frame follows with four Response Windows.



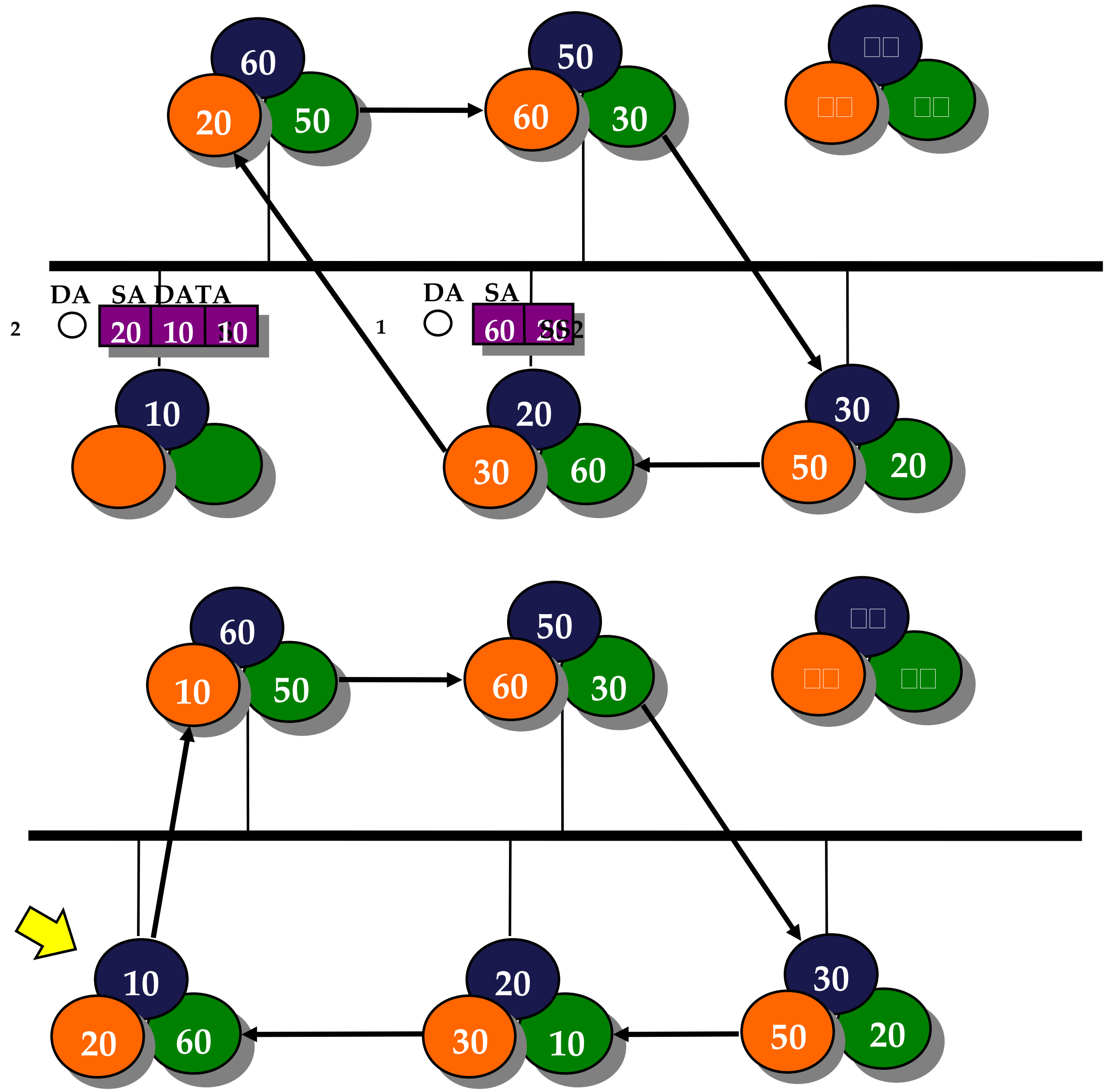
Addition of a Station (Continued)

- **Usually, NS < TS. But there is one station whose NS > TS (The one with the smallest address in the logical ring). In this case Solicit-Successor-2 (SS2) frame is issued followed by two Response Windows.**
- **The first response window is reserved for stations (X) whose address is less than this station.**
- **The second response window is reserved for stations (Y) whose address is greater than its successor.**
  - **Stations in Y will respond with the Set-Successor frame only if there is no response in the first response window.**

Addition of a Station (Continued)

- **Each response window may have the following events:**
  - **No response**
  - **One response -- A newly station is added**
  - **Multiple responses -- Resolved contention procedure is invoked to find out the largest address involved in the contention.**
- **If both the response windows have no response, then the token-holder passes the token to the next station.**

Addition Example 3 (with SS2)

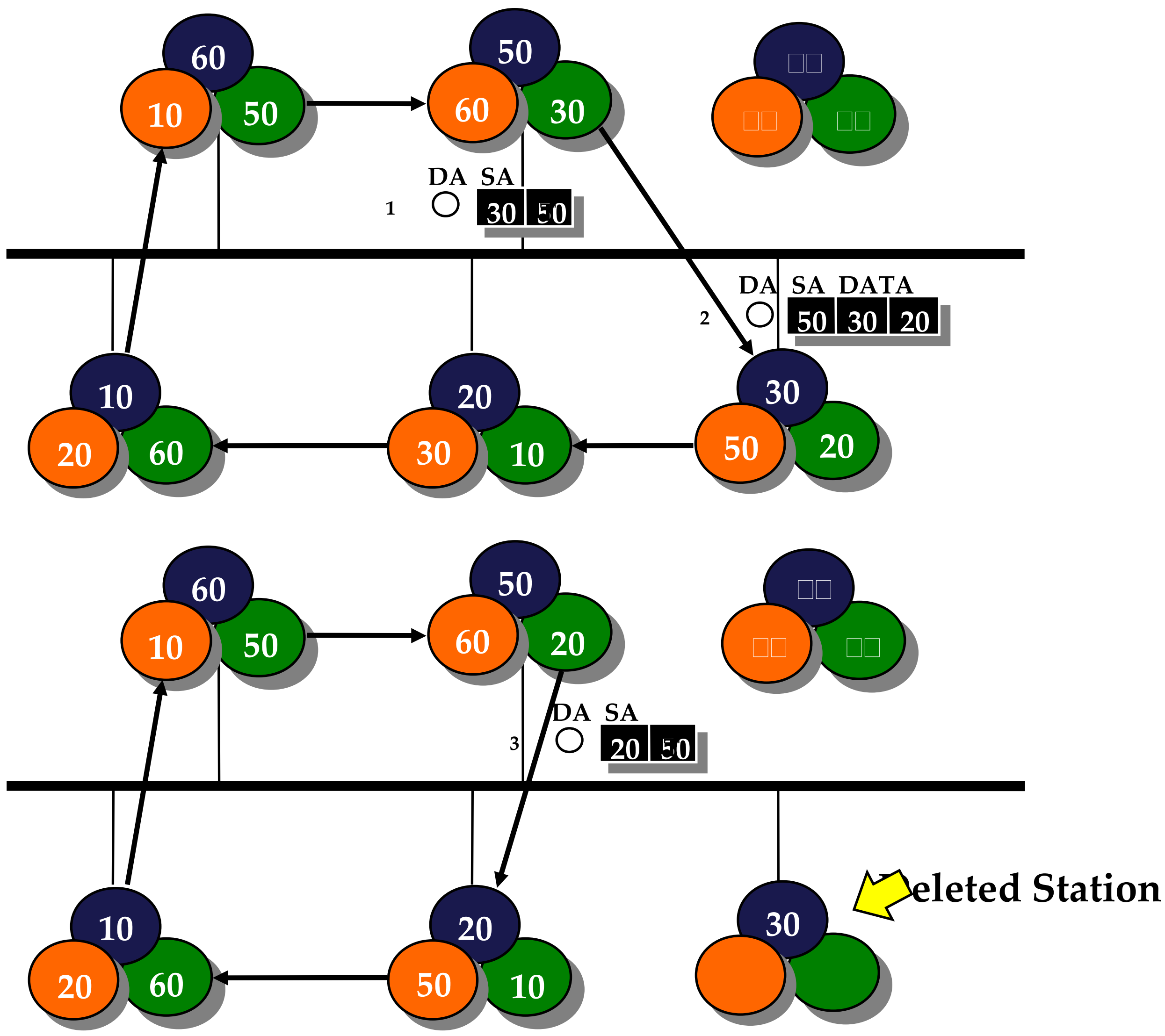


Newly added Station

Deletion of a Station

- **The station wishes to be deleted may wait until it receives the token, then sends a Set-Successor frame to its predecessor, with**
- **DA = PS**
  - **SA = TS**
  - **Data = NS**
- **The previous station once receives the Set-Successor frame will modify its NS and send a token to its new next station.**
- **The next station once receives the Set-Successor frame will modify its PS accordingly.**
- **After these two modifications, the station is removed from the logical ring automatically.**
- **If the station fails, it will not receive the token. This will be detected by the token-sender as explained later.**

Deletion Example



Fault Management

- **One of the most important issues of the token-bus protocol is to maintain the logical ring under the following possible conditions:**
  - **Multiple Tokens**
  - **Unaccepted Token**
  - **Failed Station**
  - **Failed Receiver**
  - **No Token**

Multiple Tokens

- **Cause:**
  - **Noise**
  - **Duplicate Address, each one may "receive" a token**
- **Detection:**
  - **While holding the token, the station may hear a frame on the bus which indicating that another station also has a token.**
- **Action:**
  - **Drop the token**
  - **If all stations drop the token, the network becomes the case of no token (see the procedure of handling no token later)**

Unaccepted Token or Failed Station

- **Cause**
  - **The token passed to the next station may be garbled**
  - **The next station fails**
- **Detection:**
  - **No response (Channel is idle) in one response window**
- **Action:**
  - **Try to pass token one more time**
  - **It still no response, then the next station is assumed to have failed**
  - **The token holder then issues a Who-Follows frame with**
    - ✓ **DA = XX, SA = TS, Data = NS**

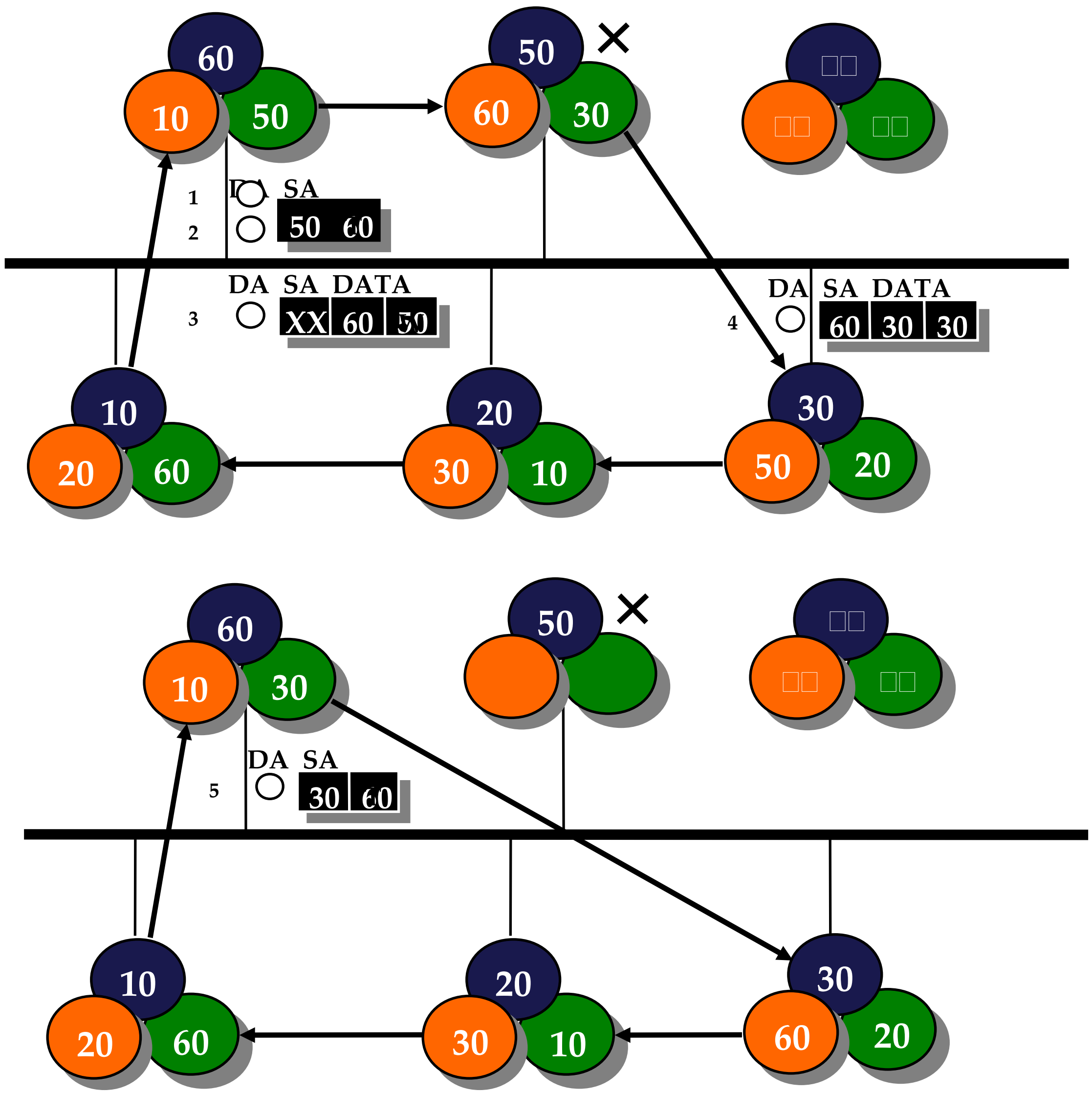
Unaccepted Token or Failed Station

- **All other stations once received Who-Follows frame will compare the data with its PS value. If there is a match, it will issue a Set-Successor frame back. Three response windows are reserved after Who-Follows. The first two are needed to make a comparison.**
- **If no response to the Who-Follows frame, the above procedure will be tried one more time.**
- **If still no response to the Who-Follows frame, then it could be that the next station to the next station has also failed.**
- **The token-holder will try to establish the ring by issuing a Solicit-Successor-2 frame, with**
  - ✓ **DA = TS, SA = TS, Data = Null.**

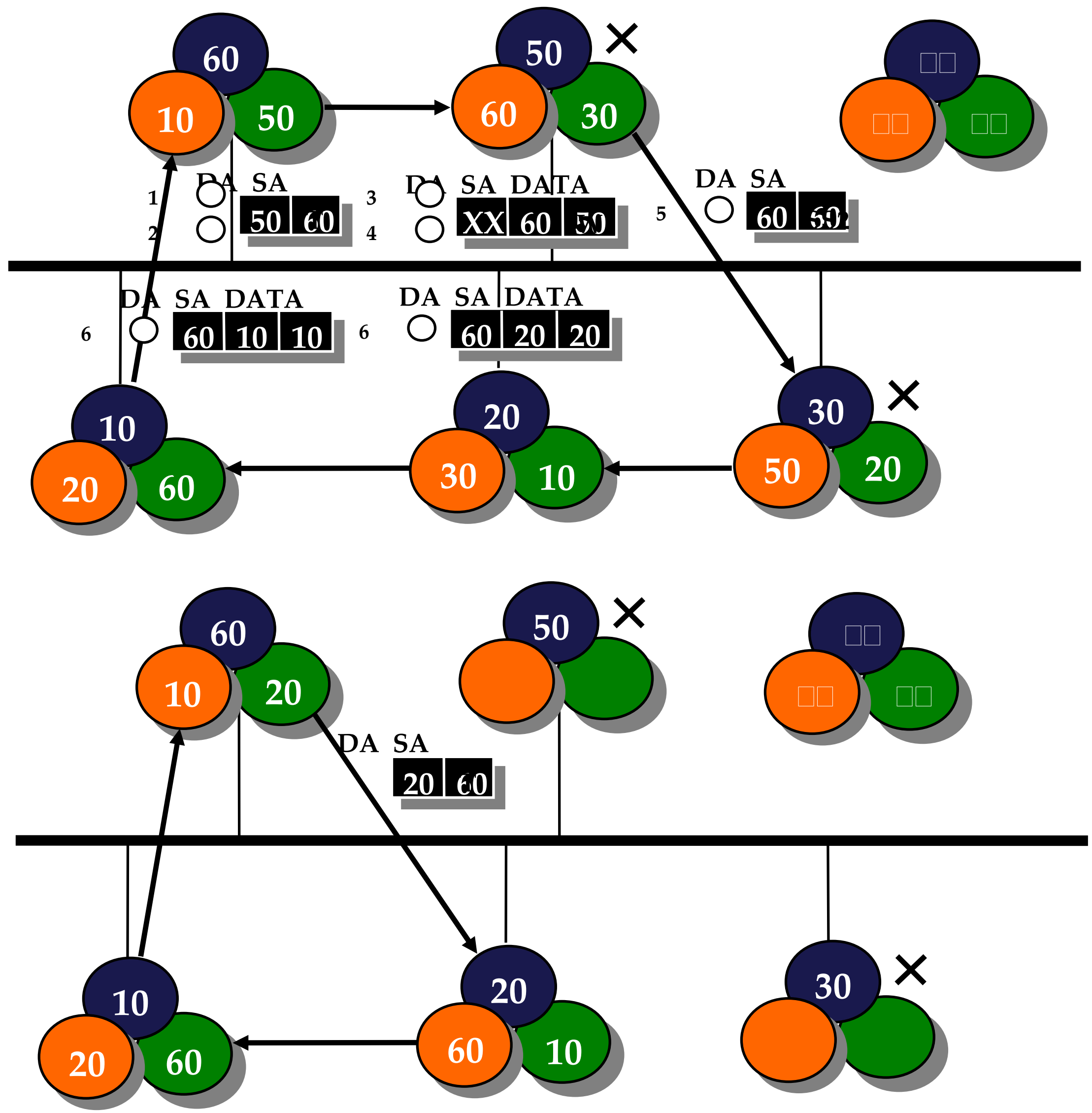
Unaccepted Token and Failed Station

- **DA=SA=TS implies that every station is invited to respond. Two response windows are reserved after this frame.**
- **The first response window is reserved for stations whose address is less than the sender.**
- **The second response window is reserved for stations whose address is greater than the sender.**
- **The procedure of add a station is then used.**
- **If still no response to the Solicit frame, then either all stations have failed (Left the ring) or its own receiver has failed (so it cannot listen).**
- **If the only one station has something to send, it sends the data. Then repeat the token passing process. Otherwise, listen to the channel.**

Station Fails Example 1 (Station 50)



Station Fails Example 2 (Stations 50 and 30)



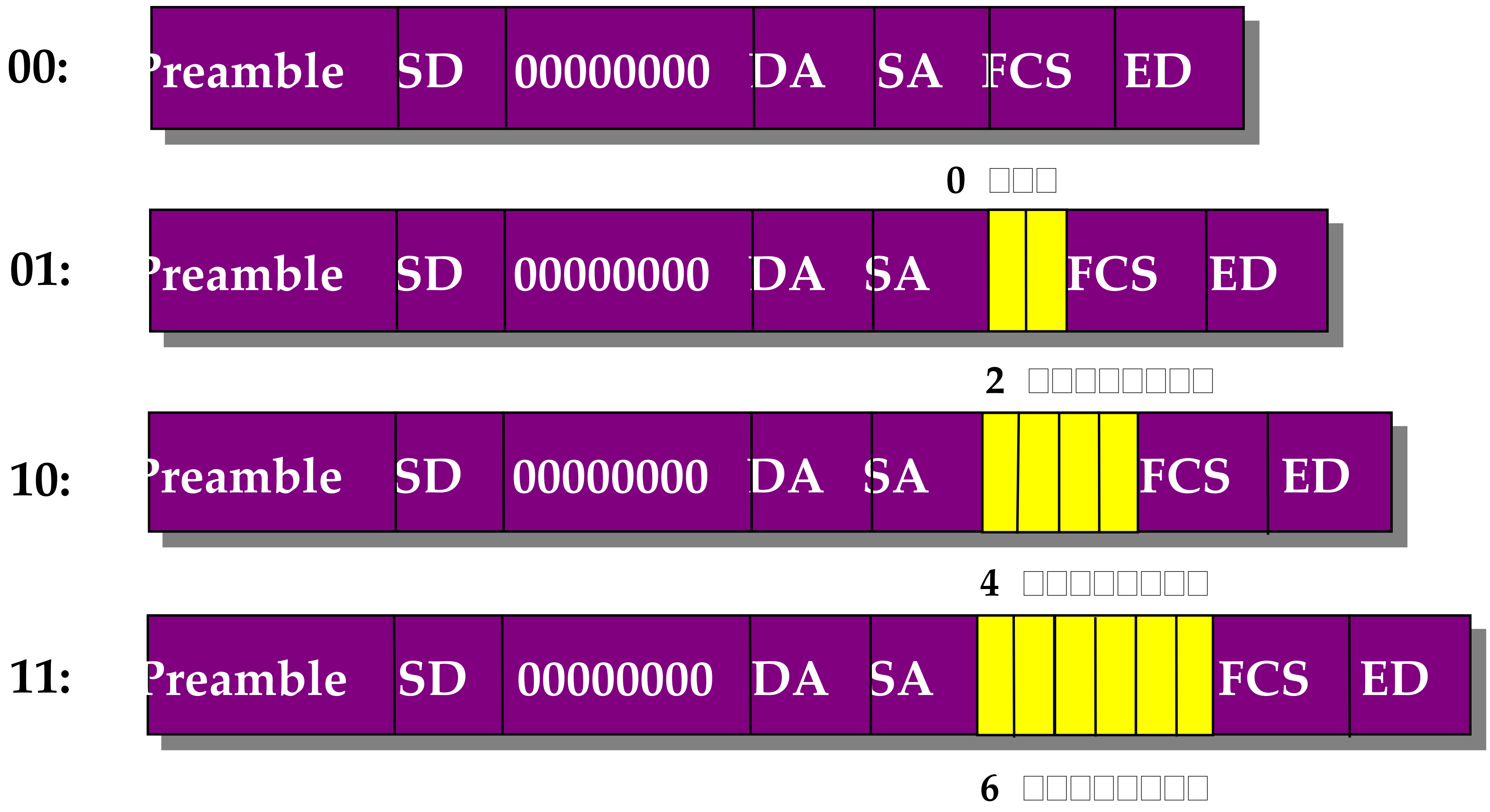
No Token or Initialization

- **Cause:**
  - **The Token-holder station fails**
  - **The token is destroyed**
  - **Network Initialization**
- **Detection:**
  - **No channel activity has been heard for a certain amount of time (Bus-Idle Timer expired)**
- **Action:**
  - **Any station when its Bus-Idle timer is expired will issue a Claim-Token frame, with**
    - ✓ **DA = XX, SA = TS, Data = Any value with (0,2,4,6) slot times depending on its address**

No Token or Initialization

- **The station with the greatest address will get the token. This is done by comparing the address. Two bits of the address are compared at a time.**
- **In each pass, only those stations who transmitted the longest frame on the previous pass try again.**
- **The station that succeeds on the last pass considers itself the token holder.**
- **The difference is 2 slots in the frame padding. The station waits one slot for its or other frame to pass. It then samples the channel at the second slot.**
- **The logical ring can then be established by issuing Solicit-Successor frames as described before.**

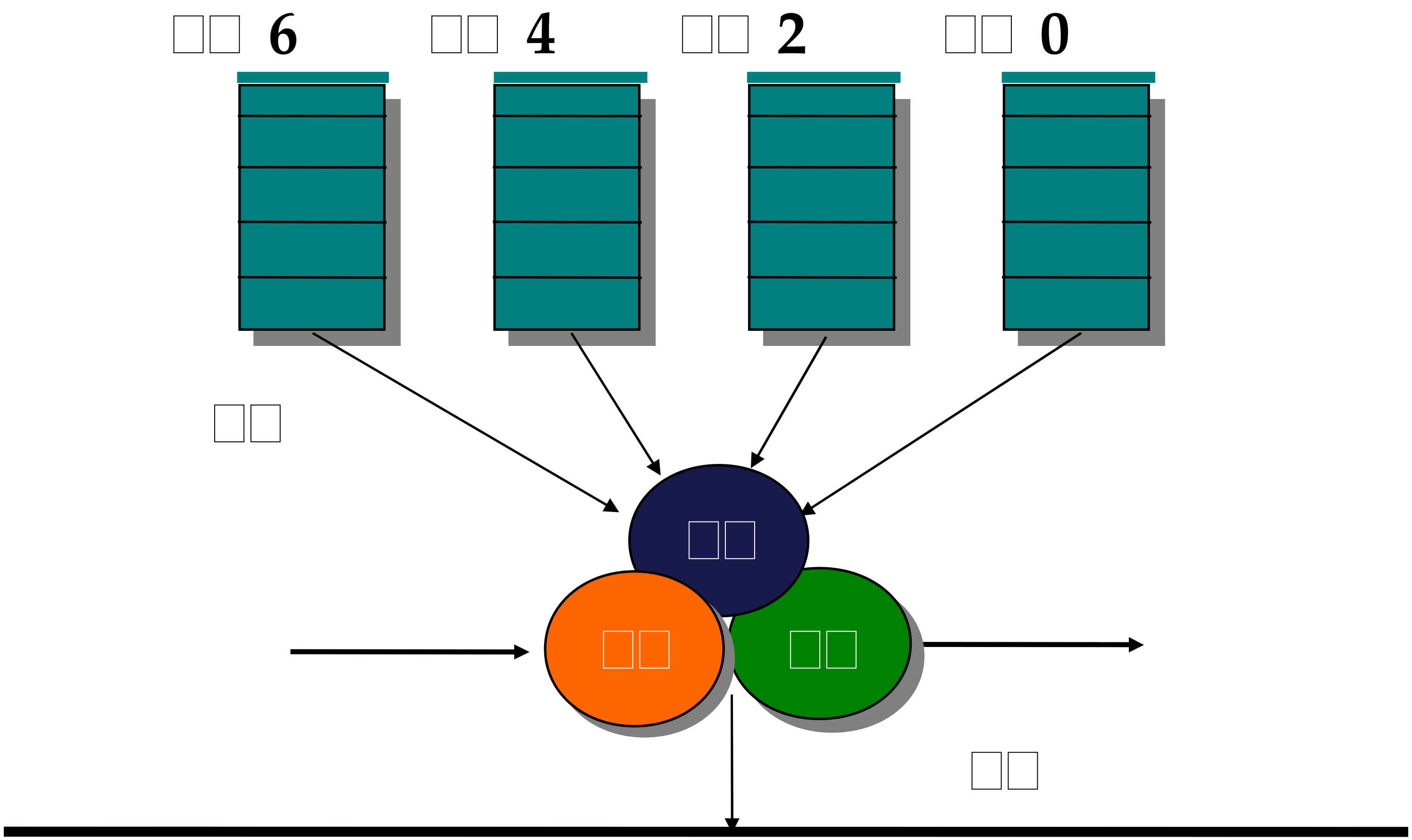
No Token or Initialization



Priority Mechanism

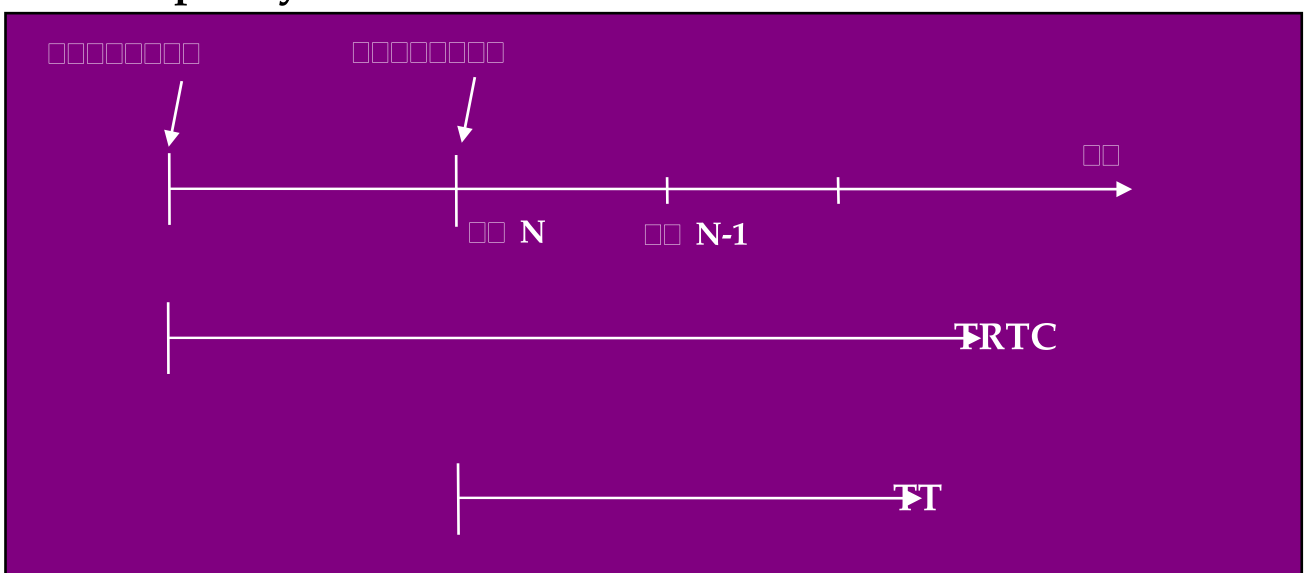
- **In the control field of data frame, three bits are reserved to indicate frame priority.**
- **Only four access classes are considered**
  - **6 : The highest priority**
  - **4**
  - **2**
  - **0 : The lowest priority**
- **Hi-Pri-Token-Hold-Time: To avoid one station dominating the network, an upper bound is set in each station to determine the maximum time that the highest priority frame can hold the token.**
- **Each lower access class in the station has a Target token Rotation Time (TRT).**

Priority Mechanism

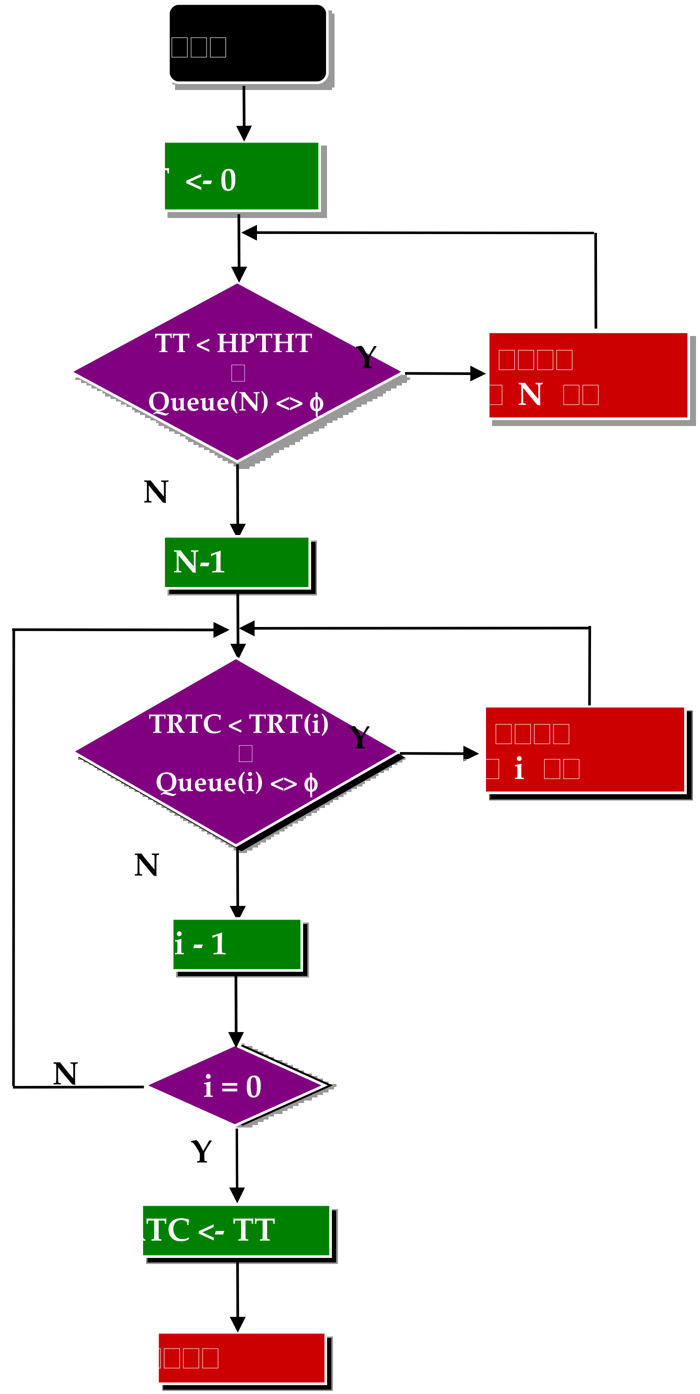


Priority Handling of Frames

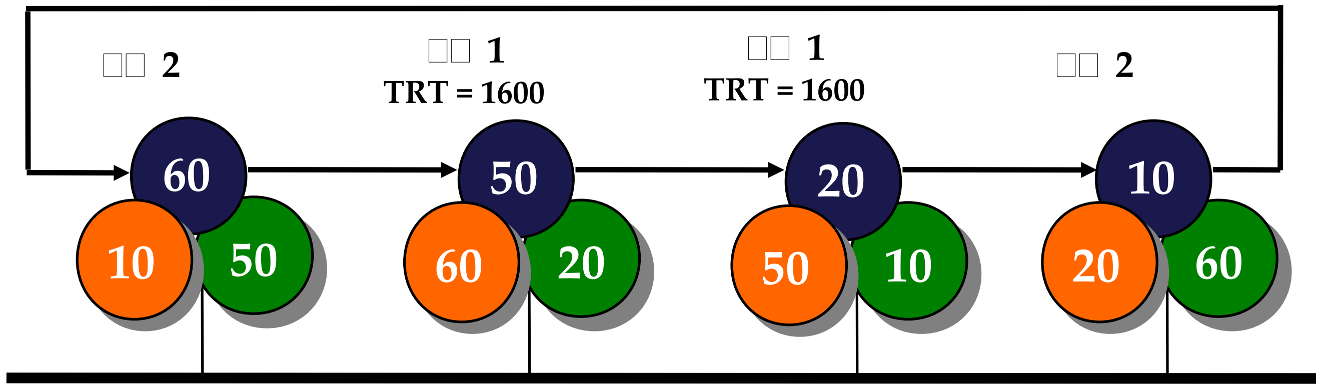
- **Some Timers set by the network management**
  - **Class N: HPTHT (Hi-Pri-Token-Hold-Time)**
  - **Class N-1: TRT(N-1) Token-Rotation-Time**
  - **...**
  - **Class 1: TRT(1)**
  - **TRTC: Token-Rotation-Time-Counter**
  - **TT: Temporary Timer**



General Priority Algorithm



Priority Algorithm Example



□□ □□□□	TRTC	□□□□	TRTC	□□□□	TRTC	□□□□	TRTC	□□□□
1	76	3	460	3	1660	0	1660	2
2	2270	3	2270	0	1070	2	1782	2
3	1782	3	1782	0	1782	0	1070	2
4	1070	3	1070	2	1870	0	1870	2
5	1870	3	1870	0	1070	2	1782	2
6	1782	3	1782	0	1782	0	1070	2
7	1070	3	1070	2	1870	0	1870	2
8	1565	3	1565	1	1165	2	1877	1
9	1877	3	1877	0	1477	1	1121	1

Priority Algorithm Example

- N = 2
- Station 60: Frame-size = 128 Bytes, Transmit three Class-2 frames at a time
- Station 50: TRT(1) = 1600, Frame-size = 400 Bytes, Transmit as many Class-1 frames as possible
- Station 20: TRT(1) = 1600, Frame-size = 356 Bytes, Transmit as many Class-1 frames as possible
- Station 10: Frame-size = 305 Bytes, Transmit two Class-2 frames at a time
- Token passing time = 19 bytes
- Initially all stations are idle
- TRTC Calculation:
  - Station 50: (Token Rotation no. 6), 1782 = 2(356)+2(305)+3(128)+4(19) > 1600
  - Station 20: (Token Rotation no. 9), 1477 = 2(356)+1(305)+3(128)+4(19) < 1600

Encoding: Differential Manchester Encoding