10
Embedded Communication Protocols

Distributed Embedded Systems

Philip Koopman

October 3, 2011

Carnegie Mellon

© Copyright 2010-2011, Philip Koopman

Test #1 Grading

- **Scores will be posted on blackboard**

- **When tests are available for pickup:**
  When you pick up your test packet, *make sure you pick up all five of the following items*:
  1. Test booklet
  2. Answer sheet (the one with your answers on it)
  3. Answer key (the one with the correct answers on it)
  4. Your note sheet
  5. Master question map (to map your test ordering into a master in case of questions)

- **If you want to submit your test for regrading, you must submit the entire packet (all five items) to Shannon Lown in the course hub and send an email to the 649 staff email list explaining what you would like regraded.** ***If you do not submit all five items, we will not regrade your test.***

Where Are We Now?

- **Where we've been:**
  - Design
  - Distributed system intro
  - Reviews & process
  - Testing

- **Where we're going today:**
  - Intro to embedded networking
    - If you want to be distributed, you need to have a network!

- **Where we're going next:**
  - CAN (a representative current network protocol)
  - Economics
  - Scheduling
  - …

Preview

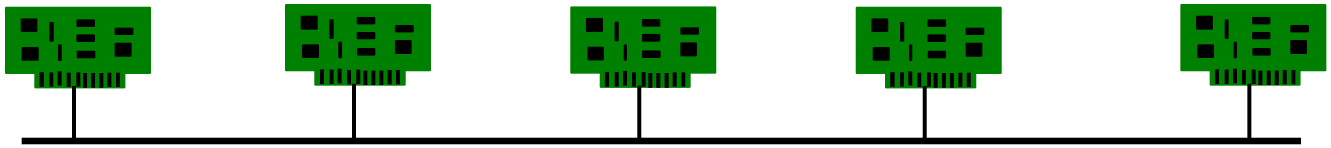
- **“Serial Bus”**
  = “Embedded Network”
  = “Multiplexed Wire” ~= “Muxing”
  = “Bus”

- **Getting Bits onto the wire**
  - Physical interface
  - Bit encoding

- **Classes of protocols**
  - General operation
  - Tradeoffs
    (there is no one “best” protocol)
  - Wired vs. wireless

Linear Network Topology

- **BUS**

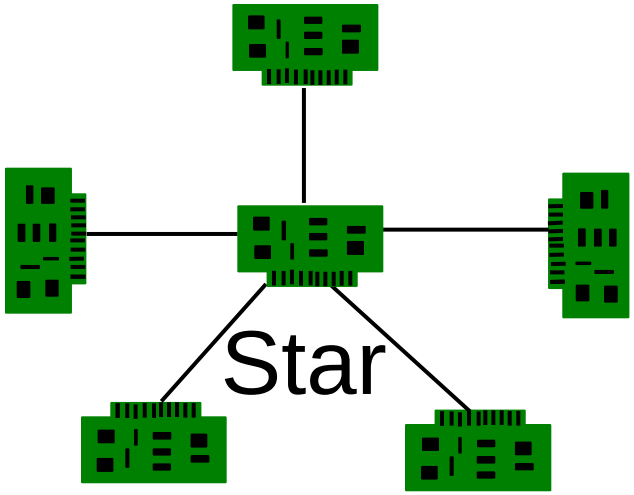


  - Good fit to long skinny systems
    - elevators, assembly lines, etc...
  - Flexible - many protocol options
  - Break in the cable splits the bus

  - May be a poor choice for fiber optics due to problems with splitting/merging
  - Was prevalent for early desktop systems
  - Is used for most embedded control networks

Star Network Topologies

- **Star**
  - Can emulate bus functions
    - Easy to detect and isolate failures
    - Broken wire only affects one node
    - Good for fiber optics
    - Requires more wiring; common for current desktop systems
  - Broken hub is catastrophic
  - Gives a centralized location if needed
    - Can be good for isolating nodes that generate too much traffic

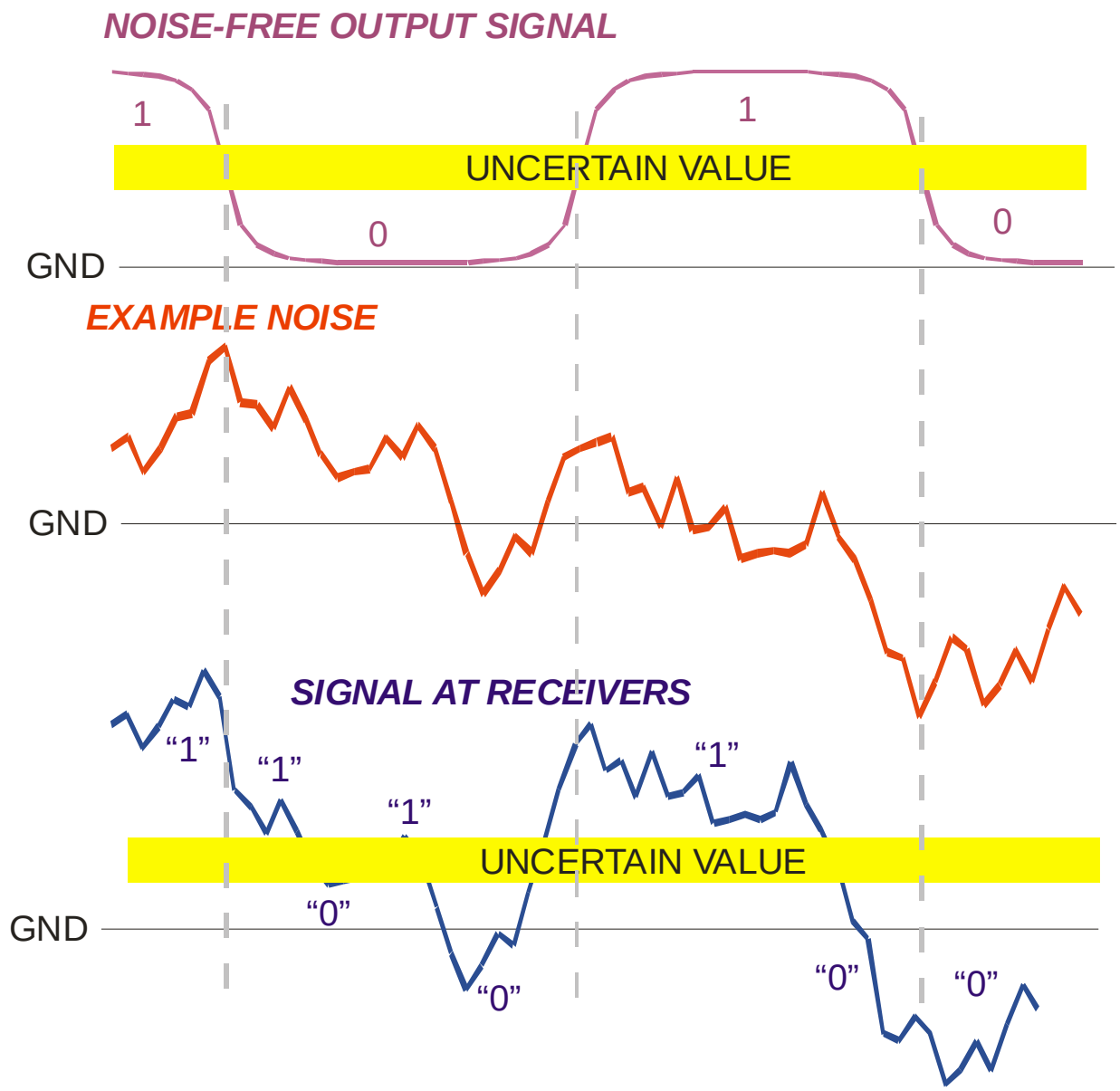


- **Star topologies increasing in popularity**
  - Bus topology has startup problems in some fault scenarios
  - Safety critical control networks moving to dual redundant star (Two independent networks, each network having star topology)

Hardware Connection Techniques

- **Circuits need to assert "HI" and "LO" on a physical bus**
  - Example:
    HI = 5 volts
    LO = 0 volts

- **Noise immunity is important**
  - Isolate noise on any single node from carrying over to network
  - Prevent noise on network from affecting nodes



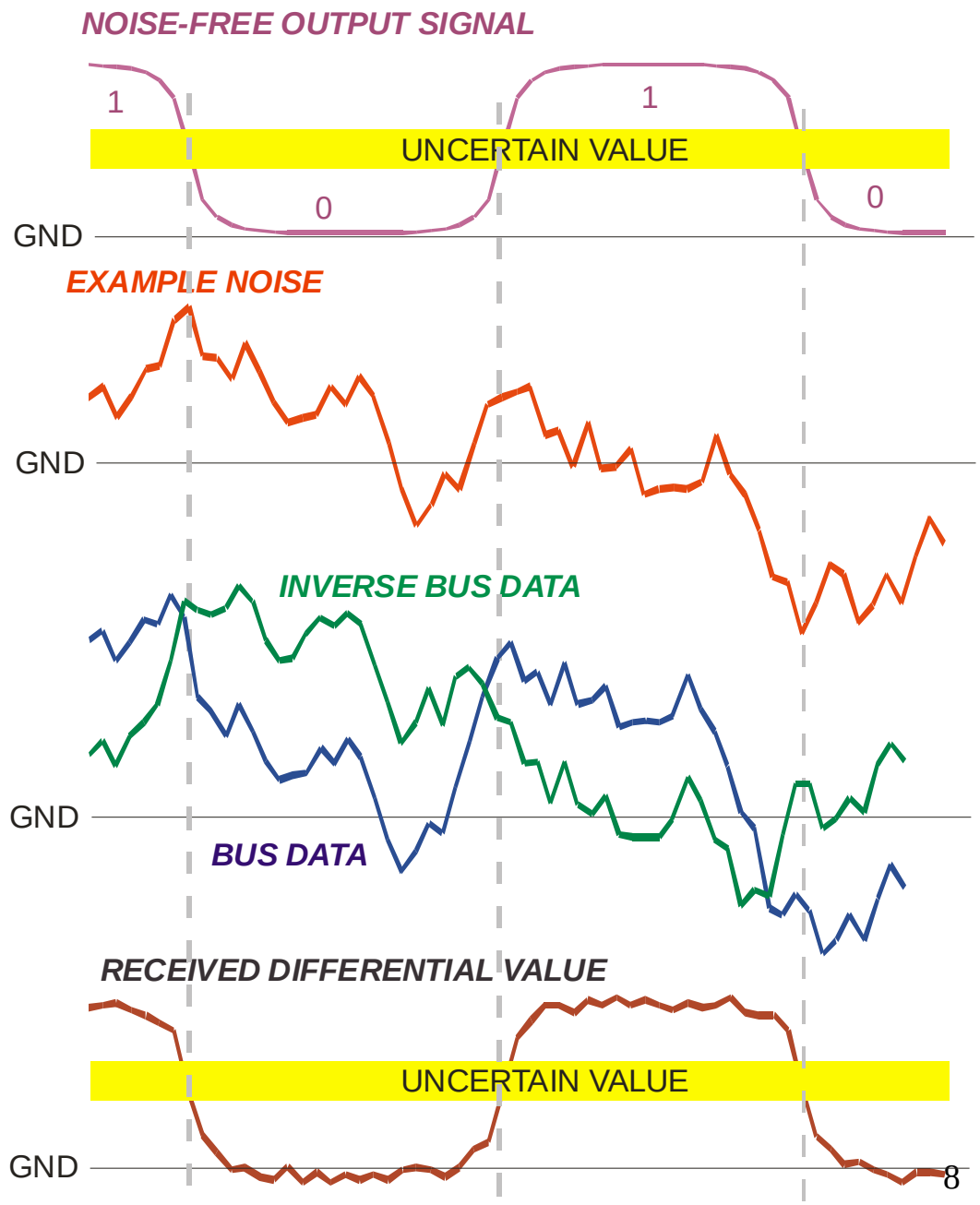
Differential Drivers To Suppress Noise

- **Send both Data and Inverse Data values on a 2-wire bus**
  - Example:

    DATA HI = 5 volts
    LO = 0 volts

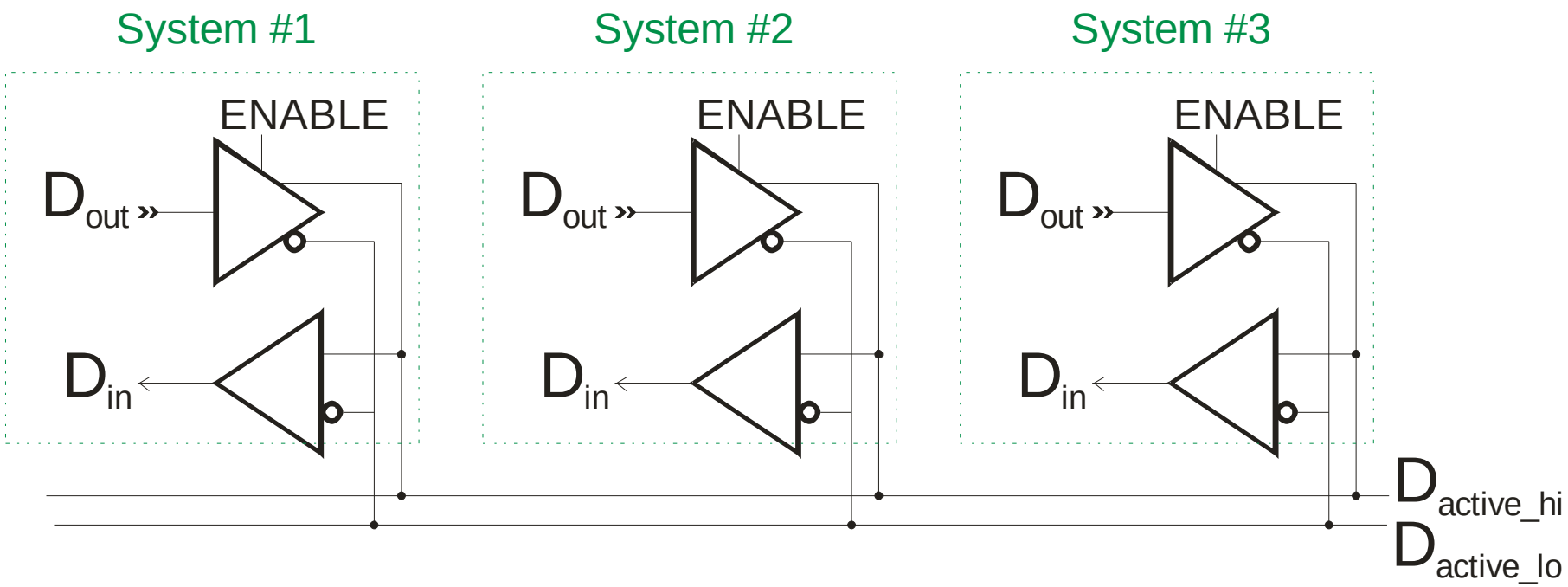
    Inverse DATA
    HI = 0 volts
    LO = 5 volts

  - Receiver subtracts two voltages
    - Eliminates common mode voltage bias
    - Leaves any noise that affects lines differently



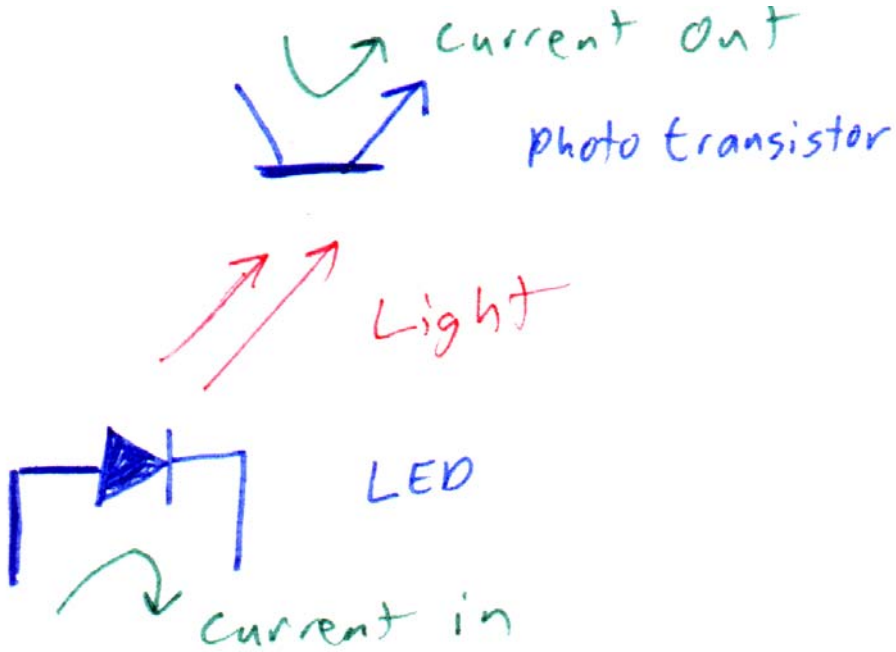
RS-485 Is A Common Multi-Master Bus

- **Used in industrial control networks (e.g., Modbus; Profibus)**
  - RS-422 differential drivers; high speed + good range (10 Mb/s @ 12 meters)
  - Add terminators to reduce noise
  - Make sure that exactly one system has its output enabled at a time!
    - Often it is "master/slave" – one system tells each other system when its turn comes



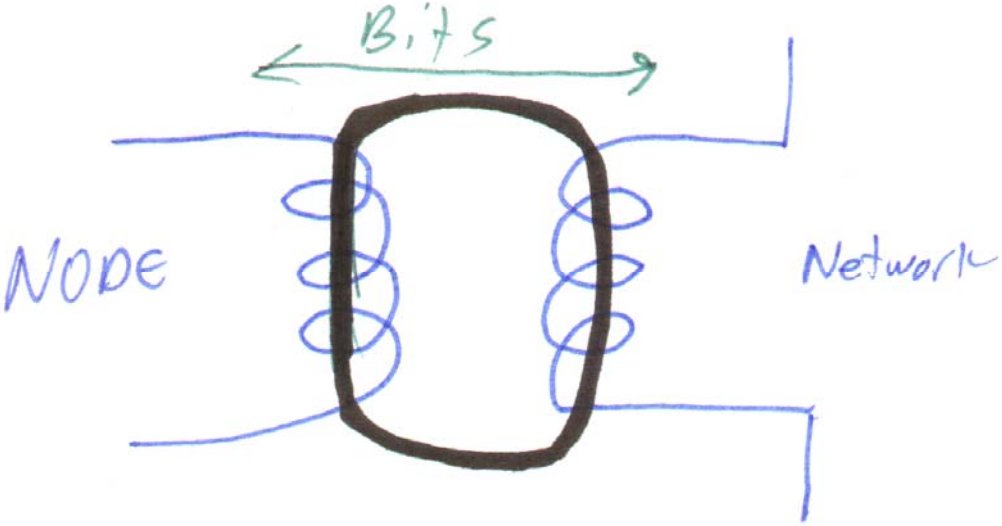
Optical Isolators For Voltage Spikes

- **Big noise spikes can cause damage to connected nodes**
  - Want isolation to help with very sharp, big spikes
- **Optical isolators provide a physical "air gap"**
  - LED illuminates when provided with current
  - Photo-transistor conducts when LED shines IR light on it
  - Two sets for each node – one set for transmit; a second set for receive
- **Provides excellent isolation**
  - No physical connection – just photons crossing a gap
  - LED saturates, preventing over-drive
  - Still subject to noise
  - Network must have its own power supply for receive LEDs
- **Supports bit dominance**
  - If LED sticks "on" network is disrupted



What About Voltage Spikes & Stuck Nodes?

- **"Stuck" nodes are a problem**
  - If a node sticks at transmitting a "low" or "high", can disable entire network
- **One common solution: current-mode transformer coupling**
  - AC component of bit edges crosses transformer
  - DC component of stuck nodes is ignored
  - Transformer's inductance protects against spikes
  - Current mode operation improves noise rejection
  - Commonly used in flight controls
- **BUT, limitations**
  - Can't do bit dominance
  - Collision detection very difficult
  - Transformer "droop" requires frequent data edges
  - Signals must be DC balanced (equal "hi" and "lo" energy)



Encoding Styles

- **RZ – Return to Zero encoding**
  - Encoding ensures that signal returns to “zero” every so often
  - Forces edges every bit or two by simple encoding rules
    - Makes it easy to synchronize receivers to bit stream
    - Makes it easy to use transformer coupling

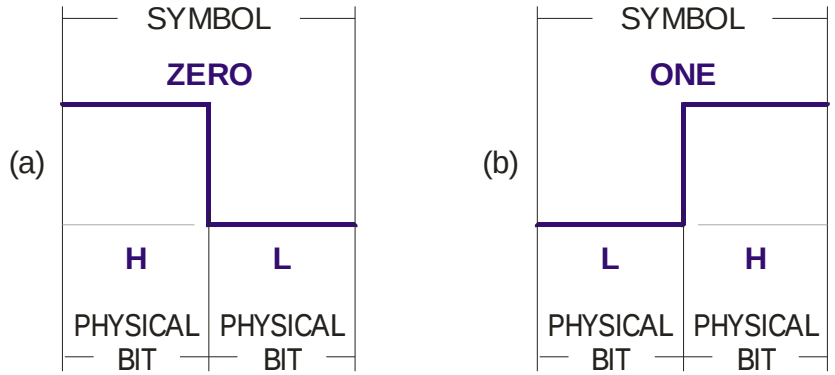
- **NRZ – Non-Return to Zero encoding**
  - Attempts to improve efficiency by just sending bit values without guaranteed edges
  - But, lack of edges makes it difficult to synchronize receivers
    - We’ll discuss ways around that problem
    - And makes use with transformer coupling difficult

- **Notes:**
  - Both encodings are subject to bit flips, even with differential transmitters
  - We’re using “physical bits” to represent HI/LO values
    - Symbols (“data bits”) might take one or more physical bits, depending on encoding

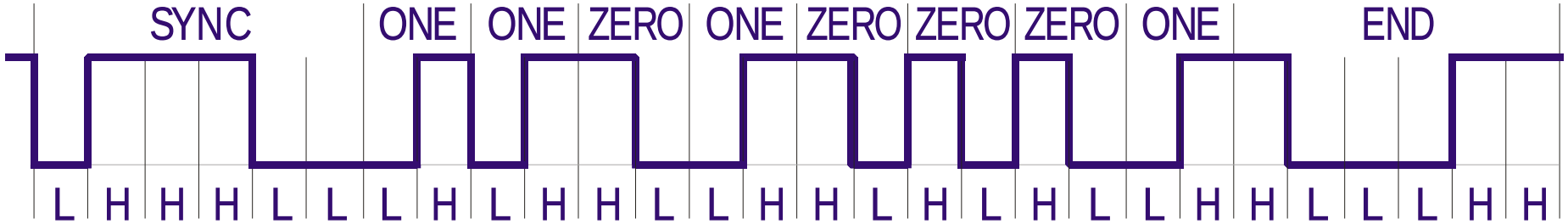
Basic Bit Encoding - Manchester

- **Manchester Encoding**
  - Data encoded by transition from high-to-low or low-to-high
  - Guaranteed transition in every bit – but worst case bandwidth is 2 edges per bit
  - Errors require inverting adjacent pairs of physical bits

Manchester Bit Encoding



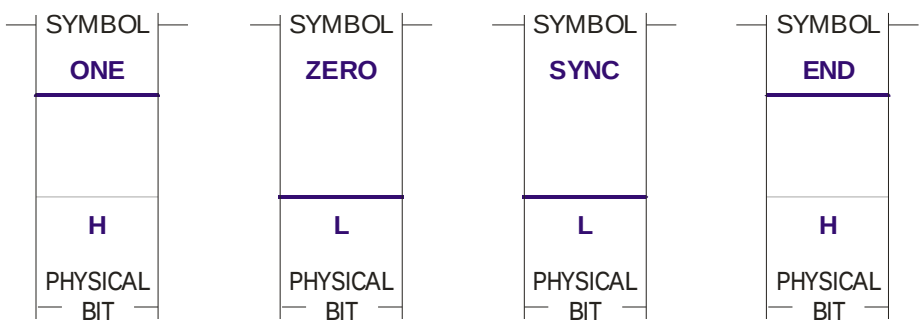
Manchester Encoding Example: 1101 0001



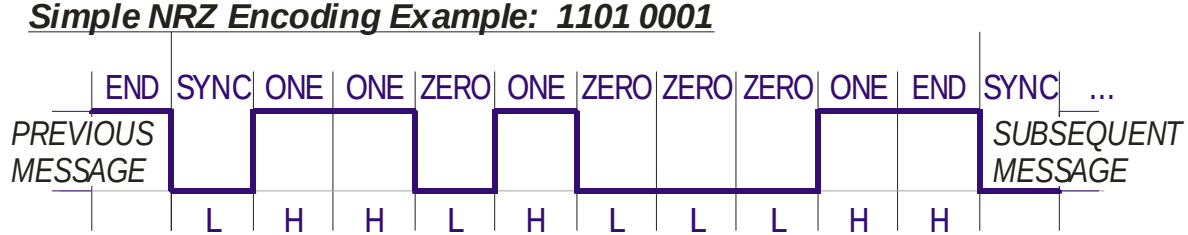
Non-Return to Zero (NRZ) Encoding (see 18-348)

- **Send a Zero as LO; send One as HI**
  - Worst case can have all zero or all one in a message – no edges in data
  - Simplest solution is to limit data length to perhaps 8 bits
    - SYNC and END are opposite values, guaranteeing two edges per message
    - This is the technique commonly used on computer serial ports / UARTs
  - Bandwidth is one edge per bit
    - But no guarantee of frequent edges

Simple NRZ Bit Encoding



Simple NRZ Encoding Example: 1101 0001



Generic Message

SYNC	HEADER	DATA	ERROR DETECTION	END

- **Sync**
  - Designates start of a message and lets receiver sync to incoming bits
- **Header**
  - Global priority information (which message gets on bus first?)
  - Routing information (source, destination)
- **Data**
  - Application- or high-level-standard defined data fields (often only 1-8 bytes)
- **Error detection**
  - Detects corrupted data (e.g., using a CRC)
- **End**
  - Designates end of message

Central Issue: Message Priority

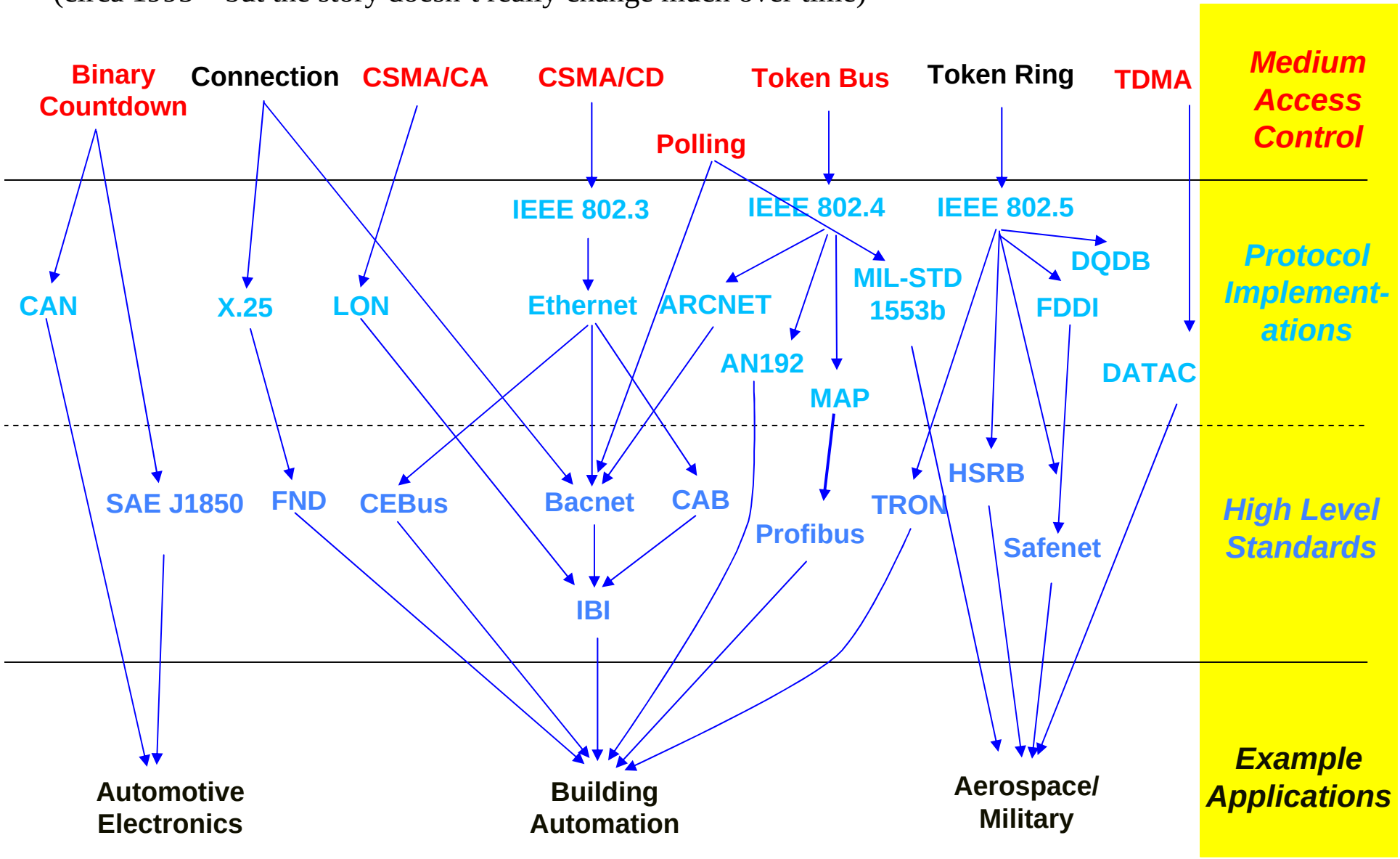
- **Local priority**
  - Each node transmits its highest priority message *when it gets a turn on the bus*
  - Or, it can implement some form of round-robin message transmission, etc.

- **Global priority**
  - Which node gets the next turn on the bus?
  - Could be a function of round-robin selection of nodes
  - Could be a function of the node's inherent priority
  - Could be a function of the priority of the highest message on the node -- a "global message priority" scheme

- **Fundamental tension:**
  - Reducing latency for high-priority nodes/messages
    vs.
  - Ensuring fairness/no starvation for low-priority nodes/messages

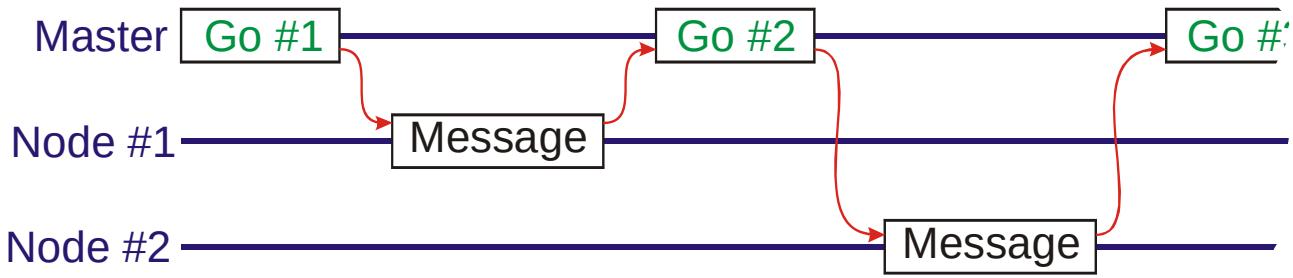
Embedded Protocol Family Tree

(circa 1995 – but the story doesn't really change much over time)

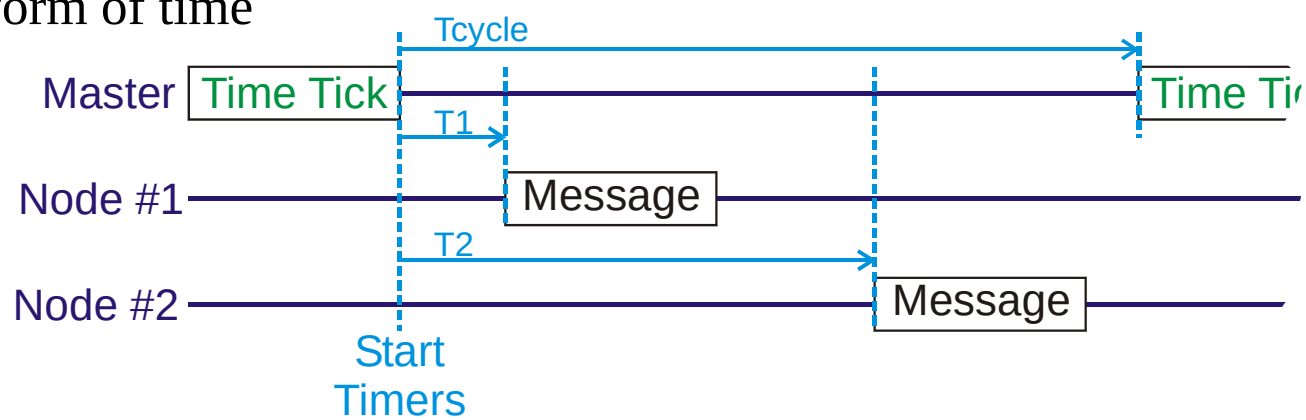


Coordination: Bus Master Approach

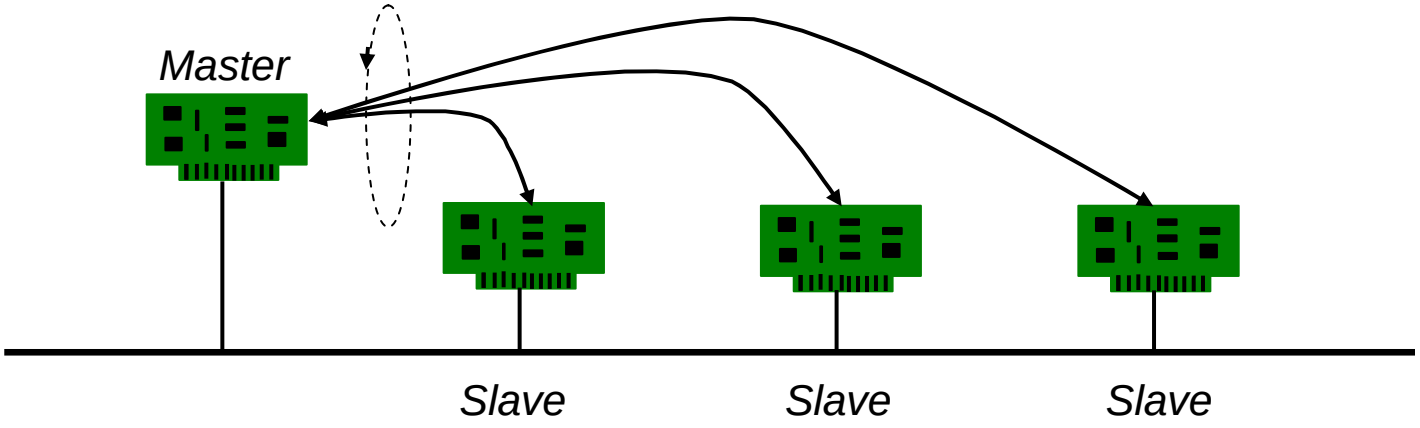
- **Bus Master can <u>poll</u> for messages & wait for response**
  - Problem: missing/slow slave
    - Master uses worst-case timeout waiting for response
    - If slave gets confused/is late, protocol fails
  - Problem: broken master



- **Master can send a time tick – <u>TDMA</u>**
  - Other nodes select response time from that time tick
  - Then becomes a form of time slice/time slot protocols (discussed later)



Polling



- **Operation**
  - Centrally assigned Master polls the other nodes (slaves)
  - Non-master nodes transmit messages when they are polled
  - Inter-slave communication through the master

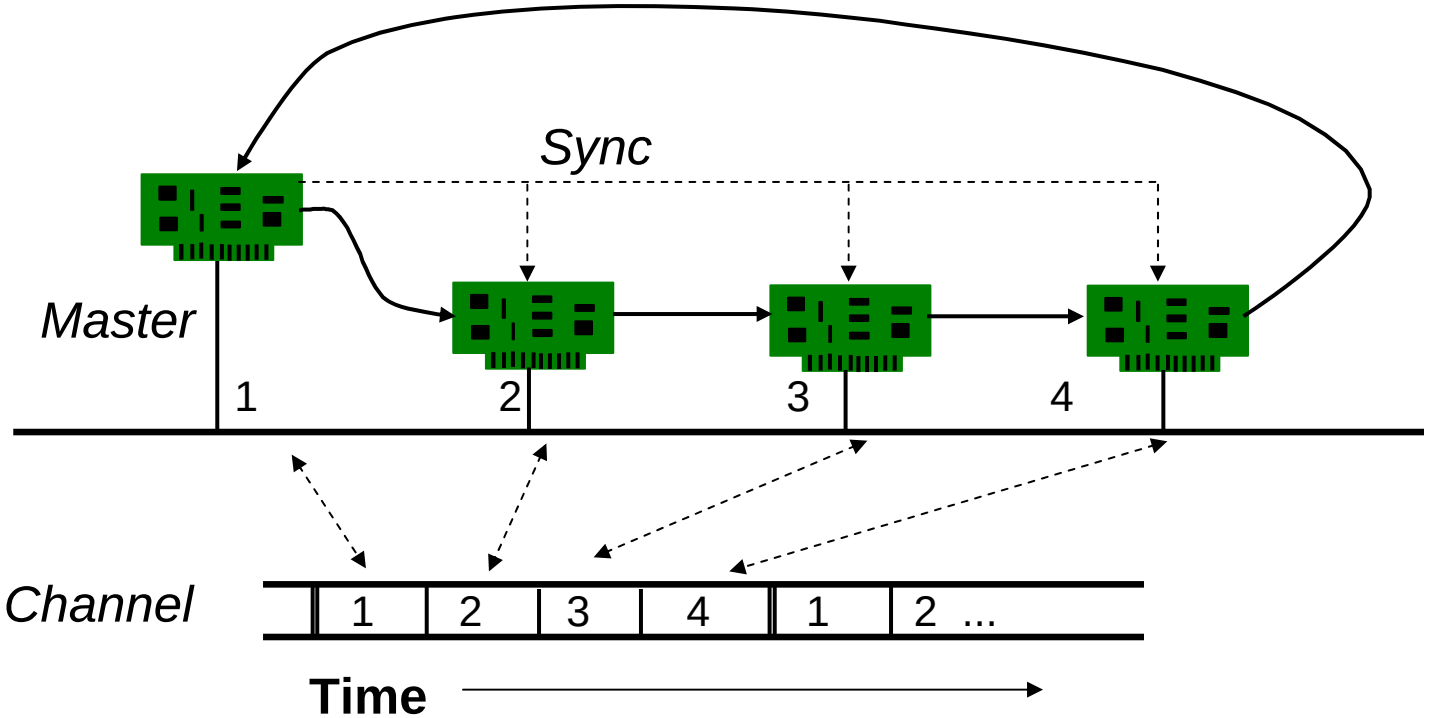
- **Examples**
  - MIL-STD-1553B, 1773, Profibus, Bacnet, AN192

Polling Tradeoffs

- **Advantages**
  - Simple protocol to implement;
    historically very popular
  - Bounded latency for real-time applications

- **Disadvantages**
  - Single point of failure from centralized master
  - Polling consumes bandwidth
  - Network size fixed during installation (not robust)
    - Or, master must discover nodes during reconfiguration
  - Prioritization is local to each node
    - But, can use centralized load balancing
    - Polling need not be in strict order; it could be, for example:
      1, 2, 1, 3, 4, 1, 5, 1, 3, 1, 6, …(repeats)

TDMA - Time Division Multiplexed Access



- **Operation**
  - Master node sends out a frame sync to synchronize clocks
  - Each node transmits during its unique time slot

- **Examples**
  - Satellite Networks, DATAC, TTP, static portion of FlexRay

TDMA Tradeoffs

- **Advantages**
  - Simple protocol to implement
  - Deterministic response time
  - No wasted time for Master polling messages

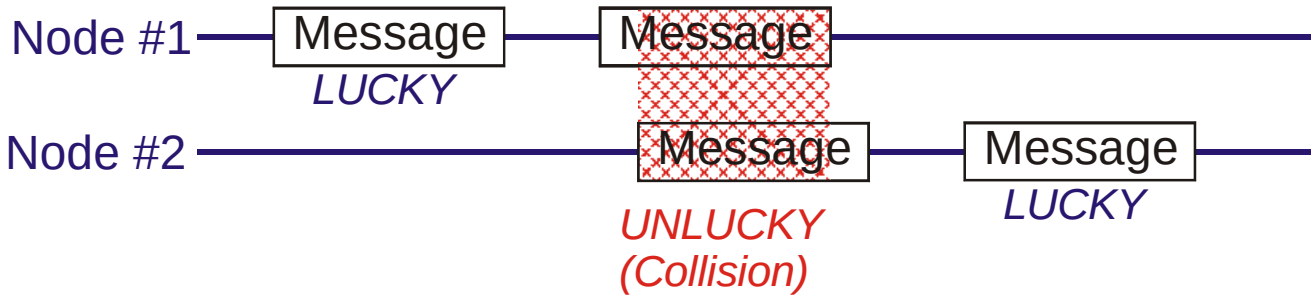
- **Disadvantages**
  - Single point of failure from the bus master
  - Wasted bandwidth when some nodes are idle
  - Requires stable clocks
  - Network size fixed during installation (not robust)
  - Prioritization is local to each node
    - (can use centralized load balancing)

- **Variation: Variable Length TDMA (~Implicit Token)**
  - Unused time slices are truncated to save time
  - More efficient use of bandwidth
  - Used in FlexRay Dynamic Segment

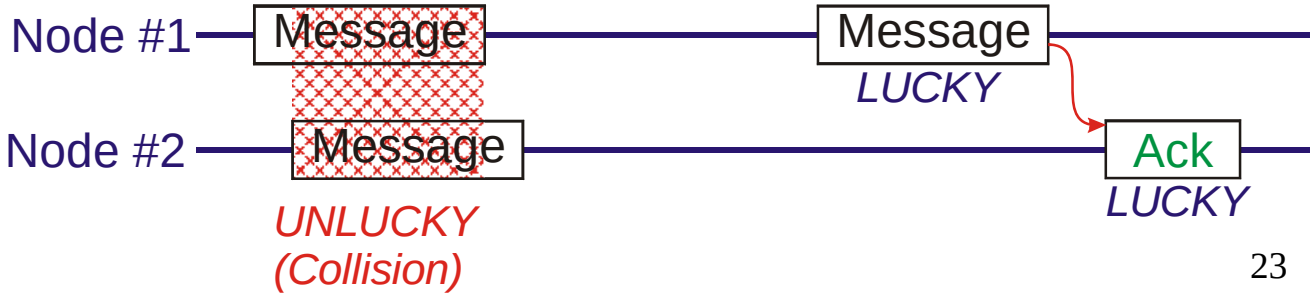
Coordination: Transmit and Hope (CSMA)

(CSMA = Carrier Sense Multiple Access)

- **Send a message and hope it made it**
  - Useful for satellites & systems with no collision detection
  - Vulnerable for entire time a message is transmitting
  - No direct way to know if message was delivered successfully



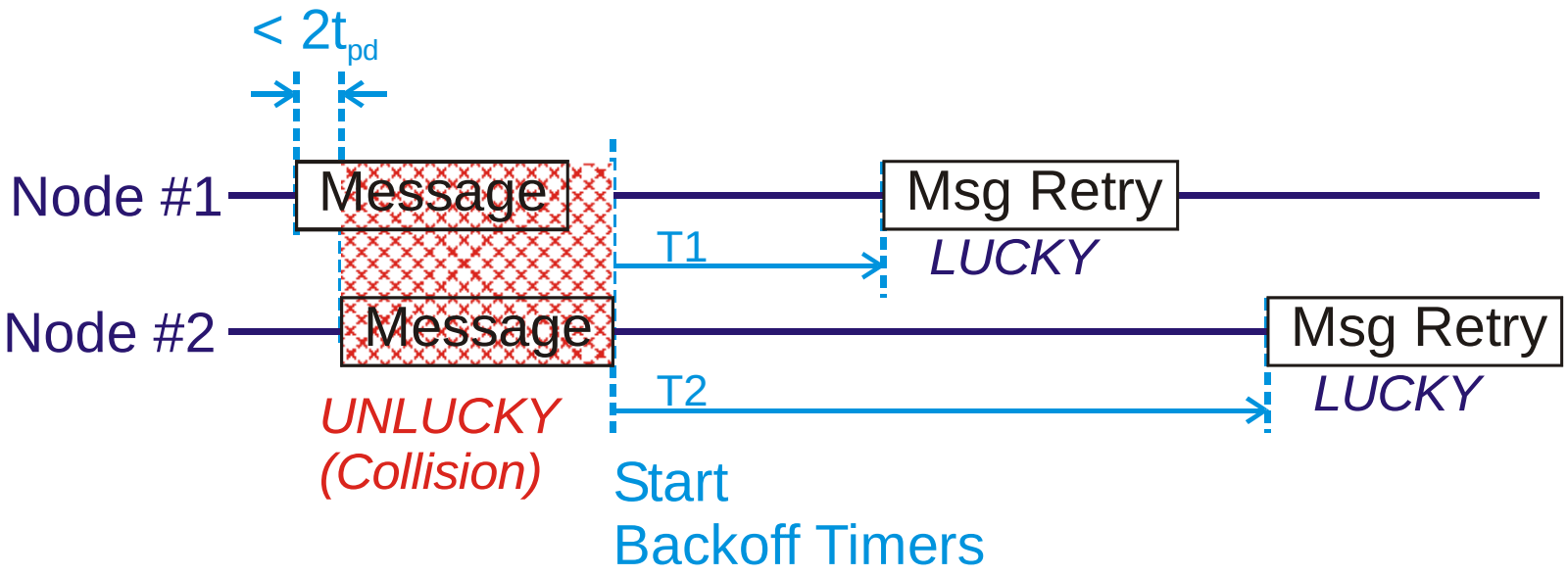
- **Send a message and wait for a response saying you made it**
  - *IMPLICIT* collision detection
  - Response might not make it even if message makes it
  - Iterate until some node pair gets lucky *twice* in a row



Transmit And Collide (CSMA/CD)

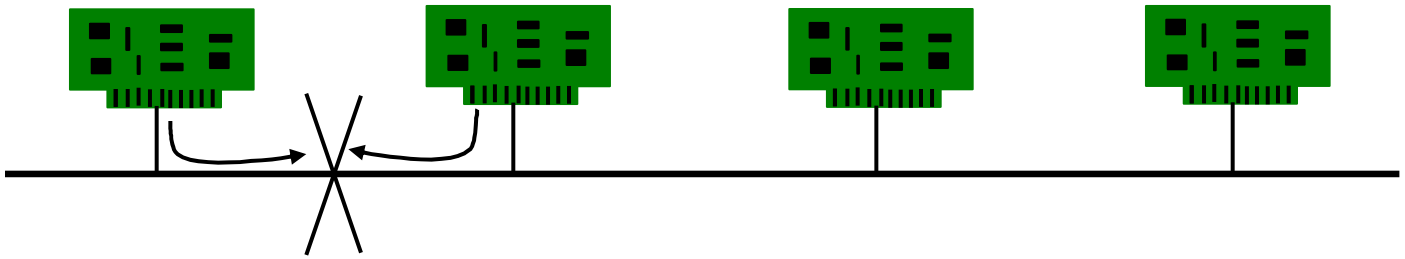
- **Transmit message; if you get lucky network transitions to "active"**
  - If you get unlucky, you get a collision event
  - Vulnerability window is about 2 t_{pd}
    - (Two propagation delays along length of network)

- **After collision, back off a certain time**
  - Amount of time to back off should vary with network load
  - Repeated collisions result in increasing backoff times



CSMA/CD

- **Carrier Sense Multiple Access / Collision Detection**



- **Operation**
  - A node waits for an idle channel before transmitting
  - Collisions can occur if two or more nodes transmit simultaneously
  - If a collision is detected, the nodes stop transmitting
    - Resolve contention using random backoff algorithm (2x longer interval each retry)

- **Examples**
  - Ethernet, IEEE 802.3, Bacnet, CAB, CEBus

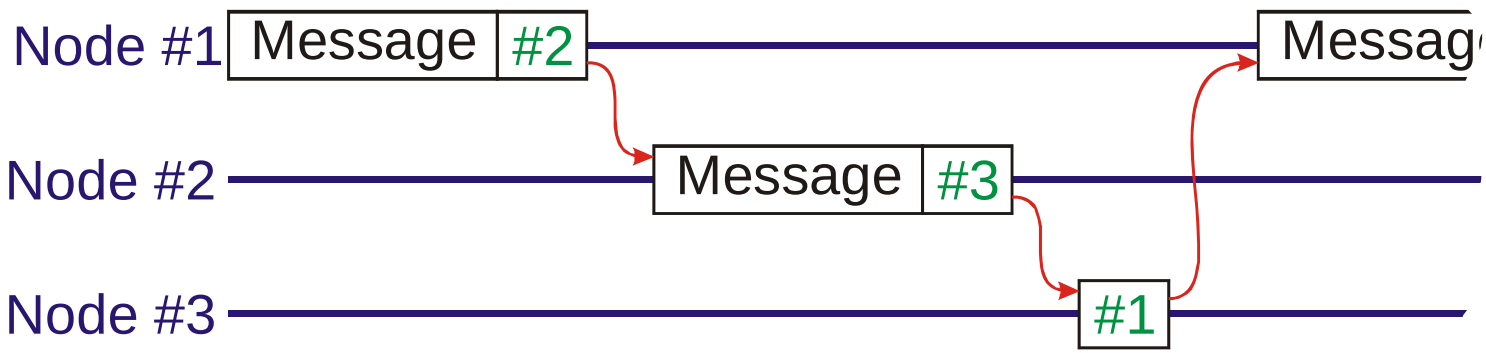
CSMA/CD Tradeoffs

- **Advantages**
  - Small latency for low traffic load
  - Network initialization/configuration is not required
  - Node can enter or leave the network without any interruption
  - Supports many nodes
  - Probabilistic global prioritization is possible
  - Extensive installed base and support

- **Disadvantages**
  - Designed for aperiodic traffic - not ideal for synchronized control loops
  - Collision detection is an analog process which is not always practical
  - Unbounded individual message latency
  - Poor efficiency under heavy loads

Coordination: Explicit Tokens

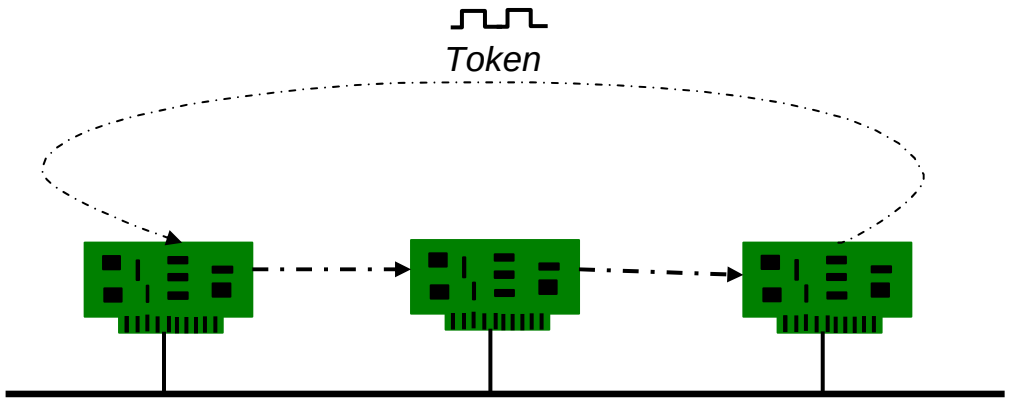
- **"Token" value says which node is transmitting and/or should transmit next**
  - Token holder = OWNER; only the owner may transmit
  - Master/slave polling is a special form where token is passed by master and returned to master by slave
  - Problems: Lost token / Duplicated token(s) / Who starts?



Token passes to next node according to # field.

- **Token passed as node number or other similar value**
  - May be tacked on to end of data-bearing message
  - Can be either node # that has token or node # that gets token next
  - Null messages with tokens must be passed to prevent network from going idle

Token Bus



- **Operation**
  - A token signal is passed from a node to node on a bus (virtual ring)
  - Only the token holder has permission to access the media

- **Examples**
  - IEEE 802.4, Arcnet, AN192, MAP, Profibus

Token Bus Tradeoffs

- **Advantages**
  - Bounded latency for real-time control applications
  - High throughput during heavy traffic
  - On-the-fly reconfiguration

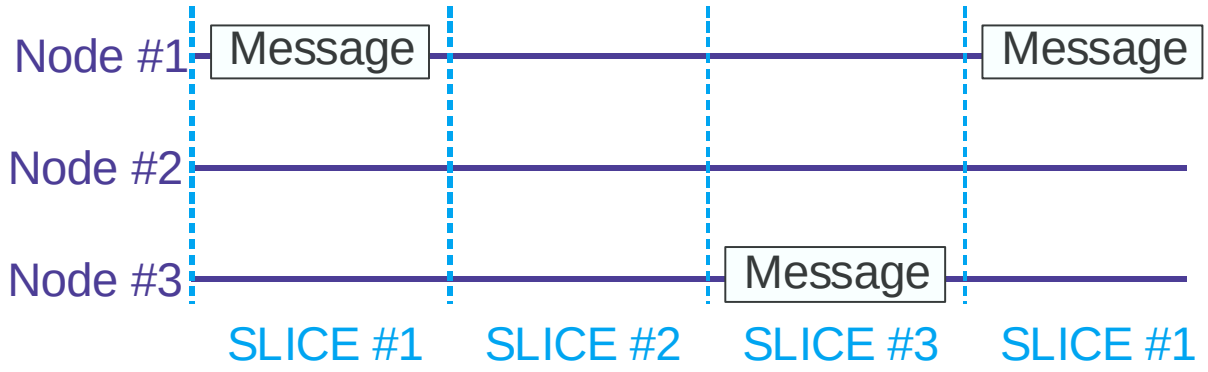
- **Disadvantages**
  - Token passing latencies under light traffic conditions
  - Prioritization local to each node
  - Lengthy reconfiguration process
  - Token initialization, loss, and duplication recovery overhead
  - Collisions may occur during initialization and reconfiguration
  - Complex protocol (especially at MAC sublayer)

- **Token bus was popular for a while, but is used less often now**

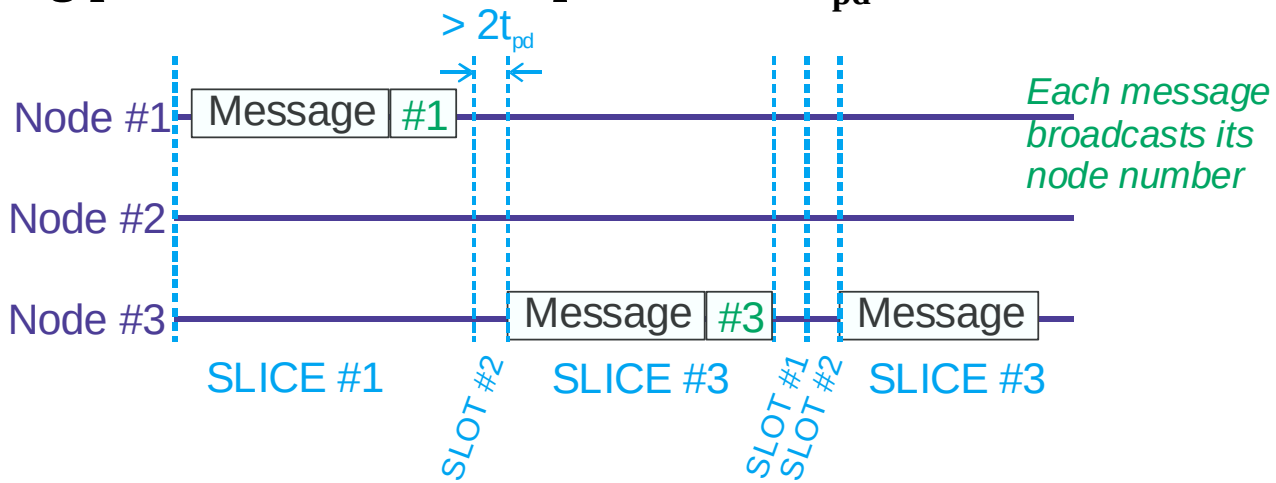
Coordination: Implicit Tokens

- **Length of waiting period is used as a time-domain implicit "token"**
  - Owner of bus determined by what time it is instead of explicit token message

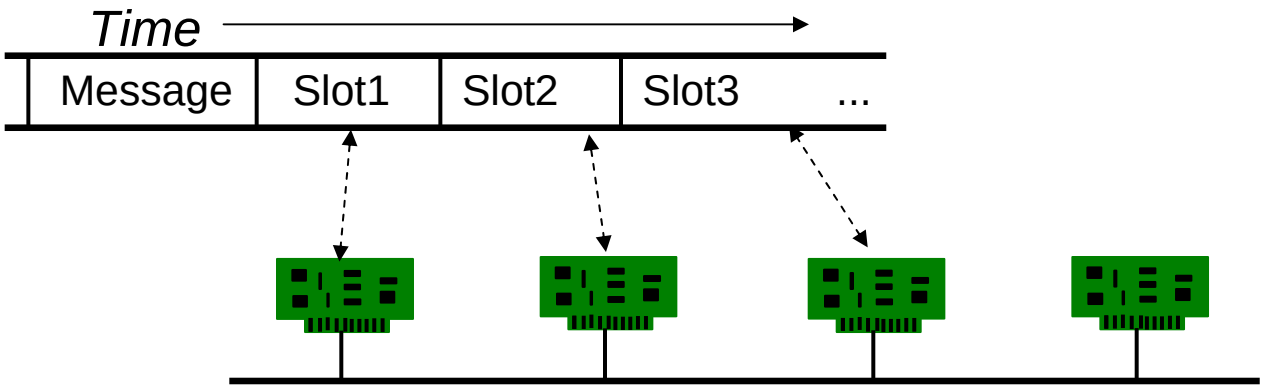
- **Time *slices* -- waiting period is a whole message long**
  - TDMA, TTP



- **Time *slots* -- waiting period is as short as possible ~ $2t_{pd}$**
  - CSMA/CA



CSMA/CA (Implicit Token)



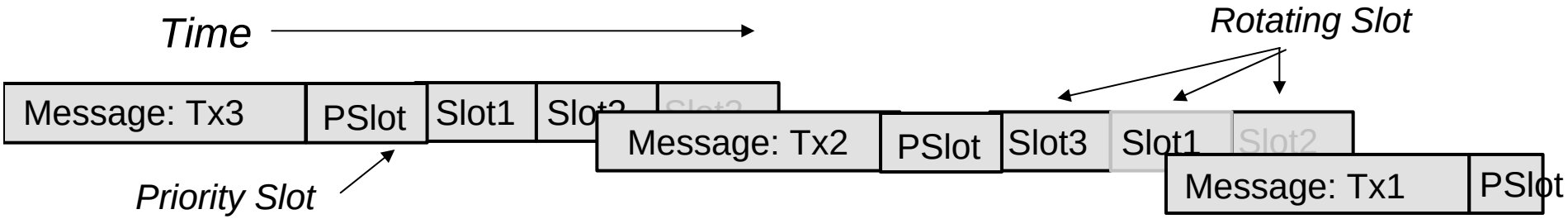
- **Operation**
  - IDLE: Active station transmits immediately
  - After each message, reserve S slots for N nodes

    IMPORTANT: Slots are normally idle – ***they are time intervals, not signals!***

  - BUSY: Transmit during your assigned slot
    - If S=N, no collisions - known as **Reservation CSMA**
    - If S<N, statistical collision avoidance

- **Example**
  - Echelon LONTalk

CSMA/CA Slot Strategies



- **One or more Priority slots (Pslots)**
  - Always in the same order after the message
  - Used for global prioritization – high priority messages
  - Each slot belongs to exactly one transmitter with a priority message
  - Could be multiple: Pslot0, Pslot1, Pslot2 assigned per message type

- **Multiple Rotating slots**
  - Rotating order based on last message sender – enables fairness
  - Generally one per transmitter, shared among all non-priority messages

- **Each slot is a few bit times – long enough for signal propagation**
  - Slots are time intervals and ***NOT SIGNALS***
    - ***Slot is "no signal" unless a message starts transmitting in it***
  - *When transmitter has a message to send, it starts during its slot time*

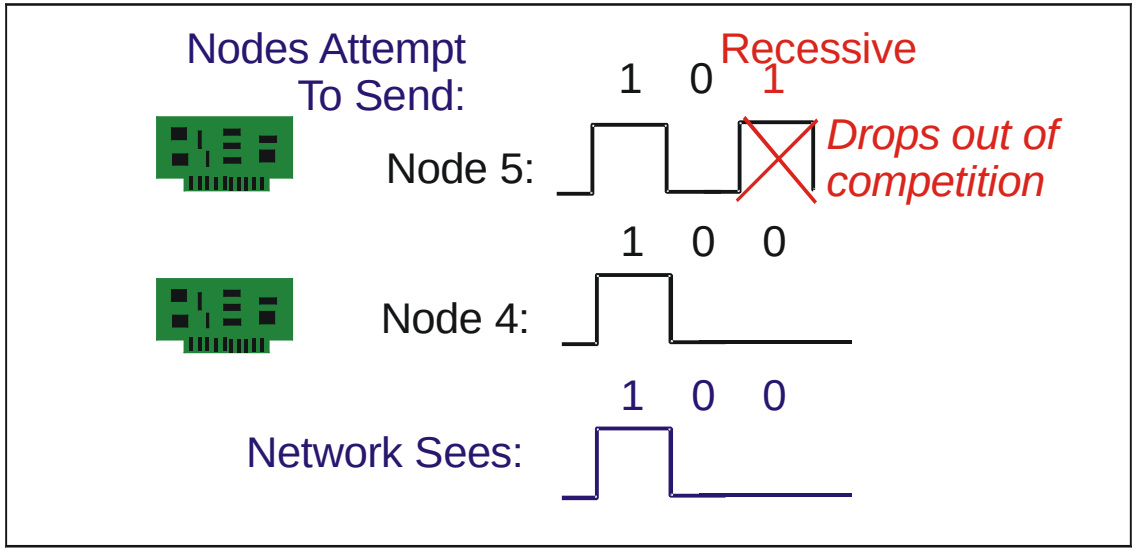
CSMA/CA Tradeoffs

- **Advantages**
  - Small latency for light traffic
  - Good throughput under heavy traffic
  - Global prioritization through fixed slots – prioritized implicit token passes
  - Bounded latency through rotating slots – non-prioritized implicit token passes

- **Disadvantages**
  - Restarting time slots from an idle bus can be difficult
    - Send dummy messages to avoid idle state
  - Collisions can occur
  - Node complexity in mapping Sth slot to Nth node

- **You'll see more of this in the FlexRay lecture**

Binary Countdown (Bit Dominance)



- **Operation**
  - Each node is assigned a unique identification number
  - All nodes wishing to transmit compete for the channel by transmitting a binary signal based on their identification value
  - A node drops out the competition if it detects a dominant state while transmitting a passive state
  - Thus, the node with the *LOWEST* identification value wins

- **Examples**
  - CAN, SAE J1850

Binary Countdown Tradeoffs

- **Advantages**
  - High throughput under light loads
  - Local and global prioritization possible
  - Arbitration is part of the message - low overhead

- **Disadvantages**
  - Propagation delay limits bus length (2 t_{pd} bit length)
  - Unfair access - node with a high priority can "hog" the network
  - Poor latency for low priority nodes

- **You'll see more on binary countdown in the CAN lecture**
  - Don't worry about exactly how this works until that lecture

EMULATION

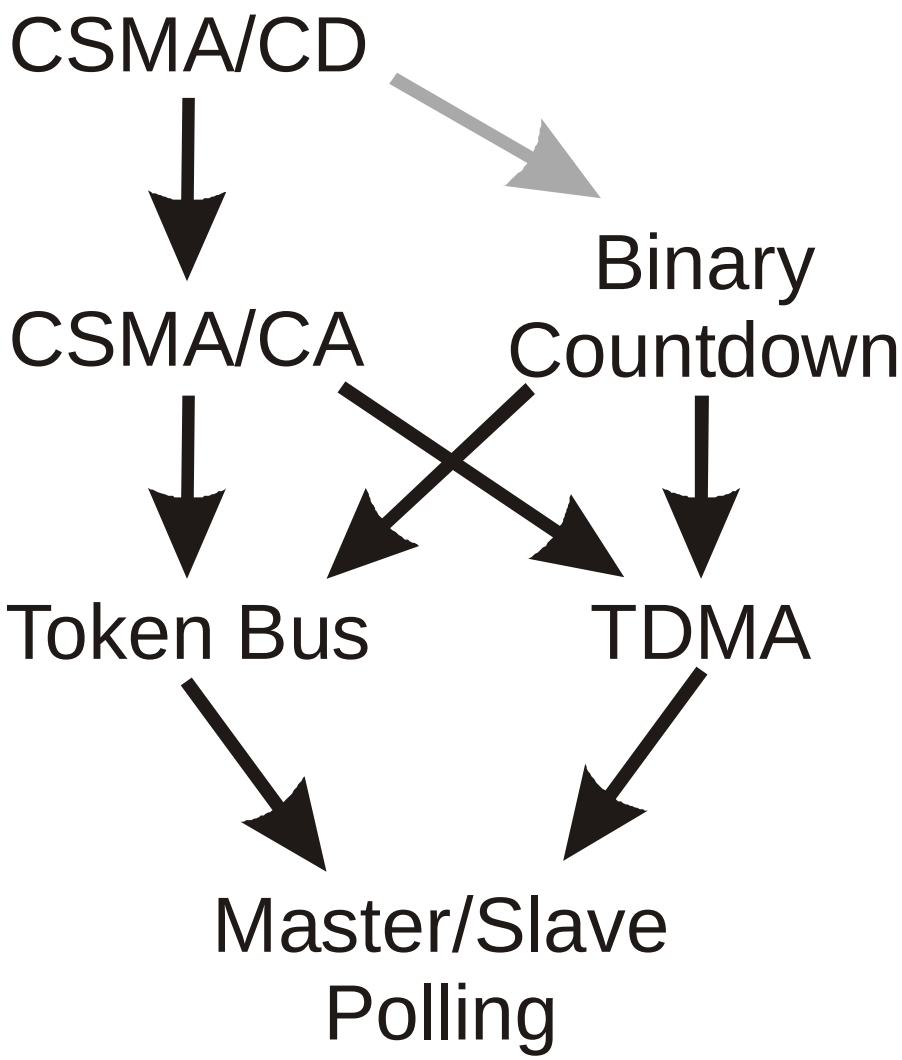
- **You can use one protocol to emulate another**

- **Use Ethernet (CSMA/CD) to emulate:**
  - Master/slave polling – slaves only respond when polled
  - Token bus – use explicit token messages; application only transmits when it has the token
  - TDMA – slaves measure time from message from master and transmit appropriately

- **But, there is no free lunch**
  - “Slot” time involves round-trip through OS – longer than a couple bit times
  - “Slice” time must account for CPU/OS jitter, not just HW oscillator drift

Emulation Capability Lattice

- **Protocols higher in picture can emulate protocols lower in picture**
  - Example: you can pass a token around on a CAN network in software



Wireless Networks

- **Strength is installation flexibility**
  - No wiring harnesses to install (except for power)
  - Can make/break networks without physical connections
  - Can have overlapping/interacting/hierarchical networks (e.g., Bluetooth)

- **Weakness is potential unreliability for critical operations**
  - Geometry may introduce standing waves/fading
  - Conflicts with other wireless systems (EMC = ElectroMagnetic Compatibility)
  - Interference from RF emitters (EMI = ElectroMagnetic Interference)
  - Limited spectrum space
  - Where does a wireless node get its power – who changes the batteries?
  - ***In general, unsuitable for use in critical applications that aren't fail-safe!***

- **Also, cost**
  - Bluetooth is getting cheap enough to be in consumer electronics
  - But has to be able to beat a piece of copper and a plastic connector
  - And that cost has to include power supply strategy

Key Overall Tradeoff Issues

- **Protocols are optimized for different operating scenarios**
  - Collision-based
    - High number of possible transmitters
    - Low number of *active* transmitters
    - Arbitration overhead proportional to activity
    - In worst case (every node active) network can effectively crash
  - Token-based, Time-multiplexed & Polled
    - Moderate number of *total* transmitters
    - Handles worst case activity without a problem
    - Arbitration overhead relatively constant
  - Binary countdown
    - Moderately large number of message types
    - Arbitration overhead constant
    - Global prioritization (*but* no mechanism for fairness)

Review

- **General embedded network issues**
  - Dynamic tension among efficiency, latency, determinism

- **Classes of protocols**
  - Time-multiplexed (polled/time-triggered)
  - Token (implicit/explicit)
  - Binary countdown
  - You should know all protocol type names and general operating principles

- **General tradeoff overview**
  - Global vs. local priority (and, priority vs. fairness)
    - Think about it – what does each protocol do about global prioritization?
  - Efficiency vs. dynamic flexibility
    - Think about it – what does each protocol do to minimize overhead if messages aren't uniformly distributed?
  - Wired vs. wireless

Supplemental Material

Protocol Tradeoffs Revisited

- **Bit encoding**
  - Self-clocking schemes are simpler, but require more bandwidth
  - Bit-stuffed schemes require extra bits for stuffing, result in nondeterministic message lengths

- **Collision-based protocols**
  - An unbounded number of collisions results in unbounded worst-case latency
    - Idea: use collision to signal start of a reservation CSMA protocol – works well
  - In general not constrained by bit speed/network length ratio (but IS constrained by message speed/network length ratio)

- **Bit dominance/binary countdown protocols**
  - Excellent efficiency
    - But must have compatible network medium
  - Constrained by network bit speed/network length ratio

Protocol Tradeoffs Revisited – 2

- **Implicit Token / Time-based protocols**
  - Longer timed intervals potentially waste bandwidth
    - Unused slices on TDMA
  - Any timed interval requires an accurate oscillator at each node
    - Worst for TDMA
    - Relevant to CSMA/CA as well
  - Constrained by bit speed/network length ratio

- **Explicit Token-based/handshake protocols**
  - Consumes bandwidth for token passing
    - Master/Slave polling the worst – individual polling message
    - Token bus OK under heavy load if token pass combined with transmission
    - Token ring is better, but requires special topology
  - Does not require precise oscillators, especially if used with self-clocking bits
  - Not specifically constrained by bit speed/network length ratio
    - But bus topologies are inefficient if network is longer than a whole message time

Protocol Tradeoffs Revisited – 3

- **Local priority**
  - Flexible, straightforward to implement

- **Global priority – requires consensus of nodes to determine winner**
  - Bit dominance does this “for free”
  - Implicit tokens approximate this by very fast (implicit) token pass to all nodes
  - Token ring approximates this by very fast (explicit) token pass to all nodes
  - Explicit token/handshake protocols in general have a difficult time doing this

- **Global fairness – requires ability to send non-prioritized messages**
  - Bit dominance must use emulation of another protocol to do this (e.g., polling)
  - Implicit tokens do this by using rotating slots
  - Explicit tokens do this as part of token passing – no additional charge

Alternative Networks

- **Optical Fiber**
  - Excellent noise immunity
  - Very high bandwidth

  - Expensive to connect/splice
  - Expensive emitter/receiver
  - Needs separate power wiring

- **Free-space optical (*e.g.*, infrared)**
  - Potential alternative for small enclosed systems
  - No wires (except for power)
  - Good for benign confined environments (e.g., TV remotes)

  - Relatively low bandwidth
  - Transceiver costs still a bit high (but being driven by palmtop PC market)
  - Still need to get power to nodes