Applied Electronics

C4 – Basic Transfer Cycles

- Skew and synchronization
- Protocol levels
- Synchronous cycles
- Asynchronous cycles with handshake

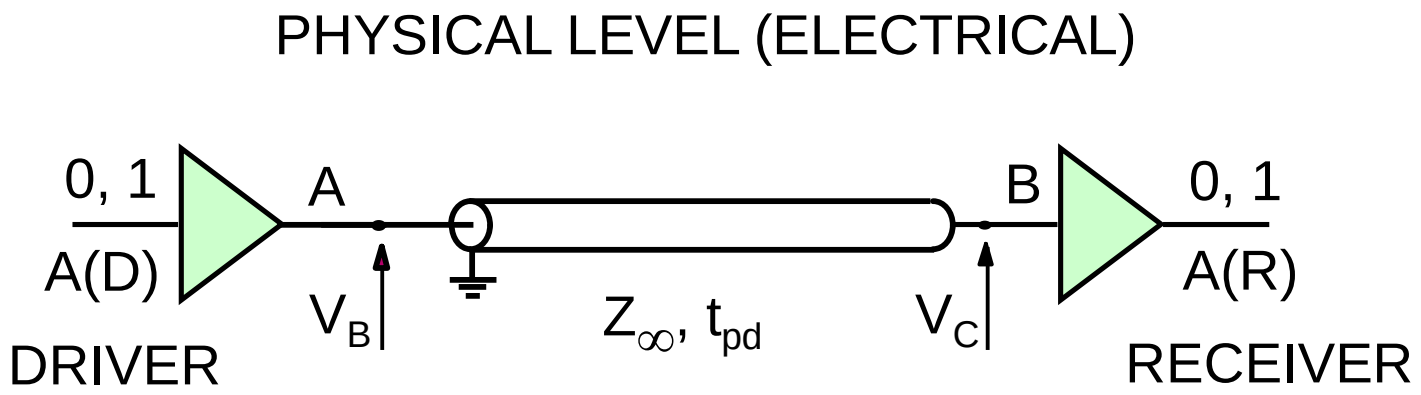
Lecture C4: Basic Transfer Cycles

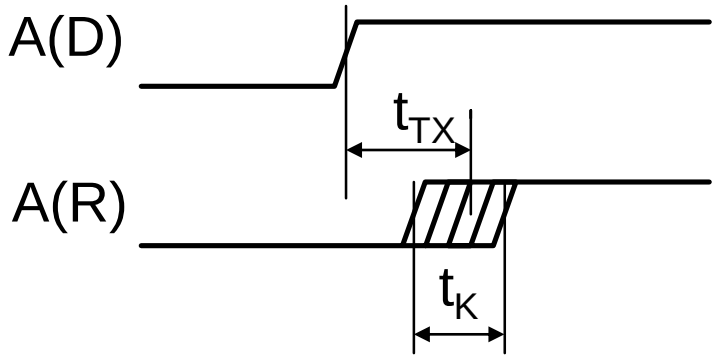
- Summary of services provided by the physical layer
- Source-destination reference model
- Basic synchronization techniques
  - Synchronous
  - Asynchronous
- Summary of the services provided by the cycle level

From physical level to cycle level

- Objective: to guarantee the correct transfer of the information
- Physical level → correct transfer of logic states (0, 1)
  - Variables: voltages, currents, impedances, ...
  - Parameters: V_O, V_I, t_{TX} , t_K , ...
  - Constraints: delays, **local** skew, noise, ...
- Cycle level → correct transfer of bit groups
  - Use the services offered by the physical layer
  - Variables: logic states (0, 1)
  - Parameters: t_{SU}, t_H, ...
  - Constraints: guarantee setup times, hold times,….

Underlying layer (physical / electrical)

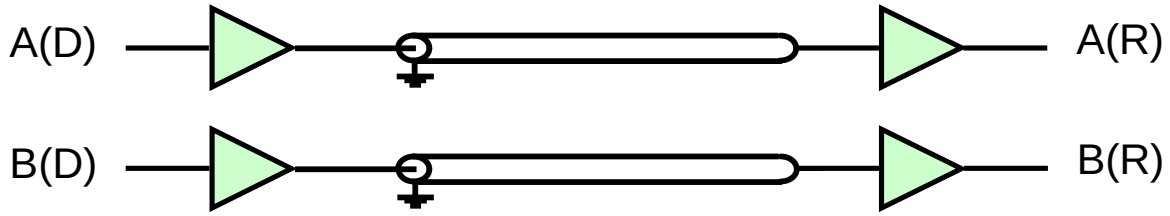


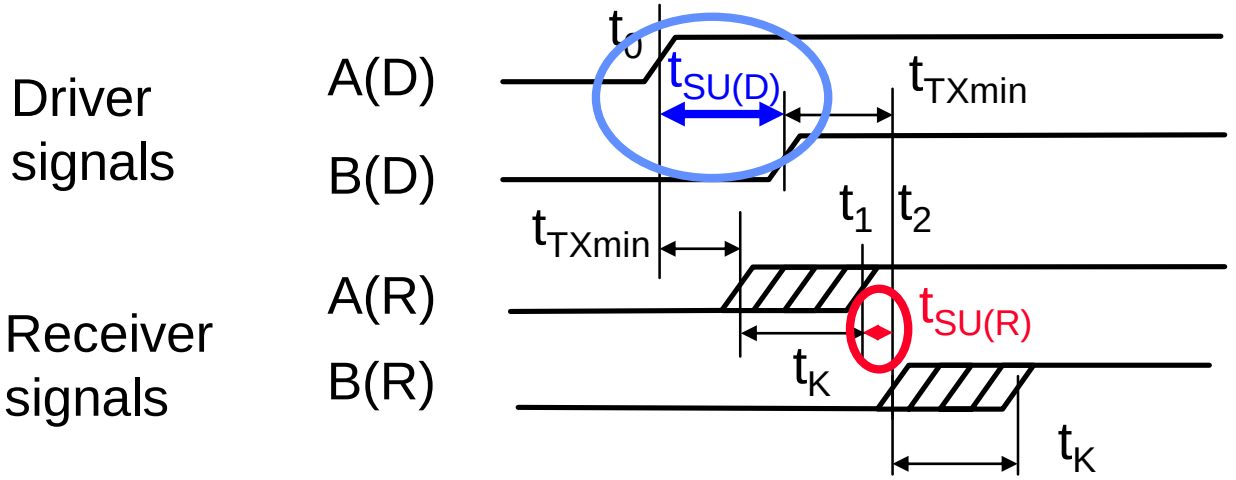


Every interconnection
- Uses energy
- Changes voltages
- Changes temporal relationships

To ensure the transfer correctness
→ Time margins

Effect of skew





$t_1 = t_0 + t_{TXmin} + t_K$; $t_2 = t_0 + t_{SU(D)} + t_{TXmin}$; $t_{SU(R)} = t_2 - t_1 = t_{SU(D)} - t_K$

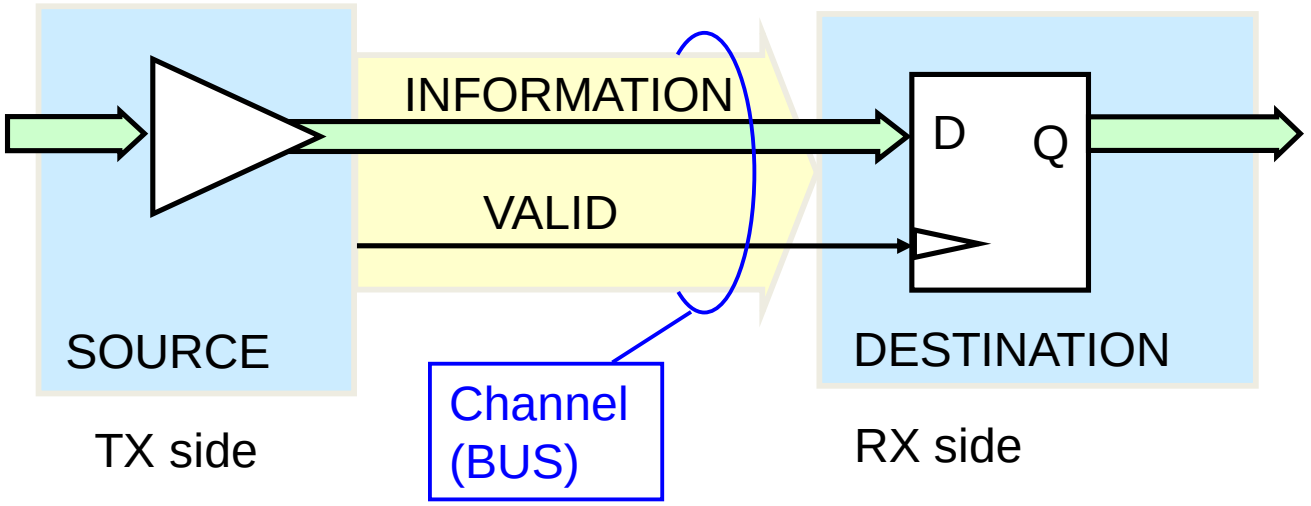
Local skew (between the various ports of a given receiver) reduces the timing margins

Cycle level protocols

- Services provided by the electrical level
  - Transfer of logic variables (bits)
    with delay (t_{TX}) and
    temporal misalignment, skew (t_K)
- Skew modifies temporal relationships
  - Different set-up and hold times on the RX and TX side
  - It can cause timing errors in registers
- The purpose of the protocols is
  - Ensure a correct transfer of information
    (allow FSMs to work properly)
  - Despite the timing variations caused by the skew

Source-Destination Model (cycle)

- Transfer in a point-to-point channel between source (TX) → destination (RX)



- Timing specifications must be met for the register to function correctly
  - Ensured by setup and hold times

Operations at the cycle level

- Two types of information transfer
  - Activated by the source → write
    (e.g., writing a register or memory cell)
    - Control and information have the same direction
      → →
  - Requested by the destination → read
    (e.g., reading a register or memory cell)
    - Control and information have opposite directions
      → ←
- The reference to describe the protocols are
  - Write operations
  - Reading operations are subsequent extensions

Objectives of the protocol: cycle level

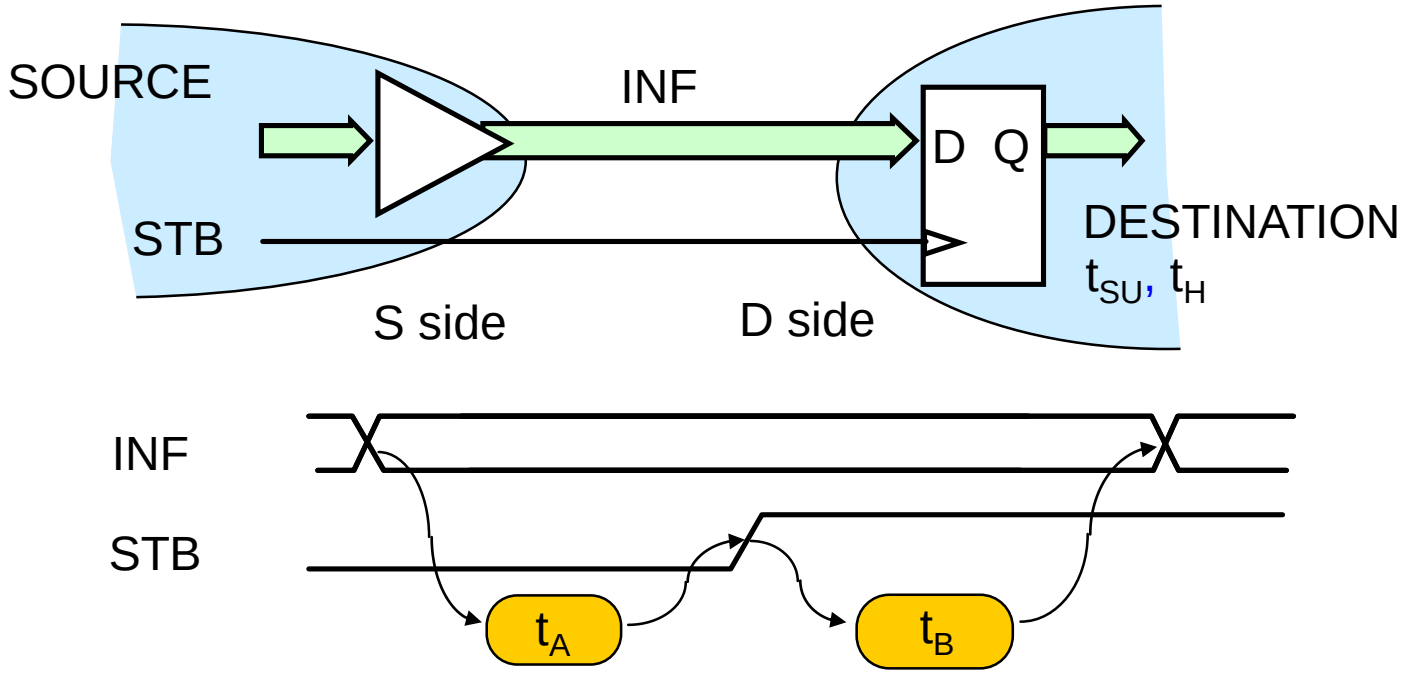
- To avoid metastability, the set-up and hold timing constraints *must be respected despite the SKEW*
- Two basic techniques
  - Fixed timing
    → Synchronous protocol
    - Must respect the most stringent specifications (worst-case) in every operation.
  - Adaptive timing
    → Asynchronous protocol
    - A module must wait for an acknowledgment from the other module involved in the transfer before continuing

Synchronization techniques

- Protocols based on cycle
  - Fixed timing → synchronous cycle
  - Adaptive timing → asynchronous cycle
- High-performance cycles
  - Synchronized by the source (source-synchronous)
  - Use both the H-L and L-H transitions (dual edge)
  - Intrinsic timing (embedded clock)
- Reference
  - Writing cycle: cycle initiated by the source

Synchronous cycle: timing

- Sequence of operations with FIXED DELAYS (t_A, t_B)

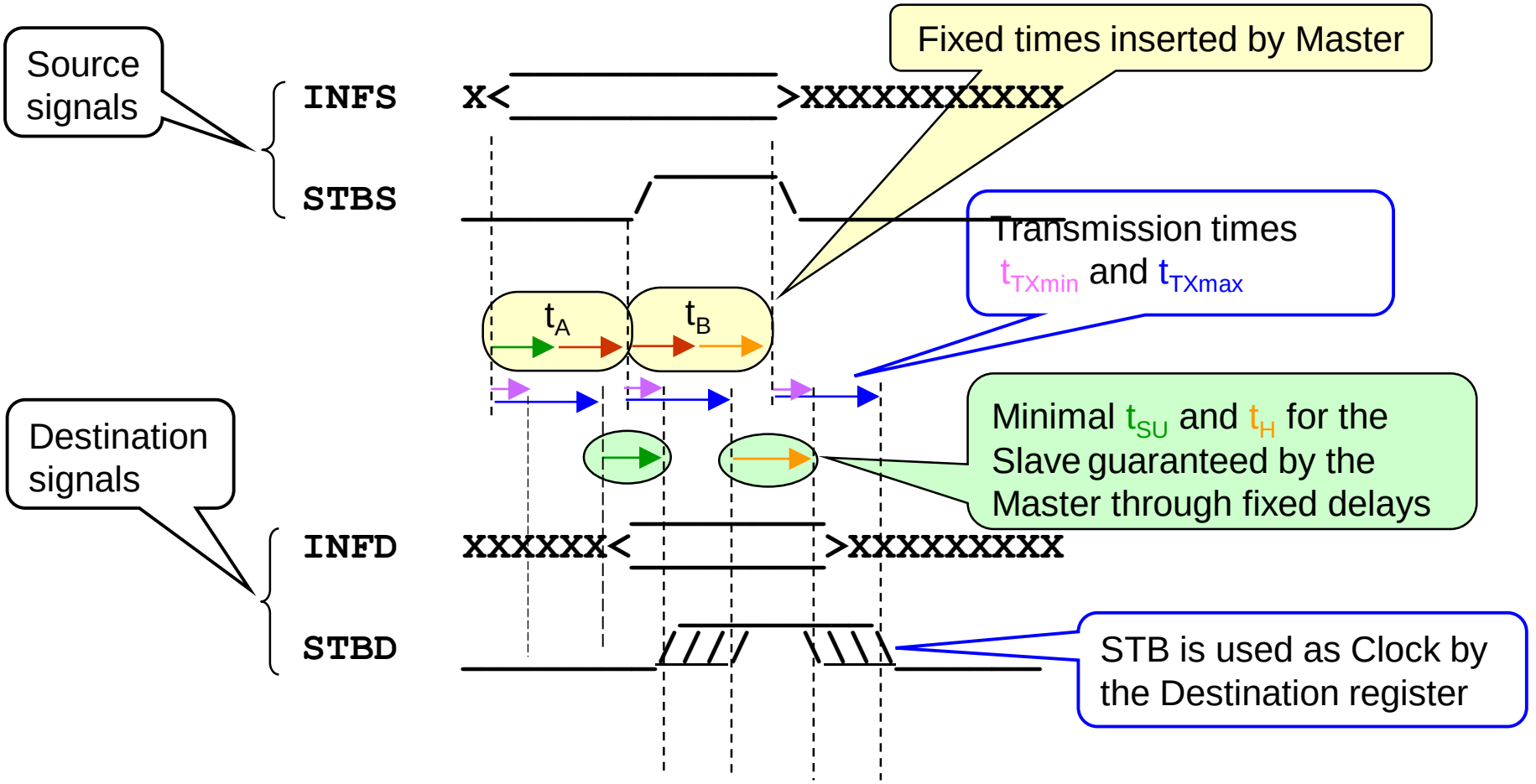


  - Real systems use different signals between Source and Destination → propagation issues!

Synchronous cycle - operations

- Operations performed in *a sequence with fixed delays* to ensure
  - The constraints of the receiver
  - Hence, the correct information transfer
- All timing parameters are controlled by the source
  - Send the information (INF)
  - Wait t_A to ensure t_{SU} $t_A > t_{SU} + t_K$
  - Send the STB command (clock for the destination register)
  - Wait t_B to ensure t_H $t_B > t_H + t_K$
  - Remove INF and STB

Example 1: synchronous write cycle



Total write time: $t_{WR} = 2\ t_k + t_{SU} + t_H$

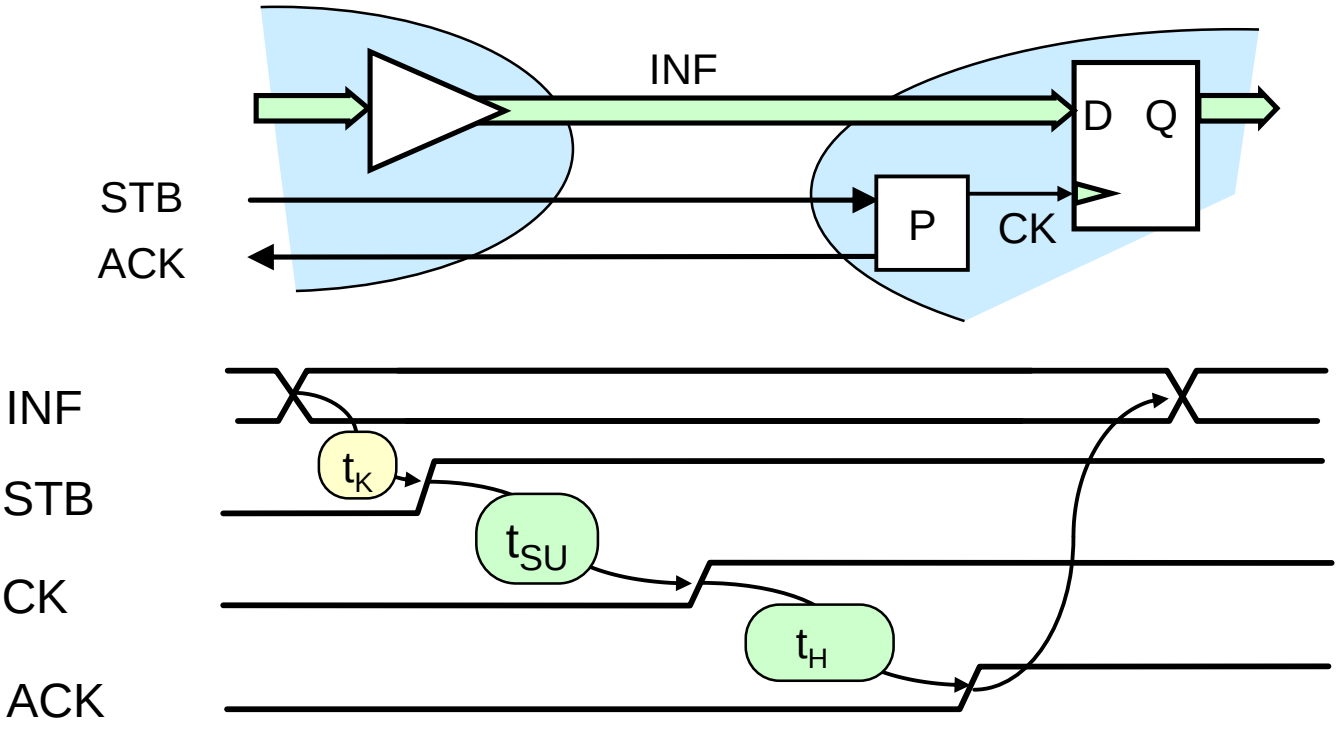
Cycle duration does not depend on t_{TXmax}

Delays in synchronous cycles

- Constraints on delays:
  - The t_A delay, source side, must guarantee the set-up time t_{SU} at the destination, *considering the skew* t_K
  - The t_B delay, source side, must guarantee the hold time t_H at the destination, *considering the skew* t_K
  - $t_A \geq t_{SU} + t_K$ $\quad\quad$ $t_B \geq t_H + t_K$
- To ensure t_A, the source unit must know t_{SU} and t_K
  - Minimum cycle time: $t_{CYS} = t_A + t_B = t_H + t_{SU} + 2\,t_K$
- For writes to multiple destinations
  - Match the *slowest destination* specifications
    - $\max(t_{SUi})$ and $\max(t_{Hi})$
  - Or adapt the speed to the target using asynchronous cycles

Asynchronous cycle: timing

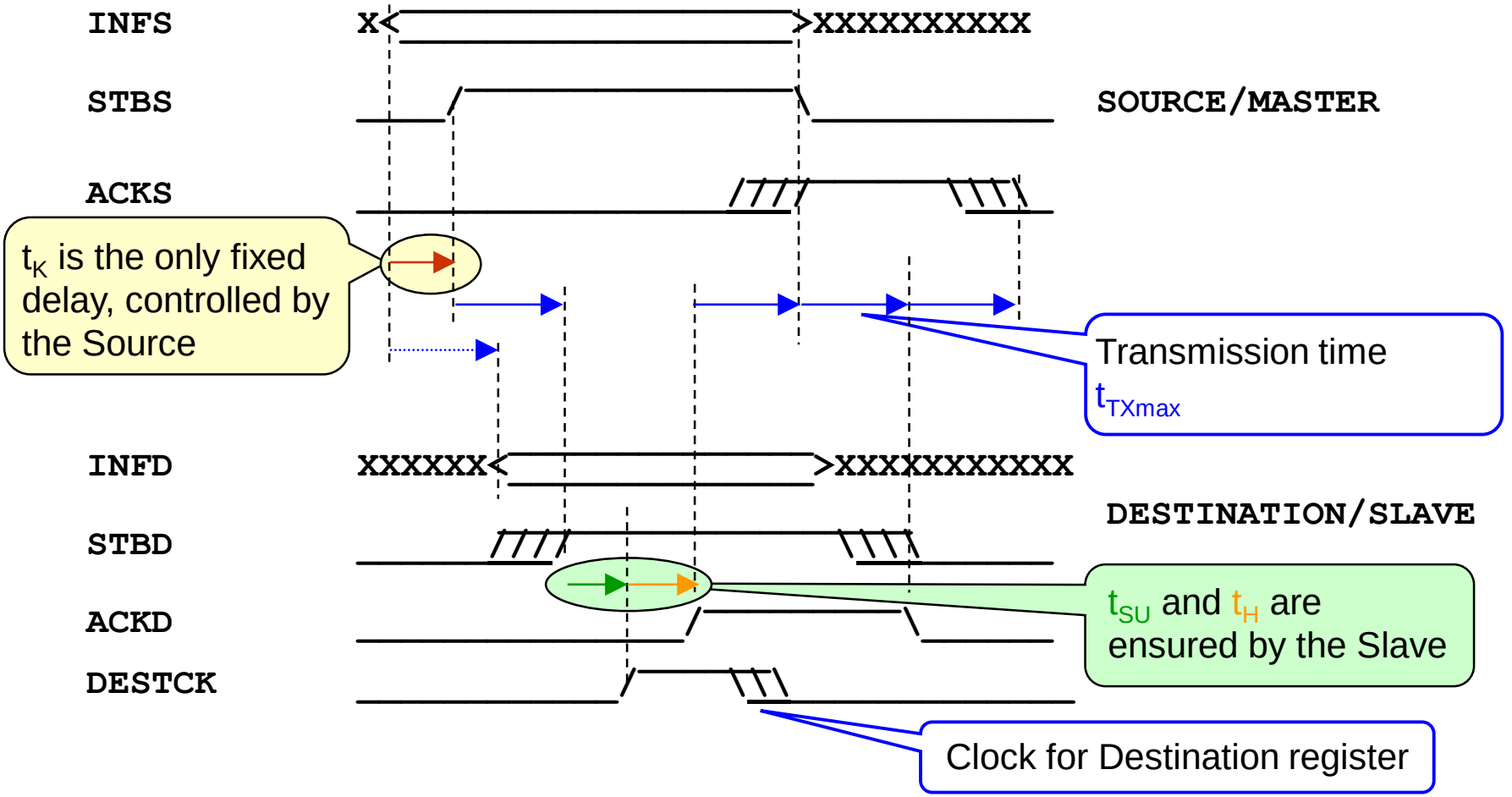
- Sequence of operations with Confirmation (ACK)
  - Timing is controlled by both modules



Asynchronous cycle: operations

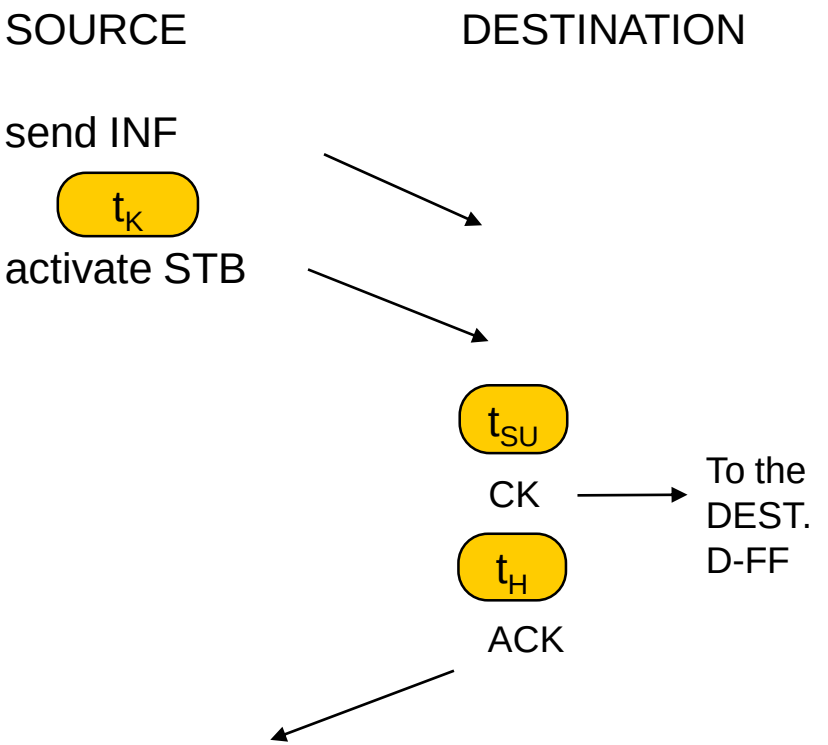
- Over the sequence of operations, timing is controlled by the Source-Destination interaction
- The Source module must know only t_K
- t_{SU} and t_H are added by the destination, within t_C
  - Send information (INF)
  - Wait t_K to ensure $t_{SU} \geq 0$ at the destination
  - Send the Strobe Command (STB)
  - Wait for Acknowledge (ACK) from the destination
  - Remove INF and STB

Example 2: asynchronous write cycle



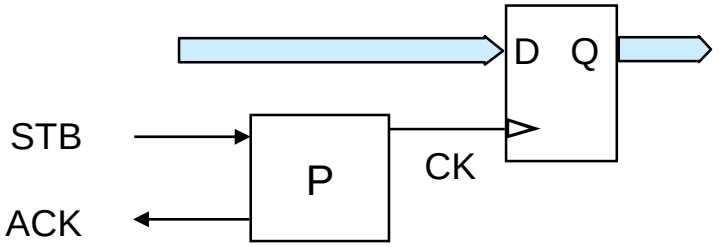
Total write time: $t_{WR} = t_K + t_{SU} + t_H + 4\ t_{TXmax}$

Asynchronous cycle: sequence



The skew t_K is inserted between sending the INF and the STB activation

Controller P inserts the setup t_{SU} and hold t_H times required by the destination



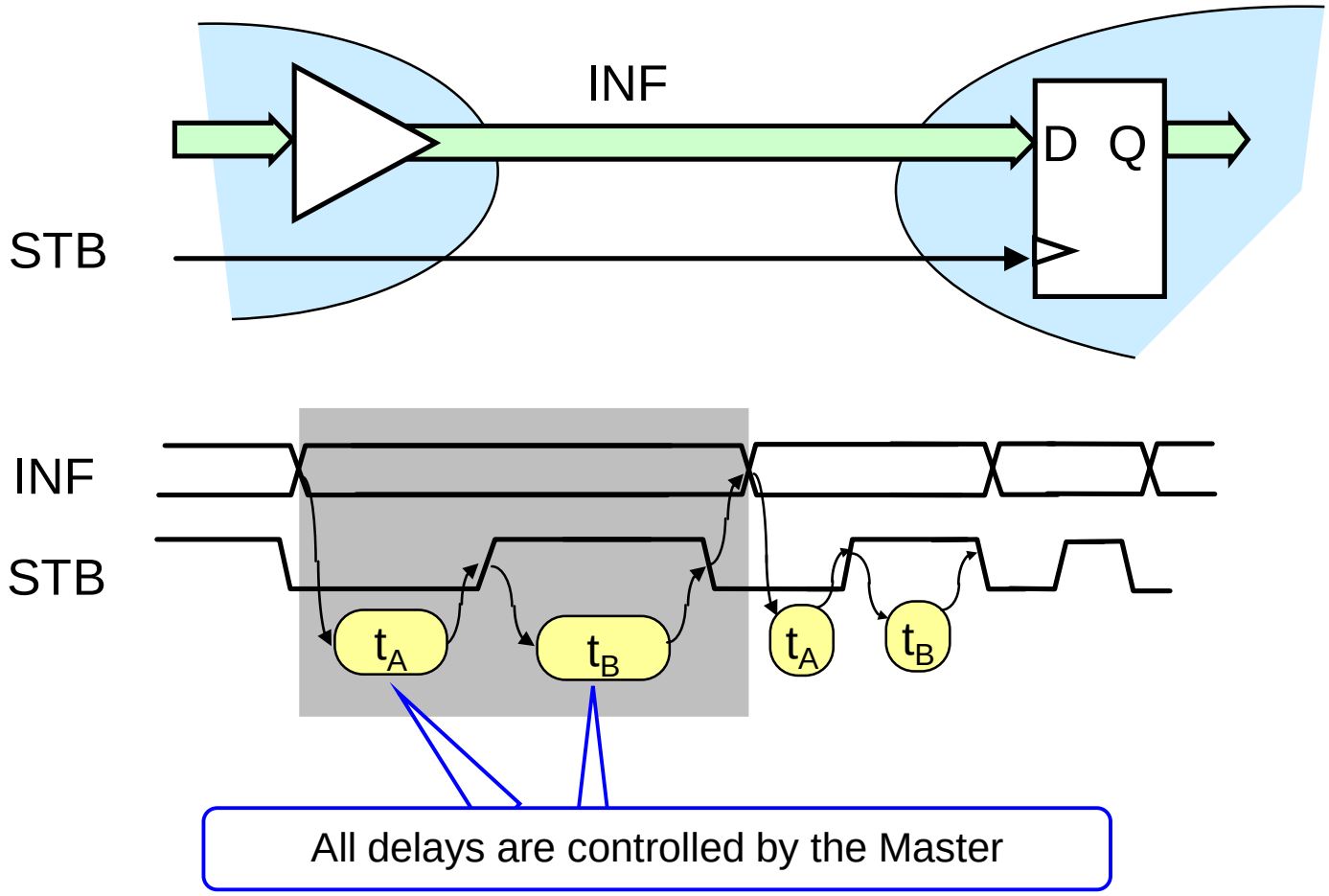
Delays in asynchronous cycles

- Delays in the operation sequence
  - $t_C = t_{SU} + t_H$ – depends on destination register parameters
  - t_K (skew time) – depends on the electrical level
- No assumptions are needed on timing (except t_K)
  - The sequence adapts to the destination speed
- STB / ACK interlacing is called Handshake (confirmation)
  - The asynchronous cycle is “with handshake”
- Minimum cycle time (write)
  - $t_{CYCLEmin} = t_K + t_{SU} + t_H + \mathbf{4\ t_{TXmax}}$

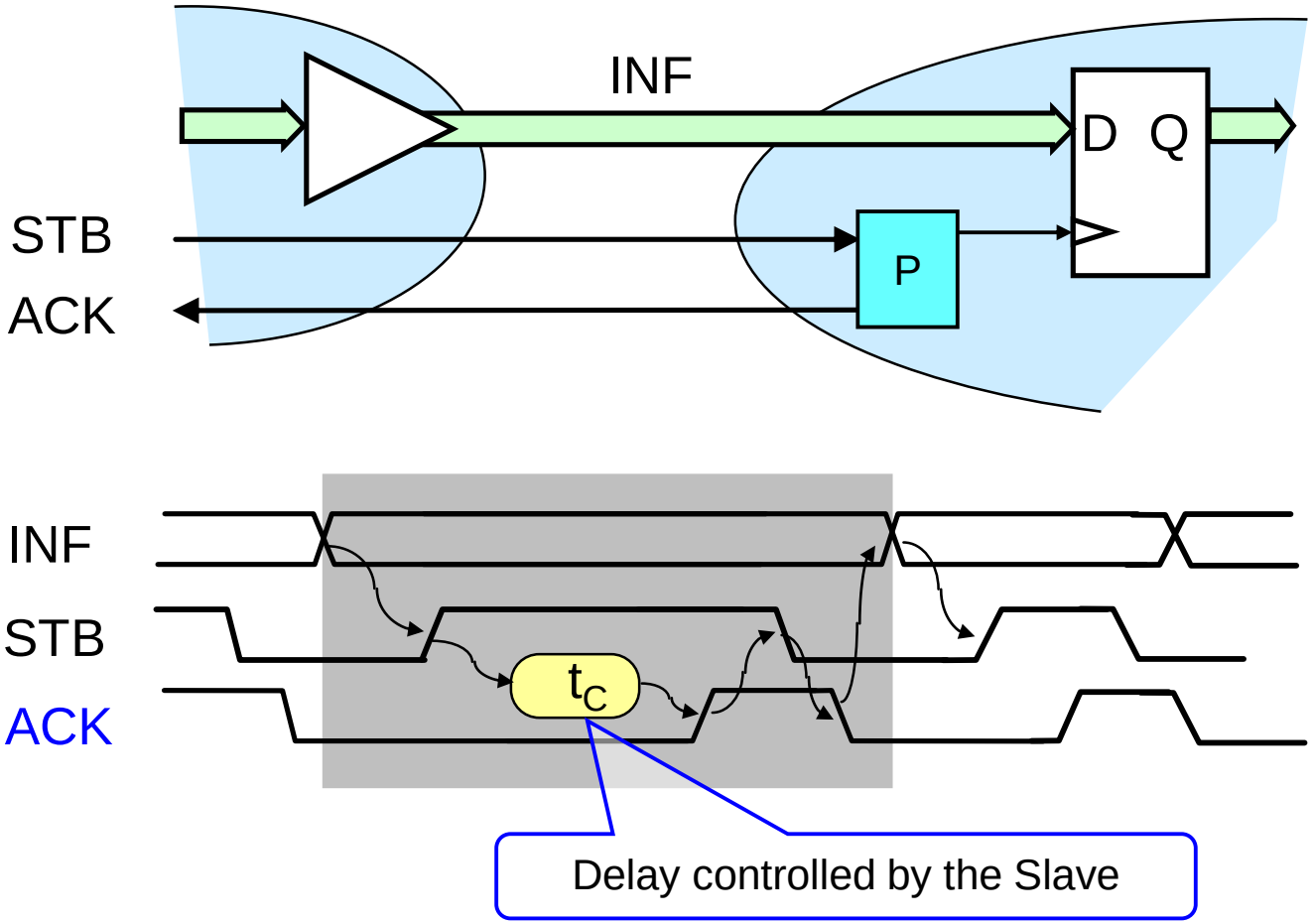
Sequence of cycles

- To transfer multiple information packets
  - Use a sequence of basic cycles
- A new cycle can begin when
  - The previous one is over
  - The control signals have returned to their inactive states
- In these conditions we have
  - Cycle CLOSED or COMPLETED

Sequence of synchronous cycles



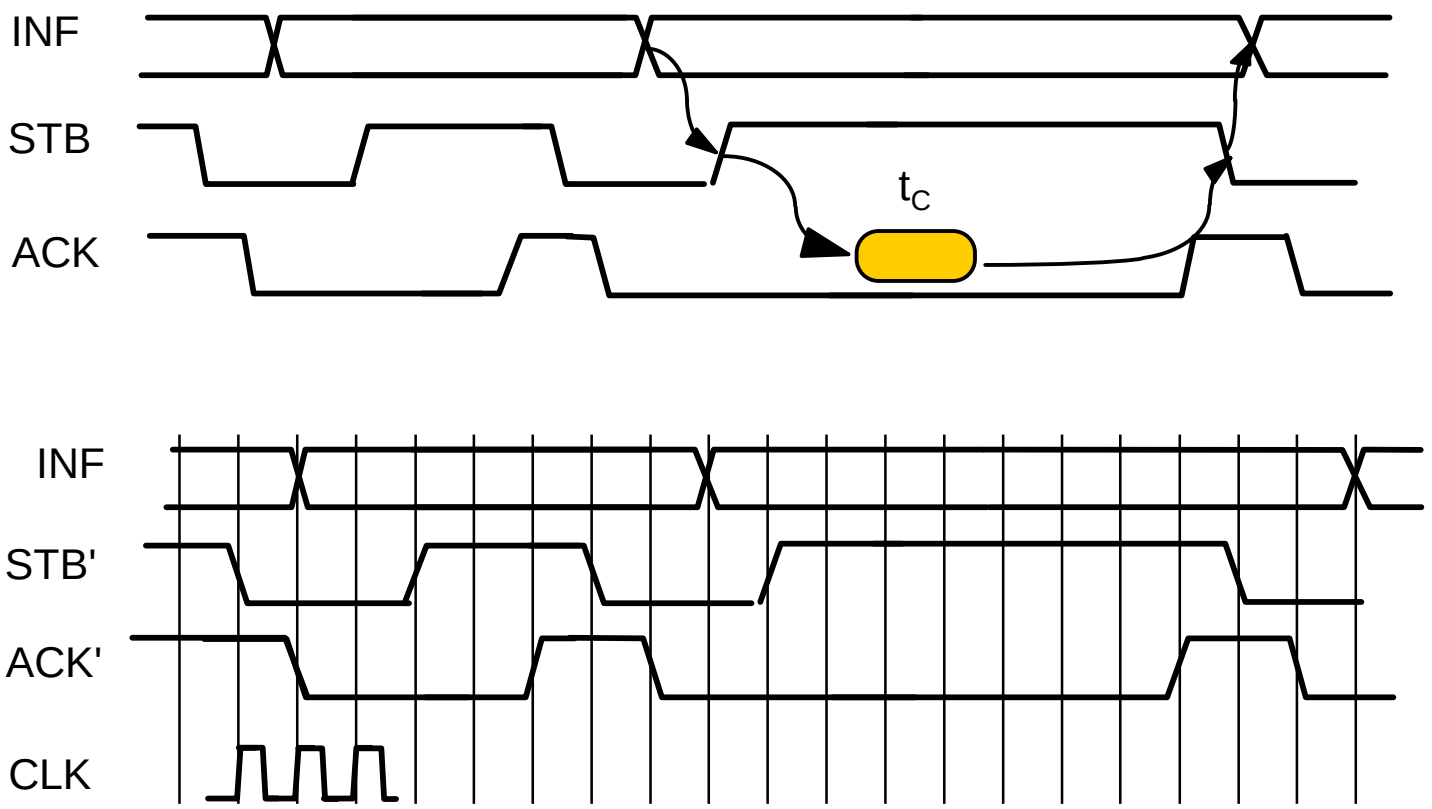
Sequence of asynchronous cycles



Timed cycles (with clock)

- The asynchronous protocol is implemented with synchronous circuits
- Cycle protocol → Finite State Machine (FSM)
  - Asynchronous FSM: can change state at any time, following the input or internal changes (all previous examples are of this type).
    - Continuous time flow
  - Synchronous FSM (with clock): can change state only at the edges of the clock (if certain conditions are met). CADENCED (CLOCKED) protocol
    - Time changes only in multiples of clock period
    - All units are synchronized by a single clock signal

Timed cycle: timing

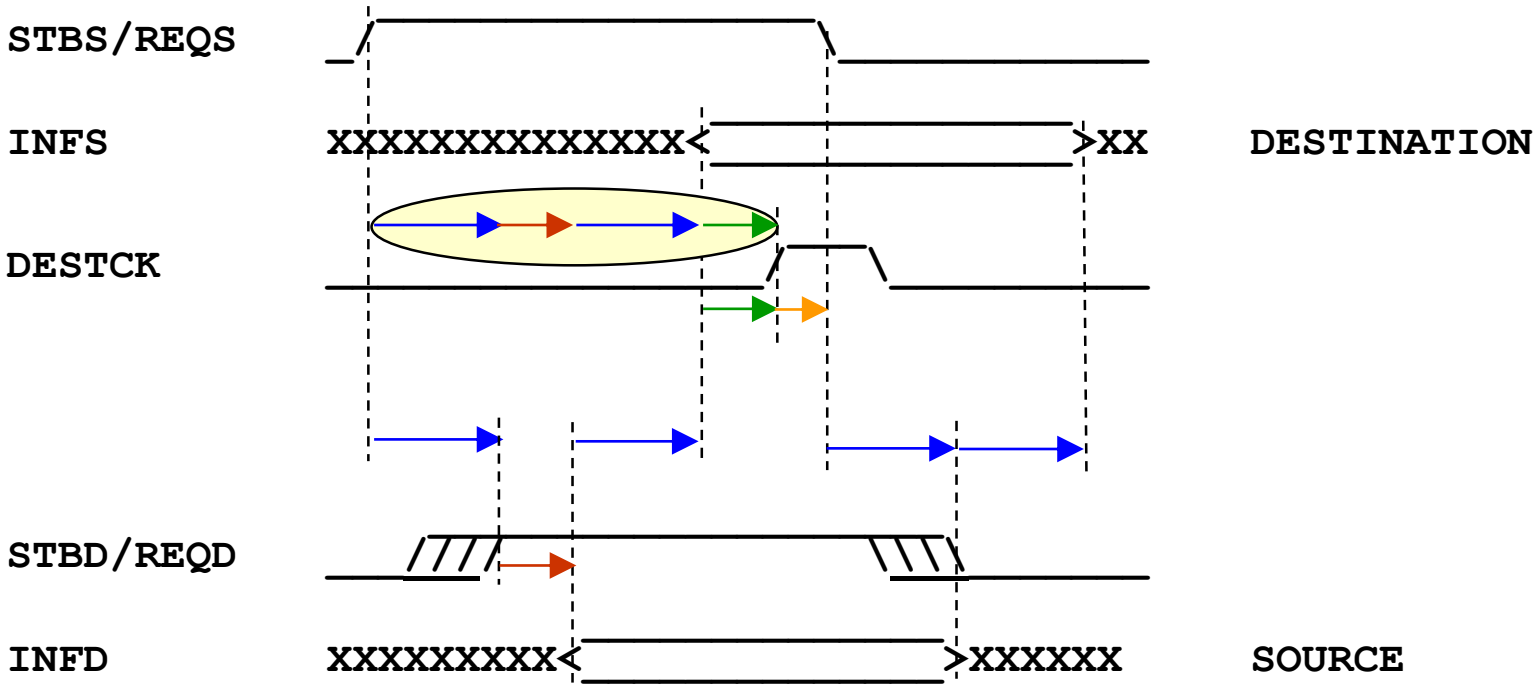


STB', ACK' : timed signals corresponding to STB, ACK

Reading and Writing Operations

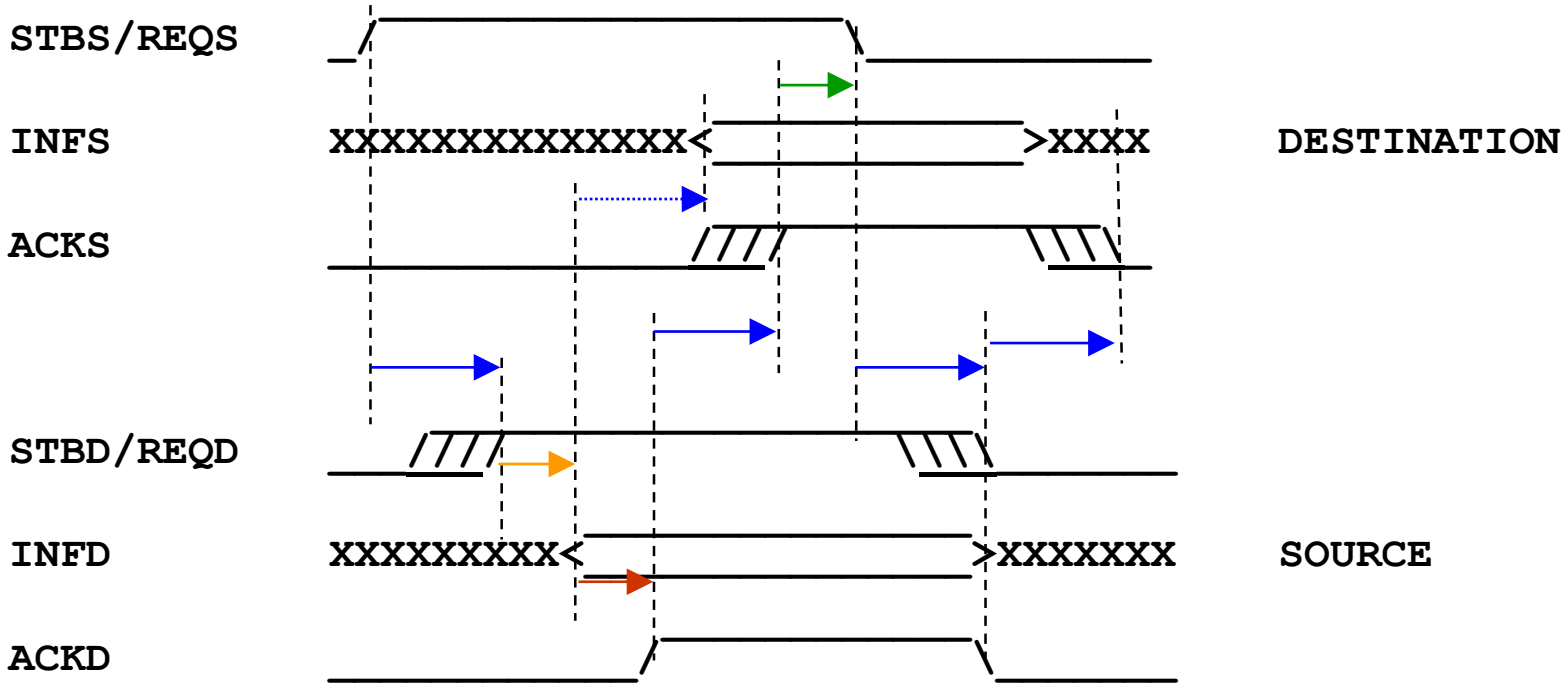
- The first command defines the modules as Master/Slave
  - The Master sends the first command (cycle start)
  - The Slave responds to the commands sent by the Master
- The information (INF) transfer direction defines the Source/Destination
  - The Source *provides* INF
  - The Destination *receives* INF
- Transfer of INF from Master to Slave
  - WRITE cycle
- Transfer of INF from Slave to Master
  - READ cycle

Example 3: synchronous read cycle



Total read time: $t_{RD} = t_A + t_{SU} + t_H + 4\ t_{TXmax}$

Example 4: asynchronous read cycle



Total read time: $t_{RD} = t_k + t_1 + t_2 + 4\ t_{TXmax}$

t_1 → setup and hold of destination register

t_2 → access time of data source

Source-synchronous loops

- In the WRITE cycles, INF and STB move in the same direction, Master → Slave
  - Higher performance when the protocol is synchronized by the data source (Master when writing, Slave when reading) SOURCE SYNCHRONOUS
  - Example: SSTL-2 (stub series terminated logic) and later, used for DRAM
- Two important timing parameters
  - Information latency (waiting to get the INF)
    - Depends on the transmission time t_{TX} and the skew t_K
  - Cycle duration (inverse of cadence = # of cycles/second)
    - Depends on the skew t_K, but not on the transmission time t_{TX}

Summary of cycle types

- Basic synchronization techniques
  - Synchronous — fixed timing (set to worst case)
  - Asynchronous — adaptive timing (waiting for ACK)
- Any of these can be
  - Without cadence — synchronized directly by signals (asynchronous FSM)
  - With cadence — synchronized by a clock signal (synchronous FSM)
- Maximum performance (such as throughput)
  - Source synchronous — write-only protocol

Services provided by the cycle level

- Objectives
  - Transfer of information units (byte, word, ...)
  - Respect the destination register timings
- The information comes from one SOURCE, and is stored/used in one DESTINATION
  - Specific protocols can manage 1→ N or N → M transfers (N-partner protocols)
- The elementary operations are
  - READING and WRITING
- These services require
  - TIME and ENERGY

Lecture C4 – final test

- What is the objective of cycle-level protocols?
- Describe cycle-level basic synchronization techniques.
- Plot the time diagrams for synchronous read operation.
- Draw the control signals for asynchronous READ and WRITE operations with destination registers activated by levels and activated by transitions.
- What protocol can transfer most information/time?
- Benefits of Source-Synchronous protocols?
- Is the synchronous protocol faster or slower than the asynchronous? Why?
- Services provided by the cycle-level protocols?