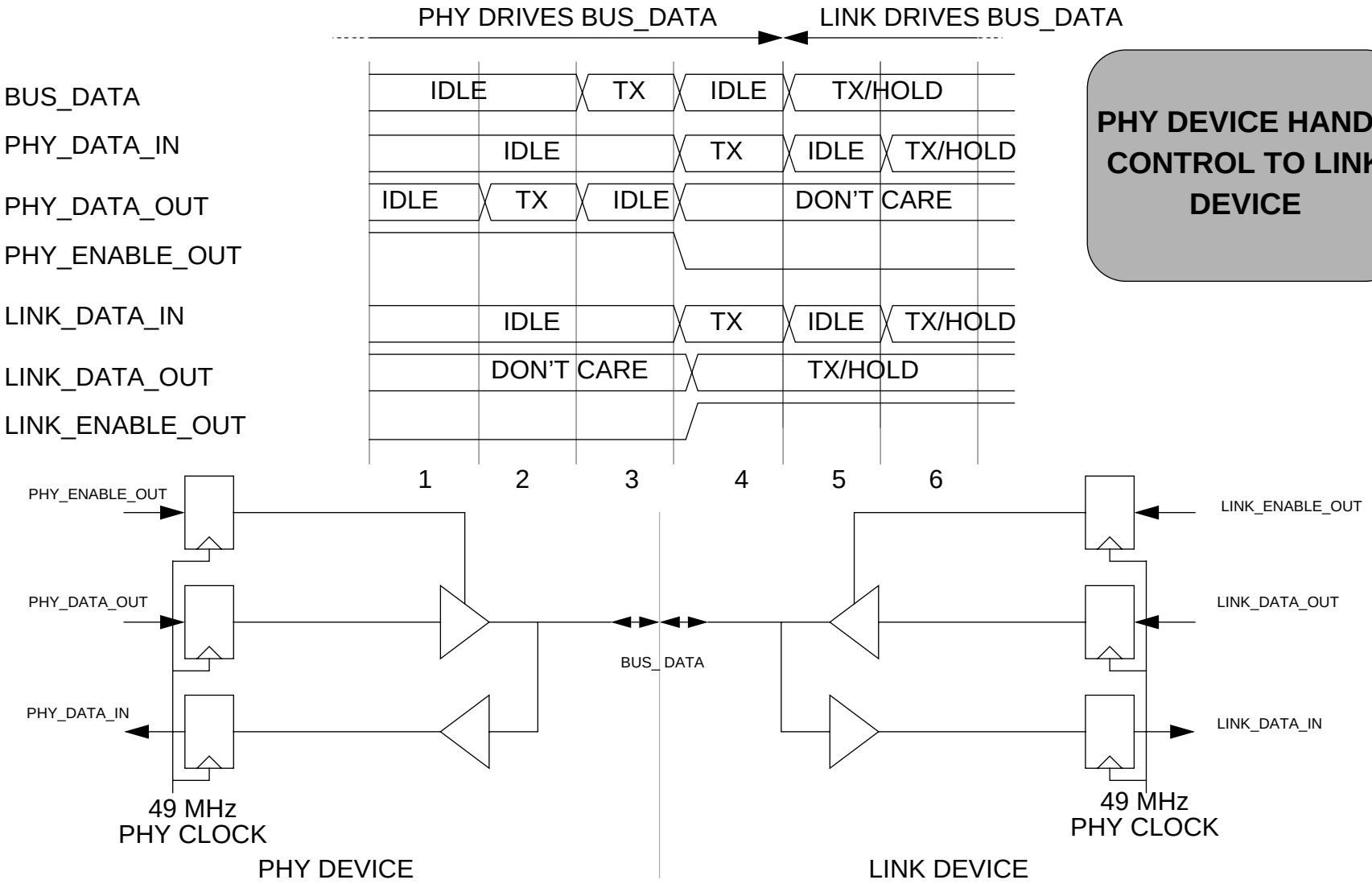


PHY DEVICE HANDS CONTROL TO LINK DEVICE

Description of Interface Behaviour for the PHY device handing control to the Link device to allow packet transmission

Cycle 1
PHY device is arbitrating on the 1394 bus for a previous Link request for transmission.

Cycle 2
PHY device has won arbitration on the 1394 bus. It internally asserts TX on its internal CTL lines (PHY_DATA_OUT) to indicate this to the Link.

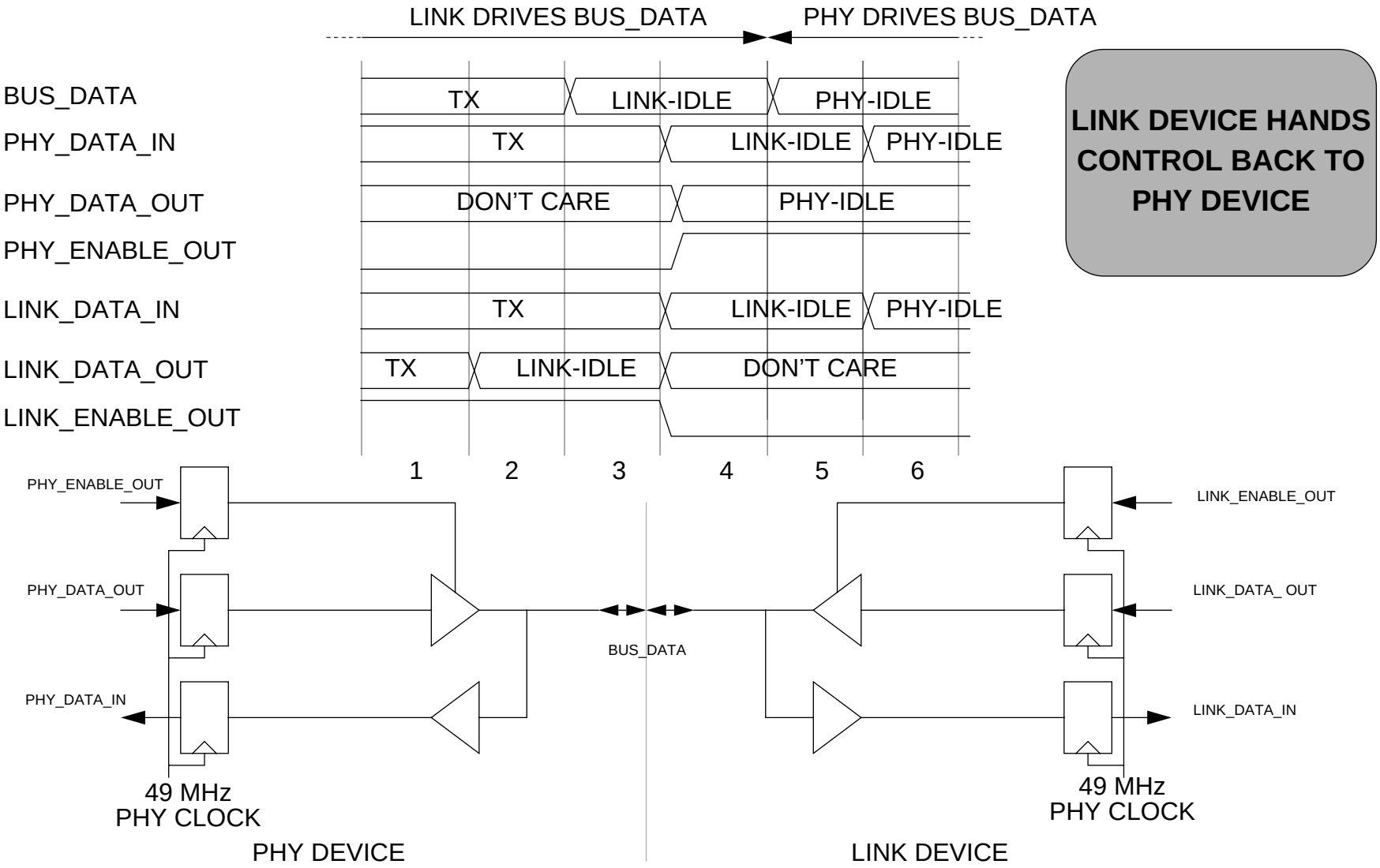
Cycle 3
The PHY device's TX indication is now visible on the PHY-Link bus (BUS_DATA). In this same cycle the PHY device drives IDLE onto its internal CTL lines (PHY_DATA_OUT)

Cycle 4
The PHY device now hands control over to the Link. It de-asserts its internal PHY_ENABLE_OUT, and can now drive any value on its internal PHY_DATA_OUT lines. Note that the PHY device is still driving the BUS_DATA bus during this cycle since the PHY_ENABLE_OUT signal is registered.

The Link 'sees' the PHY device's indication of TX during this cycle (LINK_DATA_IN). Turning around within the same cycle, it immediately drives its own LINK_ENABLE_OUT, and drives TX/HOLD on its LINK_DATA_OUT. Note that the Link cannot drive IDLE during this cycle onto BUS_DATA, since its output path is registered.

Cycle 5
The bus turnaround occurs within this cycle. The PHY device has de-asserted its tristate enable, allowing the bus to float. The Link device has now asserted its tristate enable, and so drives the PHY-Link bus. If the Link is driving TRANSMIT on its CTL lines, it also drives the first cycle of data on its DATA lines. This turnaround must be completed before the setup time to the PHY, such that the PHY device can know in the next cycle whether the Link device is transmitting or holding.

Cycle 6
The PHY device 'sees' in this cycle whether the Link drove TRANSMIT or HOLD. If TRANSMIT, it begins to resynchronise the data from the Link device and drive it onto the 1394 bus.


LINK DRIVES BUS_DATA
PHY DRIVES BUS_DATA
BUS_DATA
PHY_DATA_IN
PHY_DATA_OUT
PHY_ENABLE_OUT
LINK_DATA_IN
LINK_DATA_OUT
LINK_ENABLE_OUT
TX
LINK-IDLE
PHY-IDLE
DON'T CARE
1
2
3
4
5
6
LINK DEVICE HANDS CONTROL BACK TO PHY DEVICE
PHY_ENABLE_OUT
PHY_DATA_OUT
PHY_DATA_IN
BUS_DATA
LINK_ENABLE_OUT
LINK_DATA_ OUT
LINK_DATA_IN
49 MHz
PHY CLOCK
PHY DEVICE
LINK DEVICE

Description of Interface Behaviour for the Link device handing control back to the PHY device following packet transmission

Cycle 1
Link device is transmitting its last cycle of valid data internally on LINK_DATA_OUT for the current packet.

Cycle 2
Link device transmits its first cycle of IDLE onto its internal LINK_DATA_OUT to indicate that it is now complete. The last cycle of valid data from the Link to the PHY is on the PHY-Link bus (BUS_DATA) during this cycle.

Cycle 3
The Link transmits another cycle of IDLE onto its internal LINK_DATA_OUT. During this cycle, the first cycle of IDLE driven by the Link device is actually visible on the PHY-Link bus (BUS_DATA).

Cycle 4
The Link device is now complete. It deasserts its own PHY_ENABLE_OUT and drives DON'T CARE onto its PHY_DATA_OUT.

This is the first cycle that the PHY device actually 'sees' that the Link device is now complete (PHY_DATA_IN). Turning around within the same cycle, the PHY device now drives IDLE onto its own PHY_DATA_OUT bus, and asserts its tri-state enable signal.

Note that because of the registering of the PHY_ENABLE_OUT signals that the Link device is still actually driving BUS_DATA during this cycle.

Cycle 5
The bus turnaround occurs within this cycle. The Link device has de-asserted its tristate enable, allowing the bus to float. The PHY device has now asserted its tristate enable, and so drives the PHY-Link bus.

Cycle 6
The Link-PHY bus is now in IDLE for its second cycle.

Note: In the preceding diagram, a distinction is made between PHY-IDLE and LINK-IDLE. In reality, there is no such distinction (both are 00) and the only purpose in indicating a difference is for clarity of the diagram.