# Ultra Low Power Bus Design

(Needs better name...)

## 1 Overview

The ULPB defines two *physical* types of nodes: member nodes and a control node. An instantiation of ULPB must have one and only one control node and may have any number of member nodes (N $\geq$ 1).

During a transmission, ULPB defines three *logical* types of nodes: a transmitting node, a receiving node, and forwarding nodes. During a transmission, there must be exactly one transmit and one receive node. Any number (N $\geq$ 0) of forwarding nodes are permitted.

## 2 Physical Design

The physical design requirements for a member node and a control node are different, requiring more pads from the control node.

### 2.1 Member Nodes

A member node requires 3 signals:

`DIN` – Data In

`DOUT` – Data Out

`CLK` – Clock *(2 Pads)*

A member node must buffer and forward signals from the `DIN` pin to the `DOUT` pin. Designs should attempt to minimize effects on latency between `DIN` and `DOUT`, in a connected ULPB, the maximum `DOUT` to `DOUT` latency[1] permitted is 10 ns.

The `CLK` signal is a shared bus. Any physical design must be able to both receive and forward a `CLK` wire. On a common chip, this is simply a shared bus. Chips designed for wirebonding, however, should be sure to expose two `CLK` pads and electrically connect them on die.

### 2.2 Control Nodes

A control node requires 5 signals:

`DIN` – Data In

`DOUT` – Data Out

`CLK` – Clock

`DSNOOP` – Data Snoop

`MODE` – Mode select

The control node is responsible for the generation of the `CLK` signal. As such, it is not required to be able to physically forward the `CLK` signal (it is permitted (though not required) to only have 1 pad).

The exact functionality of the `DSNOOP` and `MODE` pins is not yet defined.

[1] The amount of time taken to propagate a change in the output of the *previous* link in the data loop to the output of the next link in the data loop. This time includes all of the internal forwarding logic from `DIN` to `DOUT`, as well as any pin/wire capacitance that the previous `DOUT` must overcome to drive the next link's `DIN`.

## 2.3 Bus Connections

`CLK`

- The clock pins should be wired together to form an electrically connected bus.
- The only `CLK` driver is on the control node. Designs should consider the drive strength of the control node, overall bus capacitance, etc.

`DIN`, `DOUT`

- The data lines should be connected in a round-robin fashion, the `DOUT` of one chip connected to the `DIN` of the next.
- The connection of data lines must form a loop when connected correctly.
- There are no requirements for the placement of nodes in the data loop, but the ordering will have an impact on bus arbitration. See Section 3.2 for more details.

# 3 Protocol Specification

During normal operation, ULPB remains in Bus Idle (3.1). A transmission begins with Arbitration (3.2), then Message Transmission (3.3), then Message End Sequence (3.4), and Message Acknowledgement (3.5). A unique feature of ULPB is that both the Message End Sequence and the Message Acknowledgement are interpreted as Bus Reset (3.6) signals by any forwarding nodes. All events except initiating arbitration occur on the rising edge of `CLK`.

The Bus Reset signal may occur at any time and supersedes any current state, resetting all nodes on a ULPB. After receiving a Bus Reset, ULPB returns to idle state.

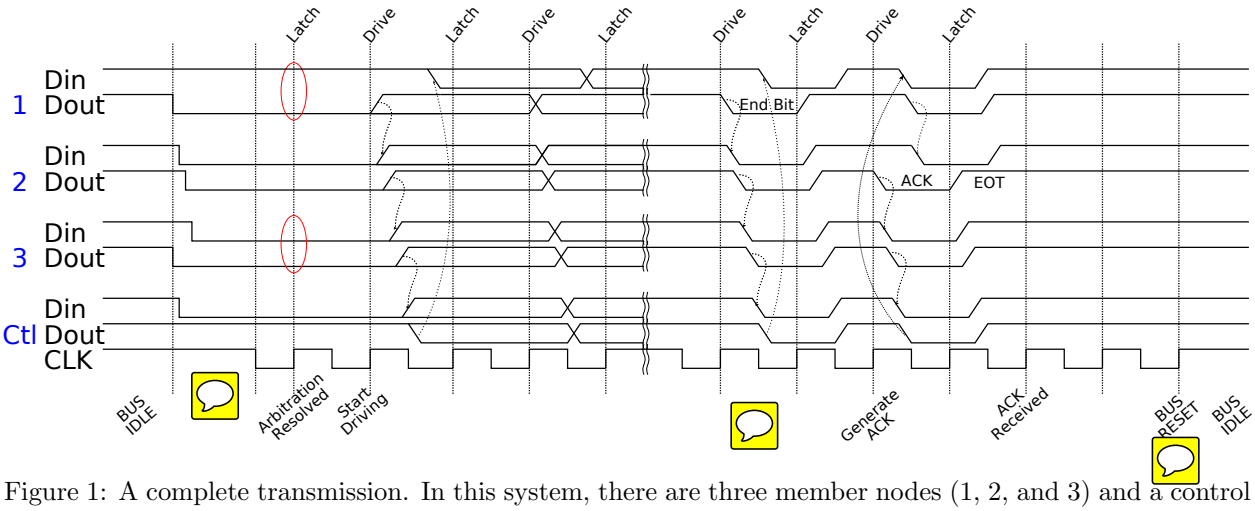


Figure 1: A complete transmission. In this system, there are three member nodes (1, 2, and 3) and a control node. The data loop is connected Ctl → 1 → 2 → 3 → Ctl. This demonstrates arbitration between member nodes 1 and 3, which node 1 wins. Node 1 then transmits data to node 2, which ACK's the data.

## 3.1 Bus Idle

In ULPB Bus Idle, all lines (`CLK`, `DIN`, `DOUT`) are high. All member nodes are in forwarding state and the control node is waiting to begin arbitration.

## 3.2 Arbitration

To begin arbitration, the bus must be in idle state.

To request to transmit on the bus, a node should pull its DOUT line low. All member nodes remain in forwarding state during arbitration, thus when their DIN line goes low, they are obligated to pull their DOUT line low. The control node, however, does **NOT** pull its DOUT line low in response to its DIN line going low. The control node only pulls its DOUT line low when it wishes to transmit on the bus. When the control node's DIN line goes low, it will pull the CLK line low and hold it low for some period $t_{long}$.

By the end of the period $t_{long}$, the effect of the arbitration is achieved. A member node is in one of three possible states:

1. CLK low, DIN high, DOUT high – Lost arbitration (didn't participate)
2. CLK low, DIN high, DOUT low – Won arbitration
3. CLK low, DIN low, DOUT low – Lost arbitration (either lost, or didn't participate)

If the control node wishes to transmit on the bus, all member nodes will be in the third state. Note this arbitration protocol introduces a *topology-dependent priority*. Firstly, the control node has a greater priority than any member node as its DOUT will propagate around the entire data loop. The priority of the member nodes is inversely related to their proximity to the control node in the data loop. That is, the furthest node from the control node, the node whose DIN is connected to the control nodes's DOUT, has the highest priority of member nodes. The closest node to the control node, the node whose DOUT is connected to the control node's DIN, has the lowest priority.

At the end of $t_{long}$, the control node drives the CLK line high. The arbitration phase ends on this rising edge. Whichever node won the arbitration transitions from a forwarding node to a transmitting node. If it is not the transmitting node, the control node also acts as a forwarding node for the duration of the transmission.

***Important:*** For consistency, and the simplification of Bus Reset-detection, the rising edge that established the arbitration winner is also treated as a LATCH state. This means the values at each node's DIN line cannot change at the next rising CLK edge. In implementation, the transmitting node (the node that won arbitration) and the control node should not change their DOUT MUX from arbitration mode to transmission mode until the *next* rising clock edge (the first DRIVE state edge).

## 3.3 Message Transmission

During transmission, ULPB alternates between the DRIVE state and the LATCH state on every rising CLK edge. Note, for forwarding (and receiving) nodes, the DIN and DOUT lines are **NOT** clocked, nodes should forward data signals immediately.

The first rising edge after arbitration starts the first DRIVE state. After CLK goes high, the transmitting node should set its **DOUT** line to 0 or 1 as appropriate. At the next rising edge, ULPB switches to LATCH state and nodes latch the value on their DIN lines. The next rising edge starts the next DRIVE state, and the transmitter pushes its next bit onto DOUT.

The first sequence of bits pushed onto ULPB are the *address* bits. All nodes on a ULPB should listen until their address:

Matches: Node promotes itself from forwarder to receiver.

Does Not Match: Node remains forwarder for the rest of the transmission.

There is no protocol-level delineation between address and data bits. The transmitting node sends address+data as a continuous stream of bits until it terminates the message with a Message End Sequence.

## 3.4 Message End Sequence

Picking up transmission from the last bit of data the transmitting node wishes to send, the transmitting node pushes its last bit on its DOUT line during the DRIVE state, and it is latched by ULPB during the **latch** state. At the next DRIVE state, the transmitting node drives its DOUT line low. At the rising edge of the CLK that starts the next LATCH state, all ULPB nodes latch 0. Also during this rising edge, the transmitting

node drives its DOUT line high. At the next rising CLK edge, which starts the next DRIVE state, nodes will observe that the data line changed since the LATCH state. All forwarding nodes will treat this transition as a Bus Reset signal. The receiving and control nodes, however, will recognize this sequence as a Stop.

At the next rising edge after transmitting the Stop sequence, the transmitting node should begin forwarding its DIN to its DOUT.

After any Bus Reset (which Stop qualifies as), the control node will drive 2 more pulses on the CLK line. For the transmitting and receiving nodes, the DRIVE / LATCH alternation continues.

## 3.5 Message Acknowledgement

At the rising CLK edge of the DRIVE state that signalled Stop, the receiving node should set its DOUT line to low. Note this is a relatively fast turnaround for the receiving node. At the rising CLK edge, the receiver must:

1. Observe the Stop condition
2. Establish that it is prepared to ACK the transmission
3. Drive its DOUT line low

In practice, it is recommended that a receiving node constantly keep its ACK/NAK state prepared, such that when a Stop condition is detected, the receiver's DOUT MUX can switch from the DIN line to the current ACK/NAK state with low latency.

During the next rising CLK edge—which begins a LATCH state, and the first of the 2 Bus Reset pulses—all nodes with latch 0 from the data loop, while the receiving node drives its DOUT line high. This Acknowledge sequence is the same as the Stop sequence, and is treated as a Bus Reset event by all nodes except the transmitting node.

After Acknowledgement/Bus Reset sequence, the control node will drive 2 pulses on the CLK line, after which the ULPB returns to Bus Idle.

## 3.6 Bus Reset

The ULPB constantly alternates between the LATCH STATE and the DRIVE STATE on every rising clock edge. Coming from Bus Idle, the first rising clock edge is defined as a LATCH STATE (this is the same rising edge that established arbitration, see the note in 3.2). The transmitting node sets DOUT to its new value on the next rising edge, beginning the DRIVE STATE. The implication of this is that at the CLK edge that starts the DRIVE STATE, *the value of DIN should not have changed.*

If DIN has changed from the LATCH STATE to the DRIVE STATE, this indicates a Bus Reset sequence. Upon receipt of a Bus Reset, all member nodes should switch to forwarding mode, **except**

A transmitting node sending a Message End Sequence.

A receiving node receiving a Message End Sequence.

(Note: Both transmitter and receiver should revert to forwarding mode after sending / receiving a Message Acknowledgement).

When a Bus Reset is received, the control node will drive two more clock pulses—one more LATCH STATE and one more DRIVE STATE. Depnding on the DIN during these cycles, the control node will:

| | |
|---|---|
| 1 1 | Return to Bus Idle. |
| 0 1 | This corrosponds to the Stop sequence and Acknowledgement sequence. In proper operation, a control node should see at most two of these in a row. If a third is observed, the control node will begin forcefully resetting the bus, as outlined below. |
| 1 0 | This state should not occur in normal operation. In response, the control node will drive two more pulses, ignoring its `DIN` line and generating a Bus Reset signal. It will then generate two more clock pulses with its `DOUT` line high, repeating this process until it samples two 1's on its `DIN` line. |
| 0 0 | |

# 4 Document Revision History

Revision 0.1 – Dec 19, 2012

Initial revision