0.1 Flow Transactions

ULL protocol designers play around with 32 bits as the minimum unit of transactional transfer, but experiments demonstrate the difficulty of making this consistently reliable i; the general consensus is that modern SerDes' work best with $\geq$ 64 bit (8 Byte) slices/flits. Ethernet has a minimum frame size of 64 bytes (although only 42 bytes were available for the payload).

We therefore choose a *fixed* 64 Byte frame for the Shannon Slots, but make them *pre-emptable* so that even the minimum size frame does not need to occupy space on the wire, increase latency, or FPGA processing steps, when the receiver has something more important it wishes to send (e.g. local status messages sent in the background can be pre-empted, giving way to a two phase commit (2PC) transaction).

Some transactional systems are sensitive to making transactions reliable, but don't mind missing events, such as highly perishable market data. We might call these one-phase commit (1PC) transactions. These can be made to flow at maximum line rate, even though each individual slice is being acknowledged. This is particularly important in HFT for example.

We therefore provide the following "flow" transactions in the encoding scheme:

0.1.1 Flow Unit Encodings

To enable ultra-low-latency transaction processing, the receiver must begin interpreting and dispatching semantic units (operator + operand) before the full 64-byte frame has arrived. This is made possible through lightweight inline encodings that declare, in the first slice of a transaction, the total number of slices that comprise that flow unit.

These encodings allow the receiver to pipeline semantic processing based on declared intent rather than full-frame arrival, dramatically reducing end-to-end transaction latency while preserving reliability.

1. **One 1-slice Flow Unit (4B payload)**
   `00` Indicates this flow unit consists of 1 slice.
2. **One 2-slice Flow Unit (12B payload)**
   `01` Indicates 2 slices are part of this flow unit. The receiver counts down remaining slices before handoff.
3. **One 4-slice Flow Unit (28B payload)**
   `10` Indicates 4 slices make up this flow unit. The receiver pipelines semantic interpretation during arrival.
4. **One 8-slice Flow Unit (60B total payload)**
   `11` Indicates 8 slices make up this flow unit. The receiver waits for

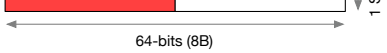


Figure 1: 1 Slice Flow Subtransaction

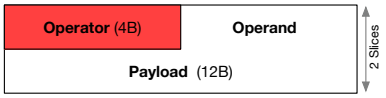


Figure 2: 2 slice Flow SubTransaction

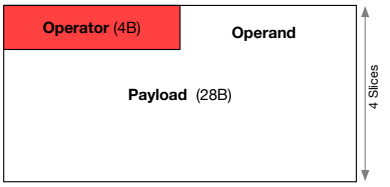
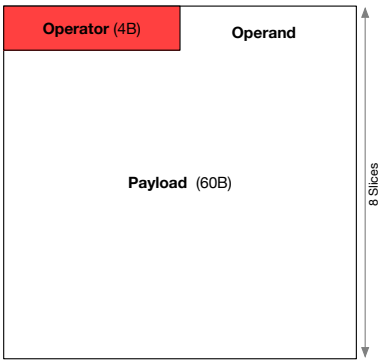


Figure 3: 4 4 slice Flow SubTransaction with 28B payload (operand)



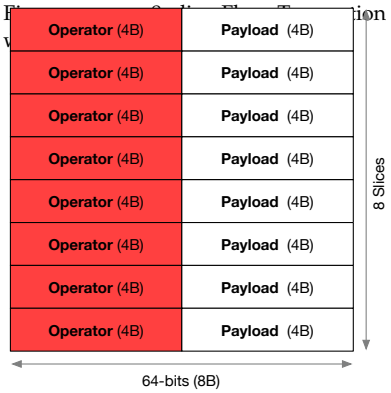


Figure 5: 8 independent Flow Transactions in a one frame

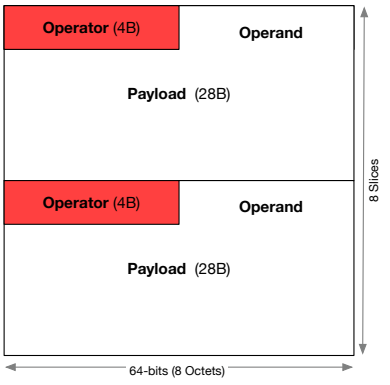
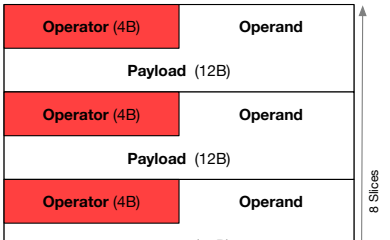


Figure 6: 2 $\times$ 4 slice Flow Transactions



the full frame before semantic interpretation.

0.1.2 Mixing and Matching Flow Transactions

You can also mix them in the same frame, but remember, they can only be used for One-Phase-Commit (1PC) in a single stream of transactions. This is because 1PC requires only one "round trip", whereas 2PC requires two round trips (although this scheme can be made to work for 2PC, and perhaps 4PC, but they have not yet been tested).

Flows	Operator	Operand	Efficiency
1	4	4	50%
1	4	12	75%
1	4	28	87.5%
1	4	60	93.75%
2	4	4	100%
2	4	12	150%
2	4	28	175%
2	4	60	187.5%
4	4	4	200%
4	4	12	300%
4	4	28	350%
4	4	60	375%
8	4	4	400%
8	4	12	600%
8	4	28	700%
8	4	60	750%

Table 1: Transaction efficiency by operator and operand size.