При проектировании данного модуля необходимы следующие вещи (По моему мнению):

* Необходимо использовать буферы FIFO для пакетов данных (количество данных буферов равно T\_DATA\_RATIO) и сигнала keep и сигнала last.
* Необходимо написать логику обработки пакета, а также логику обработки транзакции (сигнал last)

Решения, которыми я руководствовался:

* В первую очередь необходимо включить в топ модуль все буферы и сохранить параметризированную логику, следственно нужно через конструкцию generate создать все экземпляры буферов данных, равное глубине данных апсайзера.
* Включить в топ модуль FIFO для last и keep
* Далее необходимо поочерёдно записывать данные в FIFO для этого я создал регистр pointer. К примеру у нас началась передача. Первый пакет данных записывается в первый FIFO (указатель равен 1), далее он записывается во второй FIFO (указатель равен 2), реализовывал через умножение на два. Данный сигнал является push для FIFO. После достижения максимального значения равного T\_DATA\_RATIO возвращается в 1, если пришёл сигнал last также возвращается в 1
* Также был введён регистр counter\_trn\_reg. Он нужен для обработки сигнала keep. К примеру по достижению значения T\_DATA\_RATIO снимается s\_ready и подаётся сигнал pop для записи в FIFO для keep.
* Общая глубина для всех FIFO равна 10. При установке буфером сигнала full снимаю s\_ready.
* Благодаря такой реализации я обеспечиваю “развязывание” логики приёма и передачи.

Но возникла проблема с описанием данного модуля, связанная с тем, что у меня сигнал push\_data\_for\_fifo для FIFO данных ведёт себя неадекватно. В какой-то определённый момент времени он перестаёт выдавать единицу и выдаёт постоянно 0 и 2. Скорее всего проблема связанна с описанием логики в always\_comb. Если у вас будет возможность я хотел бы увидеть решение данной задачи через обратную связь, заранее благодарю вас.