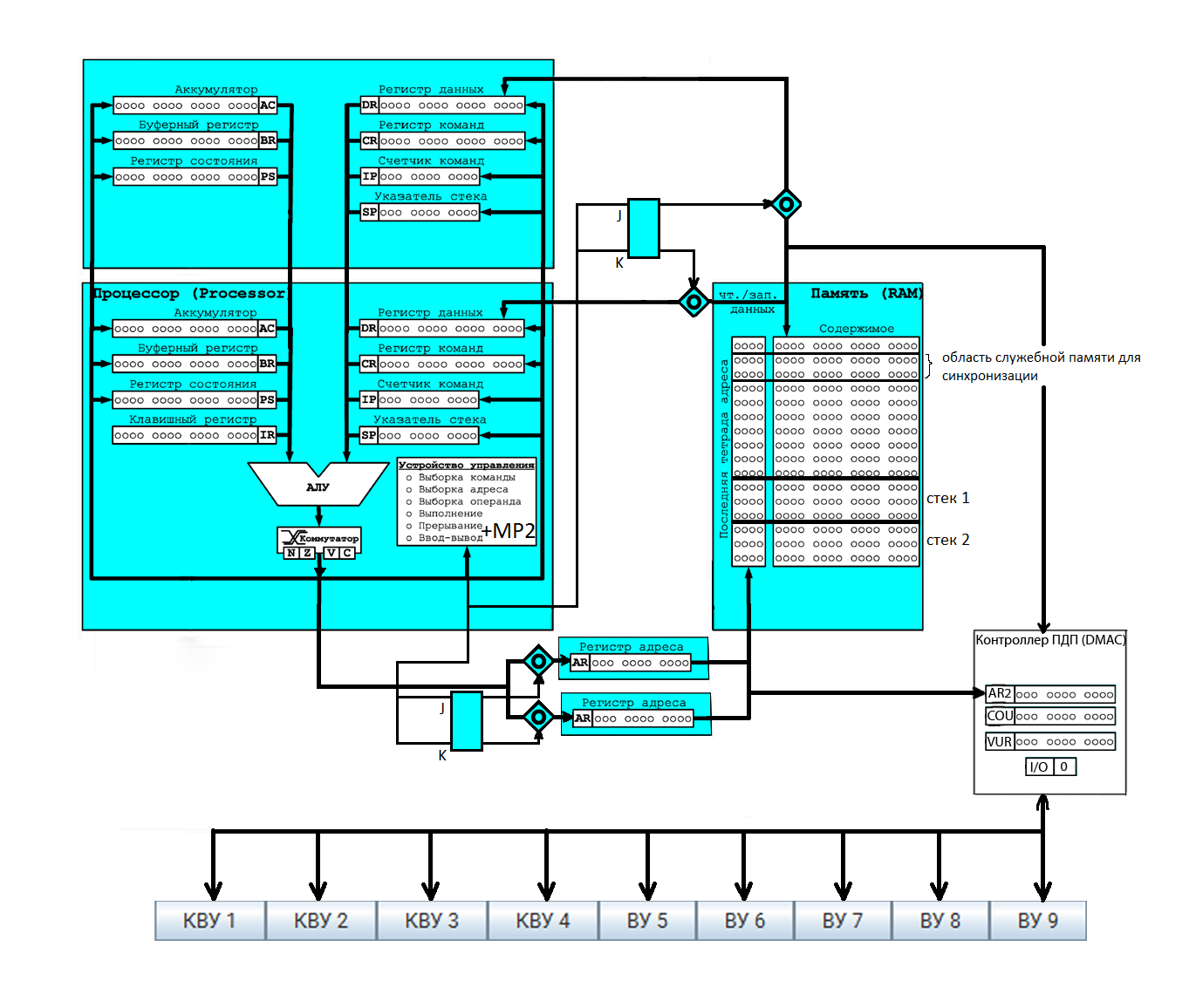
**Дополнительное задание**

**Задание:**

Нарисовать схему модифицированной БЭВМ, в которой одновременно исполняется несколько потоков команд по принципу работы barrel процессоров (<https://en.wikipedia.org/wiki/Barrel_processor>)

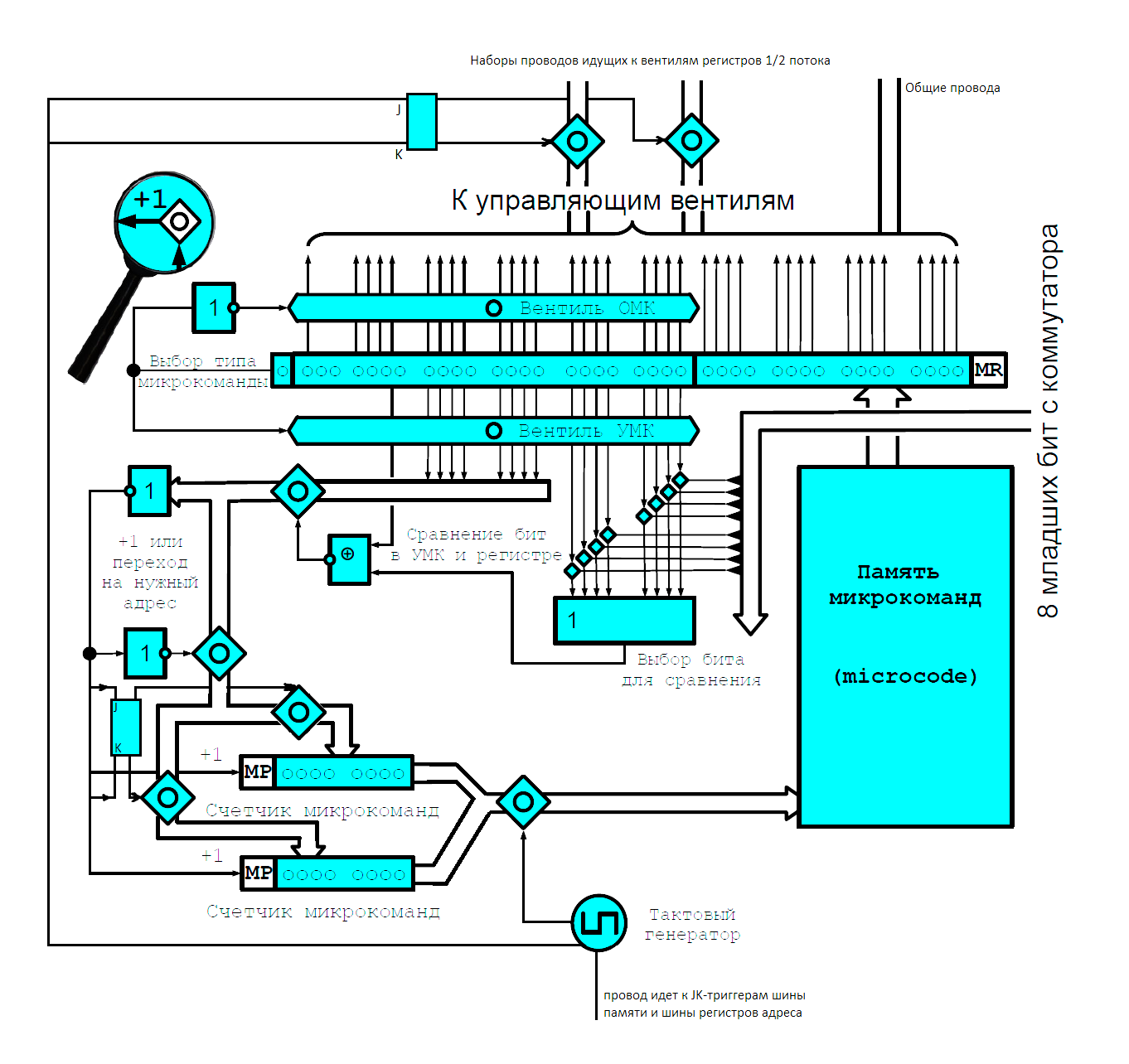
Доработать схему, показать изменения в устройстве МПУ (если нужны дополнительные вентили, то как это отразится в ОМК), добавить пояснения для а) обработки прерываний, б) взаимодействия со стеком, в) взаимодействия с ВУ

Подумать о том, как можно синхронизировать потоки для избежания гонок.



При модификации БЭВМ были добавлены некоторые регистры для второго потока. Процессор переключается между потоками на каждом такте (одна микрокоманда выполняется за один такт). Добавлен общий контроллер прямого доступа к памяти для внешних устройств. Для второго потока добавлен свой счетчик микрокоманд в устройстве управления.  
Чтобы не усложнять архитектуру микрокоманд я добавил JK-триггеры, которые подключаются к устройству управления и переключают шины потоков на каждом такте. Также на каждом такте переключаются между собой шины ведущие к регистрам, присутствующим в каждом из потоков (это сделано для упрощения микрокоманд и для того, чтобы не нужно было выделять свои адреса вентилей для новых регистров, адресация для них происходит точно также как и прежде, переключение происходит автоматически на каждом такте).

Изменения в Устройстве Управления:



**Обработка прерываний**

Для сохранения целостности данных я считаю нужным начинать обработку прерывания после завершения выполнения текущей команды, а не пытаться начинать обработку прерывания между микрокомандами. Прерывание будет также выполняться на всех потоках по принципу работы barrel процессора. В общем случае никаких отличий в обработке прерываний от обработки прерываний обычной БЭВМ нет.

**Взаимодействие со стеком**

Для работы со стеком память условно разделяется на сегменты, которые соответствуют каждому потоку. Для примера, изображенного на рисунке, имеем стек 1 для первого потока и стек 2 для второго потока. При этом ничего нам не мешает обращаться из одного стека в другой используя смещение. Разделение памяти стека на 2 фрагмента позволит избежать лишних гонок во время выполнения.

**Состояние гонок**

Конкретно в БЭВМ с данной архитектурой состояние гонок не может возникнуть, т.к. нет пирамиды памяти и в памяти микрокоманд нет 2 подряд идущих микрокоманд, которые обращаются к памяти.

Однако, в случае если мы увеличим кол-во потоков или добавим пирамиду памяти, то состояние гонок может возникнуть. В таком случае данную проблему можно решить блокировкой области памяти, которая используется другим потоком. Один из вариантов решения этой проблемы я вижу в следующем: выделить область памяти под запоминание используемых ячеек памяти другими потоками и ставить пометку, если какая-то ячейка используется. Соответственно, нужен будет контроллер, который будет заниматься проверками обращений к памяти. И если будет оказываться, что текущему потоку нужна используемая ячейка памяти, то работа текущего потока будет приостанавливаться на время, пока ячейка не освободится. Предположим, что служебный сегмент памяти будет выделен на все ячейки после векторов прерывания.

**Взаимодействие с ВУ**

Проблему взаимодействия с ВУ можно решить добавлением общего контроллера внешних устройств для всех потоков или добавлением контроллера прямого доступа к памяти. На рисунке к многопоточной БЭВМ добавлен контроллер ПДП. Описанный выше метод, который позволит избежать состояние гонок применим и для избежания состояния гонок с внешними устройствами. Область памяти, которая будет использоваться ВУ и будет записана в контроллере ПДП также будет блокироваться для доступа потоками.