1.Как обеспечивается синхронизация записи и чтения в неименованном канале?

Можно представить себе канал как небольшой кольцевой буфер в ядре операционной системы. С точки зрения процессов, канал выглядит как пара открытых файловых дескрипторов – один на чтение и один на запись (можно больше, но неудобно). Мы можем писать в канал до тех пор пока есть место в буфере, если место в буфере кончится – процесс будет заблокирован на записи. Можем читать из канала пока есть данные в буфере, если данных нет – процесс будет заблокирован на чтении. Если закрыть дескриптор отвечающий за запись, то попытка чтения покажет конец файла. Если закрыть дескриптор отвечающий за чтение, то попытка записи приведет к доставке сигнала SIGPIPE и ошибке EPIPE.

Блокировки чтения/записи обеспечивают синхронизацию скорости выполнения двух программ и их одновременное завершение.

Понятия позиции чтения/записи для каналов не существует, поэтому запись всегда производится в хвост буфера, а чтение с головы.

Неименованный канал создается вызовом pipe, который заносит в массив int [2] два дескриптора открытых файлов. fd[0] – открыт на чтение, fd[1] – на запись. Канал уничтожается, когда будут закрыты все файловые дескрипторы ссылающиеся на него.

В рамках одного процесса pipe смысла не имеет, передать информацию о нем в произвольный процесс нельзя (имени нет, а номера файловых дескрипторов в каждом процессе свои). Единственный способ использовать pipe – унаследовать дескрипторы при вызове fork (и последующем exec). После вызова fork канал окажется открытым на чтение и запись в родительском и дочернем процессе. Т.е. теперь на него будут ссылаться 4 дескриптора. Теперь надо определиться с направлением передачи данных – если надо передавать данные от родителя к потомку, то родитель закрывает дескриптор на чтение, а потомок - дескриптор на запись.

3.Как для неименованного канала организовать чтение и запись данных «без ожидания»?

Нужно создать неблокируемые операции чтения и записи.