NSU-2023-T06L1e03

В системах связи и хранения данных часто случается, что ошибки получения (чтения) данных зависят от значений этих данных. Например, жесткий диск использует определенный участок длины магнитной дорожки, чтобы сохранить один бит данных. Для записи и чтения используется намагничивание и электромеханическое устройство (головка чтения-записи). В зависимости от того, как намагничены соседние биты, читающая головка может использовать различия в намагниченности для поиска границ соседних однобитовых секторов. Эти границы не могут быть найдены точно из-за изменений скорости электромотора, вращающего диск, вибраций и других механических и физических явлений.

Аналогично, при передаче данных по сети, временное окно для посылки каждого последующего бита также определяется неточно. Получатель имеет собственный тактовый генератор, который обеспечивает приемлемую точность для получения нескольких последовательных бит. Но, чем больше проходит времени, тем больше могут разойтись часы получателя и отправителя. Один из способов борьбы с этим – гарантировать, что в потоке данных переходы 0–1 и 1–0 будут встречаться достаточно часто, чтобы приемник и передатчик могли синхронизовать свои часы.

Как мы можем обеспечить, чтобы, независимо от того, какие данные передаются и сохраняются, в них всегда будет присутствовать определенный минимум нулей и единиц в каждом сегменте сообщения заданной длины?

Схема кодирования Run-Length-Limited (RLL) – один из способов достичь этого. В этих кодах, каждая последовательность идентичных цифр, например, нулей, не превышает определенной длины. Обычно считается, что длинные последовательности нулей порождают больше ошибок, чем длинные последовательности единиц. Код RLL(k) – это трансформация оригинального сообщения, такая, что гарантируется, что в ней не содержится последовательностей идентичных цифр длины k или больше. Такая трансформация должна быть обратимой, так чтобы оригинальные данные могли быть восста-

новлены по результату. Это достигается внесением избыточности: в каждую достаточно длинную последовательность нулей вставляется дополнительная единица.

Для простоты, рассмотрим код RLL(5)

Алгоритм кодирования RLL

```
input bits:1 A, start, clock
output bits:1 set, B
internal bits:3 counter

counter:=
when start=1 or A=1 or counter=5
        0
else
        counter+1

set = (counter<5)
B = A or (counter=5)
end
```

Сигнал `start`, по существу, служит для сброса схемы к начальному состоянию. Если сигнал поднят в конце такта, счетчик сбрасывается на 0. Сигнал `set` добавлен, чтобы сообщить отправителю, что кодер готов получить следующий бит. Если кодер готов, бит из отправителя просто копируется на входной контакт `B`.

Кодер готов, и сигнал `set` поднимается на каждом такте, кроме ситуации, когда кодер получил пять последовательных нулей. В этом случае, счетчик сбрасывается на 0 и следующий бит не принимается (сигнал `set` опущен); вместо этого, на выходе `B` устанавливается 1. Получатель будет знать, что, поскольку 1 следует за 5 нулями, ее необходимо выбросить, чтобы восстановить исходные данные.

На следующем цикле, если ввод равен 1, это сбросит счетчик к 0, иначе счетчик будет продолжать считать как обычно.

Ваша задача

Используйте макетную плату `NSU-2023-T06L1e03.circ` из присоединенного файла. Постройте RLL(5) кодер в соответствии с вышеприведенными спецификациями. Проверьте ваш дизайн, нажимая входные контакты (`start` или A) для установки нужных значений, затем нажимая контакт `clock`. Вы можете использовать компоненты из любой стандартной библиотеки Logisim, за исключением транзисторов. Для экономии сил, вы можете использовать сумматор и компаратор в дополнение к любым вентилям, регистрам и мультиплексорам.

Как отправлять вашу работу на проверку

Не перемещайте входные и выходные контакты, потому что Logisim присоединяет к ним тестовую схему, основываясь на их положении, а не по имени (это неудобно, но мы ничего не можем с этим сделать). Если вы хотите, вы можете присоединить туннели к контактам и разместить другие концы туннелей в удобные для вас места на макетной плате. Таким образом, вы можете размещать вашу схему удобным для вас образом и, в то же время, быть уверенными, что тестирующий робот будет правильно подсоединен к схеме.

Проверьте устройство, нажимая входные контакты при помощи ручных контролов и записывая ваши наблюдения. Ответьте на это сообщение, присоединив файл схемы с вашим решением.