NSU-2023-T06L3e01

Вы – инженер-электронщик, работающий в компании, занимающейся безопасным хранением. Ваша текущая задача – разработать цифровое устройство для флагманского продукта компании, сейфа для гостиничных номеров.

Ваши коллеги из Отдела Ввода/Вывода уже построили 12-кнопочную цифровую клавиатуру (10 арабских цифр, а также клавиши '`*`' и '`R`') для передней панели сейфа, и 4-значный BCD дисплей с индикатором запирания. Они также установили сенсор закрытия двери и электромеханический замок.

Вам дана макетная плата со следующими туннелями, присоединенными к контактам вышеописанных устройств.

Название пучка	Биты данных	Описание
`button`	4	Нажатая клавиша: `0x0`-`0x9` для цифр, `0xA` для `*`, и `0xB` для `R`. Сигналы валидны, когда `clk` поднят и некоторое время после того, как он опущен.
`clk`	1	Строб клавиатуры. Поднят, когда клавиша нажата, и опущен после того, как клавиша отпущена.
`open`	1	Сигнал сенсора двери. Поднят, когда дверь открыта, и опущен, когда закрыта. Когда сейф заперт, дверь не может быть открыта: попытка поднять этот сигнал приведет к конфликту сигналов на этом проводе. У этого туннеля есть соответствующий входной контакт.

Назначение вашей схемы – поднимать и опускать сигнал `locked`, который управляет актуатором электромеханического замка, а также управлять 4-значным BCD дисплеем, используя 4-битные пучки `D0-D3`.

Другие ваши коллеги из Отдела Контроля Качества присоединили к вашей плате входные и выходные контакты, чтобы робот Cocomaro мог протестировать ваш дизайн. Не беспокойтесь об этих контактах. Входы (`Pad` и `clock`) плавают и не могут вызвать конфликта с другими сигналами. Контакт `D.open` полезен для вас: нажимая на него, вы в любой момент можете видеть, что происходит, когда дверь сейфа открыта (если только она не заперта) или закрыта. Наконец, выходные контакты `Disp` и `D.locked` не воздействуют на ваш дизайн.

Как сейф должен работать

Дисплей представляет собой сдвиговый индикатор, аналоги которого вы могли видеть раньше. Он всегда показывает 4-значное десятичное число. Когда вы нажимаете цифру на клавиатуре, все цифры сдвигаются влево, и новая цифра появляется справа. Например, если дисплей показывал 7648 и вы нажали 3, вы увидите 6483. Клавиша `R` с правой стороны нижнего ряда сбрасывает дисплей в 0000 при любом нажатии.

Как работает, собственно, сейф:

- В начальный момент, индикатор показывает 0000 и дверь закрыта, но не заперта. Сейф в режиме 0. Вы можете открывать и закрывать дверь без ограничений.
- Когда при открытой двери нажата клавиша ‘*’, текущий 4-значный код запоминается в регистре PIN-кода, и сейф переходит в режим 1.
- Когда в режиме 1 при открытой двери вы нажимаете любую кнопку, кроме ‘*’, сейф переходит обратно в режим 0.
- Когда в режиме 1 вы закрываете дверь, она автоматически запирается. Дисплей сбрасывается в 0000.

- В режиме 1 при закрытой двери, нажатие ‘*’ приводит к сравнению кода на дисплее со значением, которое хранится в регистре PIN. Если они совпадают, сейф отпирается и переходит в режим 0. Дверь может открываться и закрываться без ограничений, и не запирается при закрытии.

Процесс проектирования

1. Сначала постройте сдвиговый регистр. Вы не можете использовать готовый из библиотеки *Memory* Logisim, потому что этот компонент запрещен правилами проверяющего робота. Вам нужны четыре 4-битных регистра, организованные в цепочку (конвеер). Первый регистр в цепочке получает код с клавиатуры, а остальные в цепочке получают значение из предыдущего регистра. Вам следует использовать вход регисто `enable` для определения, будет ли цифра сдвинута в этот регистр (обратите внимание, что код с клавиатуры может быть не цифрой!). Добавьте логику для распознавания клавиш ‘`*`’ и ‘`R`’, чтобы они не попадали в регистр. Вам также нужен будет распознаватель для ‘`*`’ в других частях проекта. Также, обратите внимание, что первая цифра в цепочке – это последняя цифра на дисплее, вам нужно разобраться с направлениями сдвига.

2. Постройте конечный автомат. Это схема, которая помнит свое состояние и меняет его в зависимости от предыдущего состояния и входных сигналов. Вам нужен 1-битовый регистр состояния и комбинационное устройство, которое будет определять новое состояние на основе текущего состояния, распознавателя звездочки и сигнала `open`. Постройте таблицу истинности, которая определяет новое состояние на основе трех логических значений. Реализуйте ее методом минимальных произведений. Затем добавьте зависимость от внешнего условия: совпадают коды на дисплее и в регистре PIN, или нет. Временно реализуйте это внешнее условие как отладочный входной контакт и проверьте, корректно ли работает ваш автомат: нажимайте различные клавиши на клавиатуре и смотрите, переходит ли состояние из 0 в 1 и обратно в 0 в соответствии с правилами.

3. Постройте логику управления замком. Самая естественная реа-

лизация основана на RS-триггере. Преобразуйте ниспадающий фронт сигнала `open` в импульс и пропустите этот импульс через вентиль AND, управляемый битом текущего состояния. Это гарантирует, что импульс попадет на вход `S` триггера только в режиме 1. Также, преобразуйте нисходящий фронт сигнала `mode` в импульс, и подсоедините его к входу триггера `R`. Выход триггера `Q` и есть сигнал управления замком `locked`. Вы только что научились строить асинхронные триггеры, они очень полезны в устройствах ввода-вывода с нетривиальными протоколами.

Наконец, добавьте конвертор фронта в импульс в сеть сброса сдвигового регистра. Это должно сбрасывать 4-значный дисплей в момент запирания двери (на восходящем фронте сигнала `locked`). Теперь вы можете протестировать все, кроме компаратора.

4. Удалите контакт, который вы использовали вместо результата сравнения, и постройте полноценный компаратор. Он должен включать 16-битный регистр PIN, управляемый сигналом `clk` (так же, как и сдвиговый регистр) и получающий данные из конкатенации значений индикатора `D0-D3`. Защелкивание ввода должно контролироваться распознавателем звездочки и должно происходить только при открытой двери в режиме 0 (вы можете заметить, что нужный сигнал уже присутствует в одном из логических произведений, которые вы включили в ваш конечный автомат ранее, так что вы можете просто взять сигнал оттуда). Компаратор (вы можете использовать библиотечный компонент, вместо того, чтобы строить собственный) присоединяется между выходом регистра PIN и конкатенацией `D0-D3`. Присоедините выход компаратора к проводу, который раньше управлялся вашим отладочным контактом.

Ваш дизайн готов. Тщательно протестируйте его, и, если он работает, отправьте его на проверку роботом.

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 сде-

лать). 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.