NSU

“The Game of Noughts and Crosses”

Nurdolotova Sabina

Derunets Roman

Contents

[Contents 2](#_Toc72575695)

[**Brief Overview** 3](#_Toc72575696)

[**Hardware** 3](#_Toc72575697)

[TTTC 3](#_Toc72575698)

[SDR 4](#_Toc72575699)

[BPC 6](#_Toc72575700)

[GSDD 7](#_Toc72575701)

[**Software and main subroutines** 8](#_Toc72575702)

[playerturn 8](#_Toc72575703)

[drawcounter 8](#_Toc72575704)

[wincheck 9](#_Toc72575705)

[lose/win/draw 9](#_Toc72575706)

[computerturn 10](#_Toc72575707)

## **Brief Overview**

Из трех представленных проектов мы решили выбрать проект “A” – игру “Крестики-нолики”, так как посчитали это довольно занимательным. Чтобы создать эту игру, нам нужно было использовать смесь Logisim и Cdm-8.

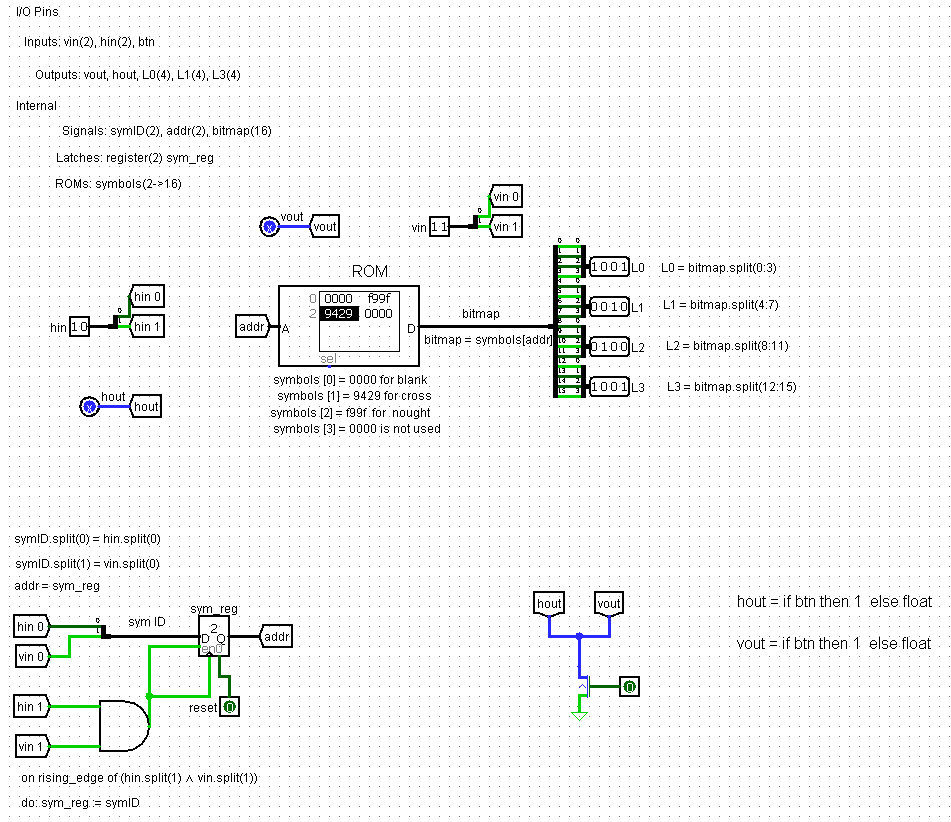
Наша группа столкнулась с множеством проблем. Поскольку мы не привыкли использовать Logisim и Cdm-8 в той сложности, в которой мы должны были, это означало, что нам нужно было научиться использовать как программное обеспечение (software), так и аппаратное (hardware) на более продвинутом уровне, чтобы использовать их в полной мере.

Основная задача — это разработать геймпад 3х3, подключить его к Сdm-8 и написать программу на языке ассемблера для управления игрой. При этом каждая из 9 ячеек геймпада управляет светодиодом размером 4х4 пикселей, который может отображать крестик, нолик или пустое место. Также каждая ячейка имеет кнопку ввода, которую использует игрок для своего хода. Геймпад имеет чипы, которые можно использовать для подключения к шине ввода-вывода Cdm-8 процессора для контроля игры.

## **Hardware**

TTTC

TTTC (Tic-Tac-Toe Chip) – это микросхема, контролирующая матричный дисплей ячейки и кнопку для ввода.

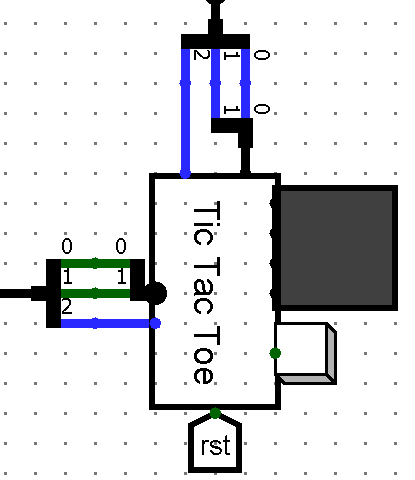


Микросхема состоит из 2-битных входных контактов vin и hin, входного контакта btn, однобитных выходных контактов vout и hout, 4-битных выходных контактов L0, L1, L2, L3 и 1-битного контакта rst (для обнуления). Также есть ПЗУ с рязрядностью адреса – 2, и разрядностью данных – 16.

ПЗУ содержит четыре значения:

symbols [0] = 0000 для пустого места, symbols [1] = f99f для нолика, symbols [2] = 9429 для крестика, symbols [3] = 0000 (не используется).

Нулевые биты vin и hin представляют идентификатор отображаемого символа (sym ID). Он фиксируется в регистре sym\_reg, когда первые биты vin и hin равны 1. А адрес в ПЗУ — это идентификатор символа для отображения. Изображение считывается из ПЗУ с использованием этого адреса (нулевые биты vin и hin). Таким образом при sym ID = 00 – на дисплее отображается пустое место, при sym ID = 01 – нолик, при sym ID = 10 – крестик.

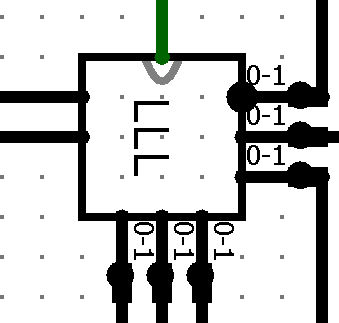
 На северной части чипа 1-битный контакт hout и 2-битный hin. Все эти три бита подключены к горизонтальной шине ячейки (hout к проводу 2 и hin к проводам 0 и 1). Контакт hout переносит выходные данные с чипа TTTC, а hin переносит входные данные к чипу TTTC.

На западной стороне 2-битный vin и 1-битный vout для подключения к соответствующей вертикальной шине.

Светодиодный матричный дисплей 4x4 подключен непосредственно к выходным контактам TTTС L0, L1, L2, L3 (по 4 бита каждый) на его восточной стороне. Они управляют строками 0, 1, 2 и 3 матрицы соответственно, чтобы отображать ноль, крест или пробел.

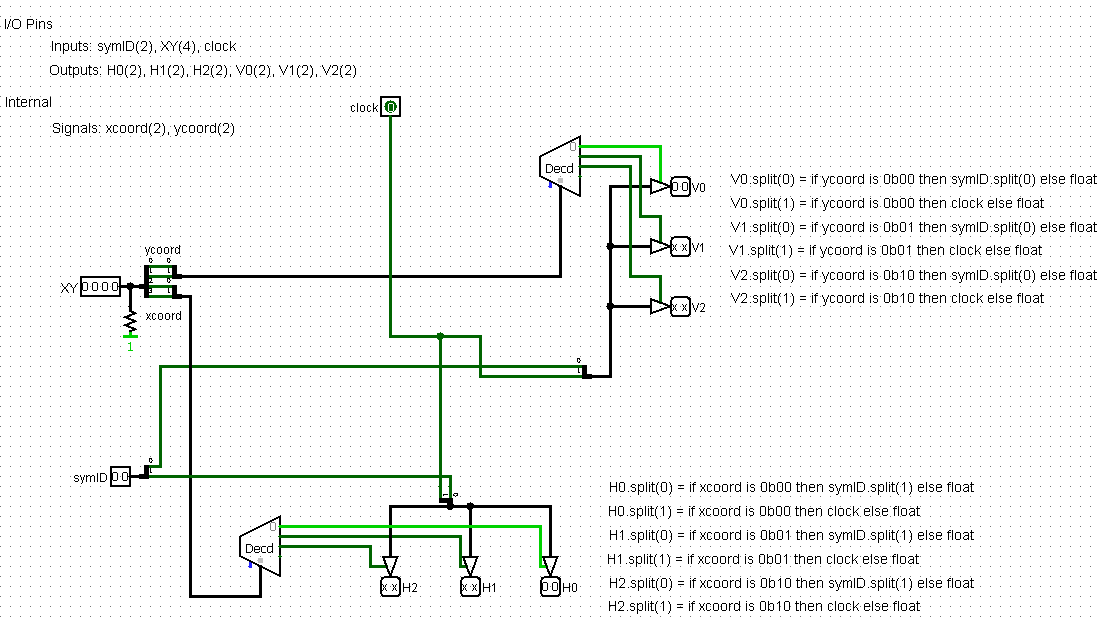
Также предусмотрена кнопка (btn), чтобы человек играл крестиком в ячейке. Она подключена к 1-битному входному контакту на восточной стороне TTTC. На южной стороне тоннель rst для обнуления ячейки.

SDR

Чип SDR (обозначенный LLL) используется для передачи идентификатора символа, который будет отображаться на игровой панели, на правильный чип TTTC, управляющий данной ячейкой. Идентификатор символа и координата xy ячейки поступают от процессора. Этот чип имеет шесть 2-битных выходных контактов H0, H1, H2 и V0, V1, V2, каждый из которых подключен к битам 0 и 1 одной из шин.

Идентификатор символа и адрес ячейки принимаются чипом SDR на западной стороне через 2-битный контакт symID и 4-битный контакт XY.

Когда процессору необходимо записать символ (нолик, крестик или пустое место) в ячейку с координатами xxyy, он должен отправить как 2-битный идентификатор символа, так и адрес ячейки xxyy на микросхему SDR. Адрес ячейки декодируется чипом SDR и используется для выбора правильной пары горизонтальной и вертикальной шин для ячейки, при этом идентификатор символа направляется на правильный чип TTTC.



XY содержит координаты x, y используемого чипа TTTC.

SDR декодирует адрес XY следующим образом:

Сначала разделяет адрес на две 2-битные координаты (xcoord и ycoord). Затем координата направляется к декодеру, который в зависимости от значения координаты, определяет на каком из выходов декодера будет 1. Далее данные с декодера идут к трем управляющим буферам, которые, в свою очередь, принимают 2-битное значение (нулевой бит зависит от symID, а первый – от импульса clock) и направляют это значение к выходным контактам (V0, V1, V2 для ycoord и H0, H1, H2 для xcoord). Нулевой бит вертикального выхода зависит от нулевого бита symID, а первый бит зависит от clock. Нулевой бит горизонтального выхода зависит от первого бита symID, а первый бит зависит от clock.

symID - идентификатор отображаемого символа. Координаты XY используются для направления комбинации symID и тактового импульса на правильный чип TTTC. Пять чипов TTTC получат тактовый импульс, но только один получит одновременные тактовые импульсы как на hin, так и на vin. Это приведет к тому, что он зафиксирует идентификатор отображаемого символа.

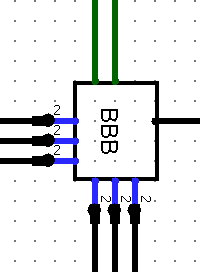
*Как правильный идентификатор символа попадает в правильный чип TTTC?*

Бит 1 идентификатора символа направлен на бит 0 выбранной горизонтальной шины, а бит 0 идентификатора символа направлен на бит 0 выбранной вертикальной шины. Таким образом, нулевые биты двух шин несут идентификатор отображаемого символа.

Чип TTTC нуждается в триггере для фиксации идентификатора символа во внутреннем 2-битном регистре. Триггер предоставляется на бит-1 соответствующей пары горизонтальных и вертикальных шин одновременно, каждая из которых поднимается, а затем сбрасывается после утверждения данных. Это фиксирует данные в регистре. Только одна ячейка на перекладине будет иметь бит-1 как горизонтальной, так и вертикальной шин, поднятых одновременно, поэтому только одна ячейка будет фиксировать данные несмотря на то, что каждая шина подключена к 3 ячейкам.

BPC

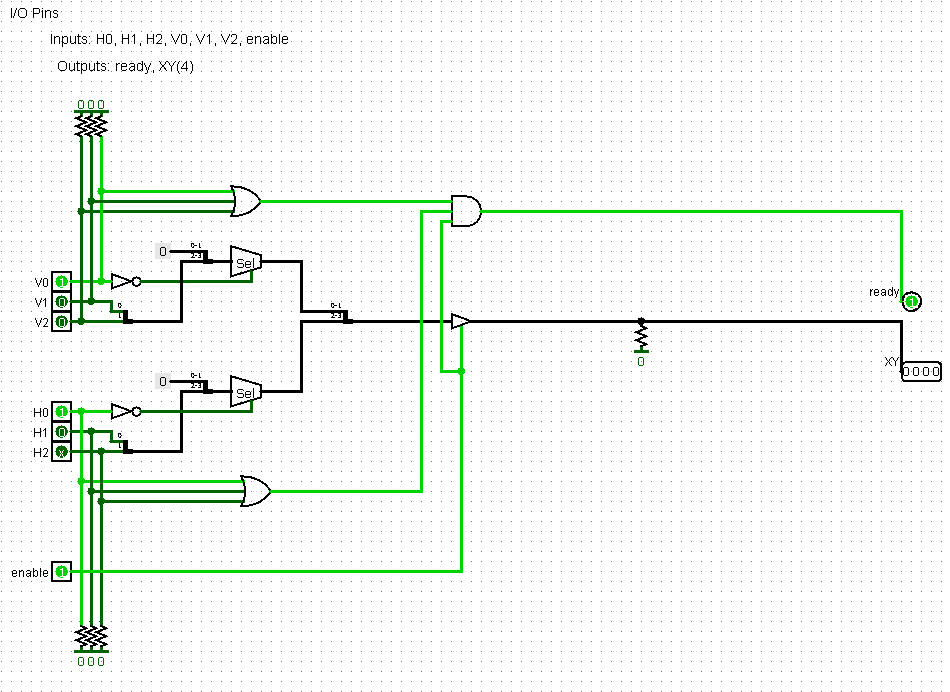
Задача BPC – захватить и удерживать адрес ячейки, в которой произошло последнее нажатие кнопки. Затем адрес передается процессору.

Чип BPC фиксирует адрес сетки ячейки, содержащей последнюю нажатую кнопку. Каждый из контактов H0, H1, H2 и V0, V1, V2 подключен к биту 2 на одной из шести шин. Чип BPC не подключен к битам 0 и 1.

BPC имеет входные контакты H0, H1, H2, V0, V1, V2 и enable и выходные ready и XY. Также есть согласующие регистры, которые в случае плавающих значений поменяют их на ноль. С помощью селекторов битов преобразуем входные контакты, т.е. если V0 = 1, тогда координата yy = 00, V1 = 1, тогда координата yy = 01, V2 = 1, тогда координата yy = 10. Тоже самое с горизонтальными шинами: если H0 = 1, тогда координата xx = 00, H1 = 1, тогда координата xx = 01, H2 = 1, тогда координата xx = 10.

Затем, если кнопка была нажата, координаты xx и yy направляются к выходному контакту XY.Таким образом, ячейка 1 в геймпаде имеет адрес 0 (0000), ячейка 2 – 1 (0001), 3 – 2 (0010), 4 – 4 (0100), 5 – 5 (0101), 6 – 6 (0110), 7 – 8 (1000), 8 – 9 (1001), 9 – a = 10 (1010)

Когда кнопка нажата, выходной контакт ready равен 1 (только в случае, когда хотя бы один из контактов H0, H1, H2 и хотя бы один из V0, V1, V2 будут равны 1)



GSDD

GSDD (Game State Display Driver) – чип, определяющий победу, поражение или ничью, когда игра закончилась, и управляет соответствующими светодиодами.

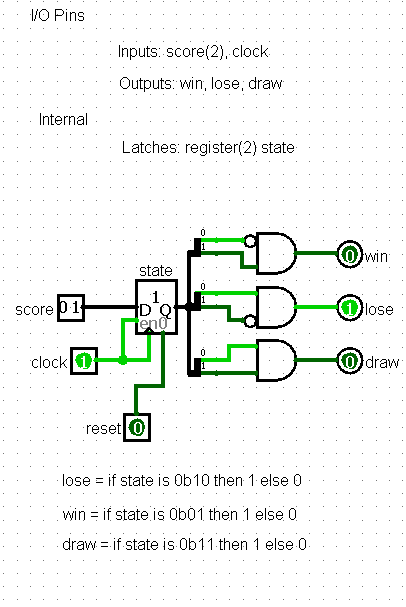
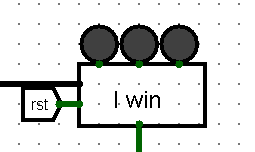
GSDD принимает 6 и 7 биты из I/Odat. Изначально оба бита устанавливаются нулями. В последний ход процессор выводит 2-битный сигнал на эти два провода, который определяет, какой из светодиодов должен быть зажжен.

GSDD имеет 2-битный входной контакт score и 1-битный clock, три выходных контакта: win, lose, draw, 2-битный регистр state и reset (для обнуления).

Когда score = 01, статус игры win (для компьютерного игрока) и загорается красный светодиод (поражение для человека)

Когда score = 10, статус игры lose и загорается зеленый светодиод

Когда score = 11, статус игры draw и загорается желтый светодиод

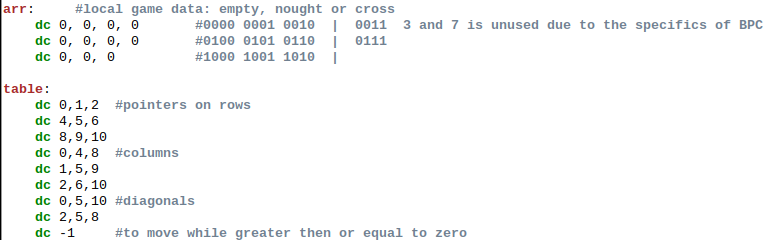
 

На западной стороне вход score (состояние игры)

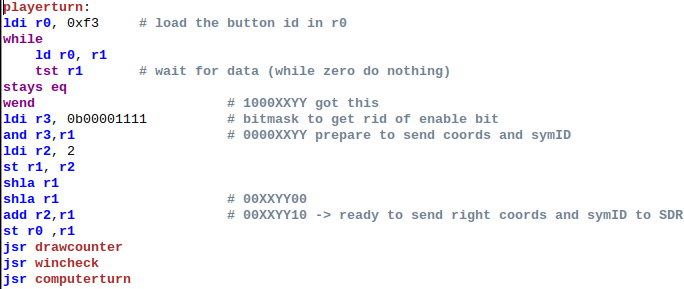
На южной – clock

На северной - светодиоды

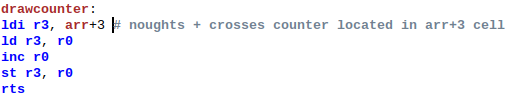
## **Software and main subroutines**

В памяти выделяется arr под локальные данные игры, чтобы хранить информацию, где находятся крестики и нолики,  
и table под указатели на строки, столбцы и диагонали, чтобы можно было удобно по ним перемещаться циклом

playerturn

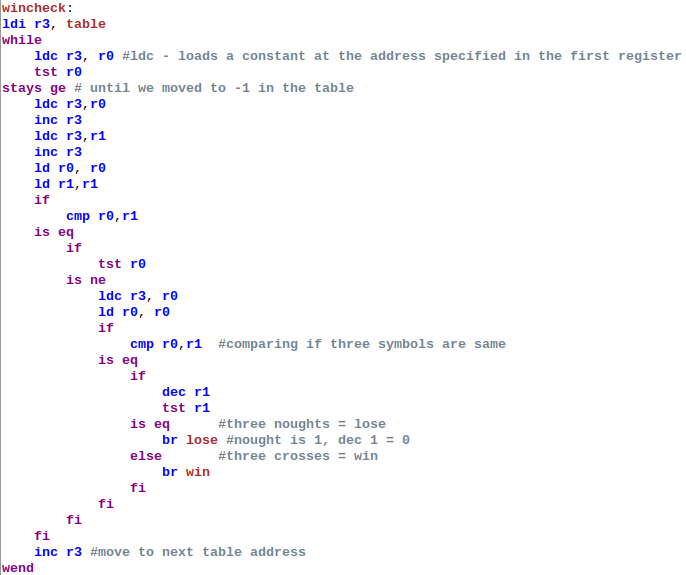
Считывает данные из схемы, которые передаёт BPC при нажатии кнопки, сохраняет локально значение крестика и переводит данные в формат, который принимает SDR (первые 2 бита -symID, следующие 4 — координаты, биты под GSDD пока нулевые)

drawcounter



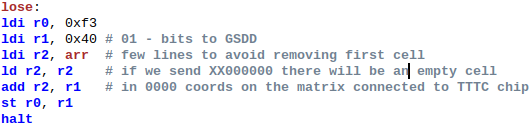
счётчик поставленных на игровое поле объектов, чтобы упростить проверку условия ничьей. Хранится по адресу 0x03.

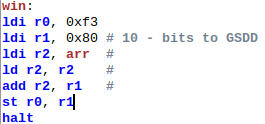
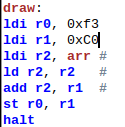
wincheck



Проходит по таблице и проверяет наличие трёх крестиков или ноликов в строке, столбце или на диагонали, переходя при выполнении условий к lose или win. Если указанные условия не выполнены, то смотрит на счётчик объектов, если он равен 9, то переходит к draw

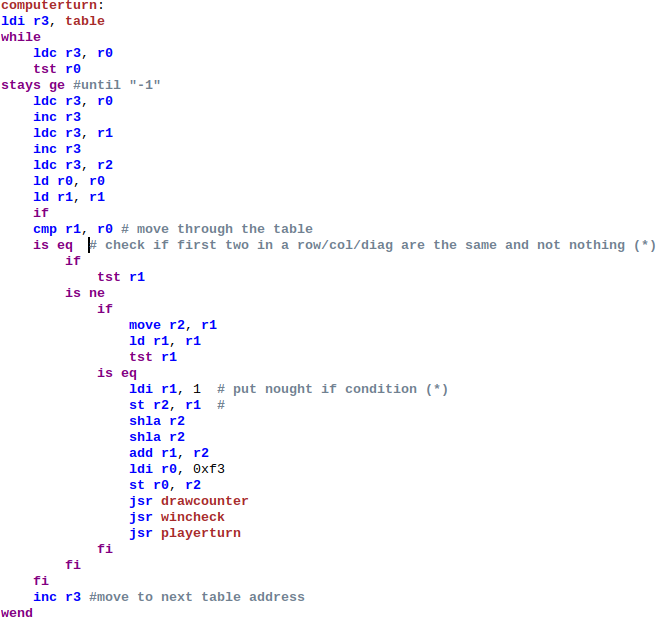
lose/win/draw



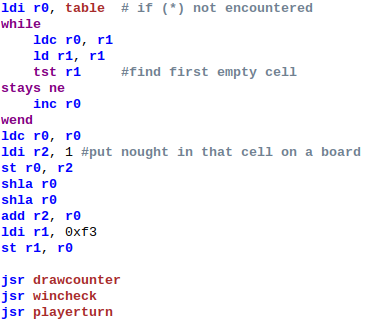


устанавливают 6 и7 биты, которые обрабатывает GSDD чип, чтобы загорелся нужный светодиод, вызывает команду остановки halt

computerturn



Алгоритм компьютерного игрока при движении проверяет, есть ли два одинаковых объекта в первых двух элементах строки/столбца/диагонали, если такое встречается, то третьим элементом ставится нолик.



Если нет, то нолик ставится в первую свободную ячейку.