Створення керуючого автомата на основі жорсткої логіки в середовищі ISIS Proteus VSM

1.Теоретичне підґрунтя

Для організації системи контролю механічного цеху, використовуємо запірний пристрій, який застосовується для замикання дверей (дверний замок) з метою запобігання викраденню, він представлений на рисунку 1.1.

Рисунок 1.1 Пульт керування дверним замком

Для керування роботою даного дверного замка (див. рис.1.1) використовуємо керуючий автомат на жорсткій логіці, тобто з неможливістю подальшого його перепрограмування.

У цій роботі керуючий автомат створюється на основі побудови графа станів цього автомата (рисунок 1.2).

Граф станів – це чітка хронологічна послідовність виконання необхідних дій покроково (послідовно), тобто в кінці графу станів отримується відповідний необхідний результат (двері відчиняться), але з умовою правильного вводу вхідних аргументів (в даному випадку це послідовний і правильний ввід

кнопок на пульті керування для відчинення дверей (рис1.1)), також на графі станів графічно представлено поведінку роботи керуючого автомата у випадку неправильного вводу вхідної інформації, але в цьому разі необхідний результат не забезпечиться (двері залишатимуться зачиненими) і виконається відповідна блокуюча дія, або тривожний сигнал, що дасть інформацію про спробу зламу дверей (тобто не був введений правильний вхідний код, який відомий тільки тим особам, що пов'язані з роботою даного цеху).

Також важливо те, що перехід від першого стану до останнього стану, тобто до повністю виконаної функції (коли двері відчиняться) буде здійснюватися при послідовному вводі вхідних аргументів (тобто натиснувши декілька кнопок одночасно, в тому числі і коли серед них будуть ті кнопки, що треба, то все одно буде сприйматися автоматом, як неправильний вхідний сигнал і переходу до наступного стану (рис.1.2) не відбудеться). Отже двері відкриються тільки при натисненні першої правильної кнопки і витисненні її, а вже потім натисненні другої правильної кнопки і її витисненні).

Граф станів роботи даного дверного замка зображений на рисунку 1.2.

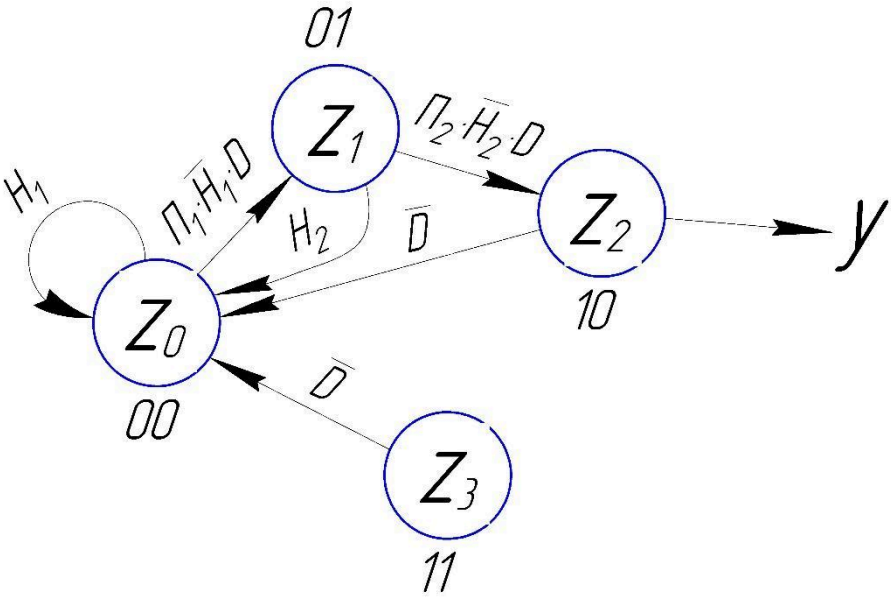


Рисунок 1.2 Граф станів

$П_1$ – правильний перший аргумент (натиснута потрібна перша кнопка),

$П_2$ – правильний другий аргумент (натиснута потрібна друга кнопка),

H_1 – неправильний перший аргумент (якась інша перша кнопка, яка є неправильна, тому вона повинна бути ненатиснута, отже інвертована),

H_2 – неправильний другий аргумент (якась інша друга кнопка, яка є неправильна, тому вона повинна бути ненатиснута, отже інвертована),

D – аргумент, що вказує на стан дверей (тобто, двері повинні бути замкнутими).

$\overline{D}$ *– аргумент, що вказує на стан дверей (тобто, двері повинні бути відчиненими).*

Розглянувши граф станів (рис.1.2), можна детальніше його аналізувати:

Отже, Z_0 – означає початковий стан автомата, (тобто, ніхто нічого на пульті керування ще не вводив), початковий стан описується двійковим числом 00, про що більш детально буде вказано дещо пізніше.

Із стану Z_0 до стану Z_1 можна перейти при необхідних вхідних аргументах: *D* (тобто, необхідно щоб двері були замкнуті, бо тоді нема сенсу їх відкривати), $\overline{H}_1$ (тобто, необхідно щоб не було натиснуто неправильну першу кнопку, або декілька кнопок одночасно, бо, як видно з рисунка 1.2, при натисненні неправильної першої кнопки H_1 переходу до наступного стану не буде, а залишимось знову на Z_0), $П_1$ (тобто, необхідно щоб була натиснута перша правильна кнопка і зразу витиснута).

Отже при вводі цих трьох аргументів і переході до стану Z_1 далі необхідно перейти до стану Z_2 і вже вводити наступні вхідні аргументи (рис.1.2): *D* (тобто, необхідно щоб двері і надалі були замкнуті), $\overline{H}_2$ (тобто, необхідно щоб не було натиснуто неправильну другу кнопку, або декілька кнопок одночасно, бо, як видно з рисунка 1.2, при натисненні неправильної другої кнопки H_2 переходу до наступного стану не буде, а відбудеться повернення в початковий стан Z_0), $П_2$ (тобто, необхідно щоб була натиснута друга правильна кнопка).

Отже при переході в стан Z_2 виконується функція Y (двері відкриються).

Ще передбачено той випадок, коли двері хтось відчинить зсередини, тоді не має сенсу їх далі відкривати (коли вони вже відкриті), то на цей випадок передбачений спеціальний стан Z_3, тобто коли $\overline{D}$ (двері хтось відчинив зсередини), тоді стан апарату повертається із стану Z_3 до Z_0 (рис.1.2).

Як видно з рисунка 1.2, для виконання функції Y (тобто, для того, щоб двері відчинилися), треба пройти три стани (Z_0 , Z_1 , Z_2), і є ще один додатковий стан Z_3 на випадок, коли хтось двері відчинить зсередини. То загалом є чотири стани, які описуються кодом стану (двійковим числом), тобто так як є тільки чотири стани, то використовується двохбітний код, так, як його достатньо щоб описати четвертий стан який буде 11. Код першого стану Z_0 буде 00, другого стану Z_1 буде 01, третього стану Z_2 буде 10 і четвертого стану Z_3 буде 11. Оскільки кожен стан описується двохбітним кодом, то для запису кожного коду стану (пам’яті скінченого автомата) необхідно використати два RS тригери.

2. Хід виконання роботи

2.1 Вибір вхідної інформації

Отже для прикладу організуємо роботу керуючого апарата, таким чином, щоб двері відчинились при натисненні кнопки 1 і зразу її витисненні, а вже потім натисненні кнопки 3, так, як показано на (рисунку 2.1).

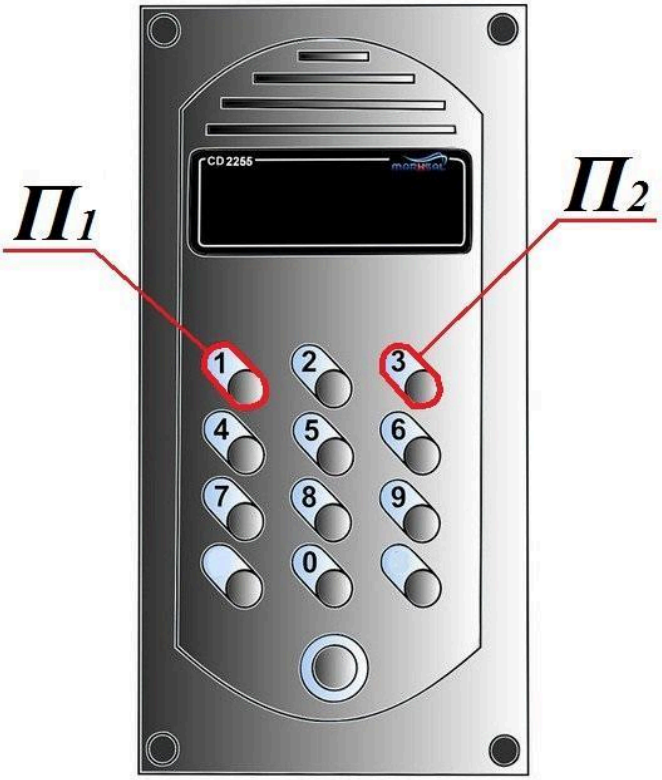


Рисунок 2.1 Послідовний ввід кнопок (спочатку натиснемо кнопку 1, наступну кнопку 3)

Так, як в нас тільки два вхідних аргументи (дві кнопки), то з графу станів (рис. 1.2), видно що для переходу з початкового стану Z_0 в стан Z_1, необхідно натиснути кнопку 1 і зразу її витиснути, так ми перейдемо в стан Z_1, а вже потім натиснувши кнопку 3, ми перейдемо з стану Z_1 до стану Z_2 і в результаті двері відчиняться. Як було зазначено раніше ще передбачено той випадок, коли хтось відкрив двері з середини (стан Z_3), тоді повернемося знову в стан Z_0.

Так як використовуються тільки чотири стани (кожен стан має двохбітний двійковий код від 00 до 11), то для їх запису достатньо використати два RS тригери. І перехід до наступного стану залежить від вмісту коду попереднього стану і натисненні правильної кнопки.

Дані вмісту тригерів для всіх можливих станів заносимо в таблицю станів (табл. 1.1).

Таблиця 1.1

Стан	Код стану	Q_2 до	Q_1 до	$П_1$	H_1	$П_2$	H_2	D	S_2	R_2	S_1	R_1	Q_2 після	Q_1 після	Код стану	Стан
Z_0	00	0	0		1				0	0	0	0	0	0	00	Z_0
Z_0	00	0	0	1	0	×	×	1	0	0	1	0	0	1	01	Z_1
Z_1	01	0	1	×	×	1	0	1	1	0	0	1	1	0	10	Z_2
Z_1	01	0	1	×	×	×	1	×	0	1	0	1	0	0	00	Z_0
Z_2	10	1	0	×	×	×	×	0	0	1	0	1	0	0	00	Z_0
Z_3	11	1	1	×	×	×	×	0	0	1	0	1	0	0	00	Z_0

Як видно з таблиці, перший рядок – це неможливість переходу від початкового стану Z_0 до стану Z_1, так як $H_1 = 1$ (тобто натиснута неправильна перша кнопка), і в результаті залишаємось надалі в першому стані Z_0.

Другий рядок показує перехід від стану Z_0 до стану Z_1, при $П_1 = 1$ (тобто натисненні першої правильної кнопки), $H_1 = 0$ (при цьому не натиснута будь-яка інша кнопка) і $D = 1$ (тобто двері залишаються зачинені), стан Z_1 має код 01, тому необхідно встановити перший RS трігер (надати логічну одиницю на вхід $S_1 = 1$).

Третій рядок показує про перехід від стану Z_1 до стану Z_2, при $\Pi_2 = 1$ (тобто натисненні другої правильної кнопки), $H_2 = 0$ (при цьому не натиснута будь-яка інша кнопка) і $D = 1$ (тобто двері залишаються зачинені), стан Z_2 має код 10, тому необхідно встановити другий RS трігер (надати логічну одиницю на вхід $S_2 = 1$) і онулити перший RS трігер (надати логічний нуль на вхід $R_1 = 1$).

Четвертий рядок показує про неможливість переходу від стану Z_1 до стану Z_2 так як $H_2 = 1$ (тобто натиснута неправильна друга кнопка) і в результаті повертаємось в початковий стан Z_0, отже необхідно онулити перший RS тріггер (надати логічну одиницю на вхід $R_1 = 1$).

П'ятий рядок показує про повернення з стану Z_2 в початковий стан Z_0 при випадку, коли хтось відчинив двері з середини, тобто $D = 1$, отже необхідно онулити другий RS тріггер (надати логічну одиницю на вхід $R_2 = 1$).

Шостий рядок показує про випадок відкривання дверей з середини (стан Z_3), коли перебуваємо в будь-якому іншому стані, тоді з будь-якого стану, відбувається повернення в початковий стан Z_0, а отже необхідно онулити всі два RS трігери (подати логічну одиницю на входи $R_1 = 1$, $R_2 = 1$).

Для графічної побудови керуючого автомата дверним замком, запишемо за допомогою таблиці 1.1, його аналітичний вираз (рівняння за допомогою яких потім будемо будувати в програмі ISIS).

Отже встановлення входу ($S_1 = 1$) першого RS трігера, буде згідно таблиці 1.1, тільки в рядку №2, тому $S_1 = \bar{Q}_1 \cdot \bar{Q}_2 \cdot \quad _1 \cdot \bar{\ }_1 \cdot \quad$.

Онулення входу $(R_1 = 1)$ першого RS трігера, буде згідно таблиці 1.1, тільки в рядку №3,4,5,6, отже $R_1 = 1$ можливий в чотирьох різних рядках, тому між ними стоятиме елемент АБО (знак елемента АБО - $\vee$)

$$R_1 = Q_1 \cdot \bar{Q}_2 \cdot \quad _2 \cdot \bar{\ }_2 \cdot H \vee Q_1 \cdot \bar{Q}_2 \cdot D_2 \vee \bar{Q}_1 \cdot Q_2 \cdot \bar{D} \vee Q_1 \cdot Q_2 \cdot \bar{\ }.$$

Встановлення входу ($S_2 = 1$) другого RS трігера, буде згідно таблиці 1.1, тільки в рядку №3, тому $S_2 = Q_1 \cdot \bar{Q}_2 \cdot \quad _2 \cdot \bar{\ }_2 \cdot \quad$.

Онулення входу $(R_2 = 1)$ другого RS трігера, буде згідно таблиці 1.1, тільки в рядку №4,5,6, отже $R_2 = 1$ можливий в трьох різних рядках, тому між ними стоятиме елемент АБО

$$R_2 = Q_1 \cdot \bar{Q}_2 \cdot D_2 \vee \bar{Q}_1 \cdot Q_2 \cdot \bar{D} \vee Q_1 \cdot Q_2 \cdot \bar{\ }.$$

2.2 Організація роботи в середовищі ISIS

Для побудови симуляції пульта керування дверним замком в програмі ISIS, використовуючи інструмент “Компоненти” (в лівій верхній частині інтерфейса програми), вибираємо необхідні компоненти з папки “выбрать из библиотек” (на рисунку 2.2, показано як кнопка P).

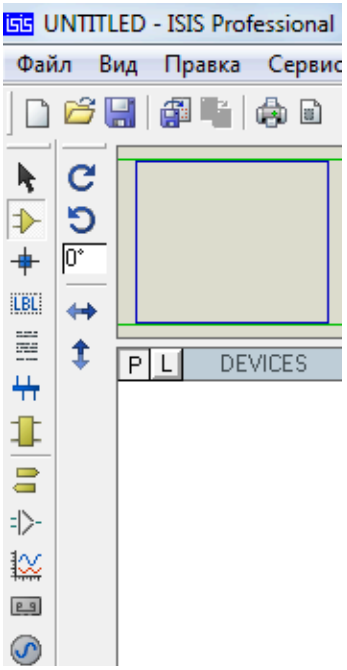


Рисунок 2.2 Представлення інструмента “Компоненти”

Вибираємо резистори з паки “Resistors → 0.6W Metal Film → MINRES10R” (див. рис 2.3)

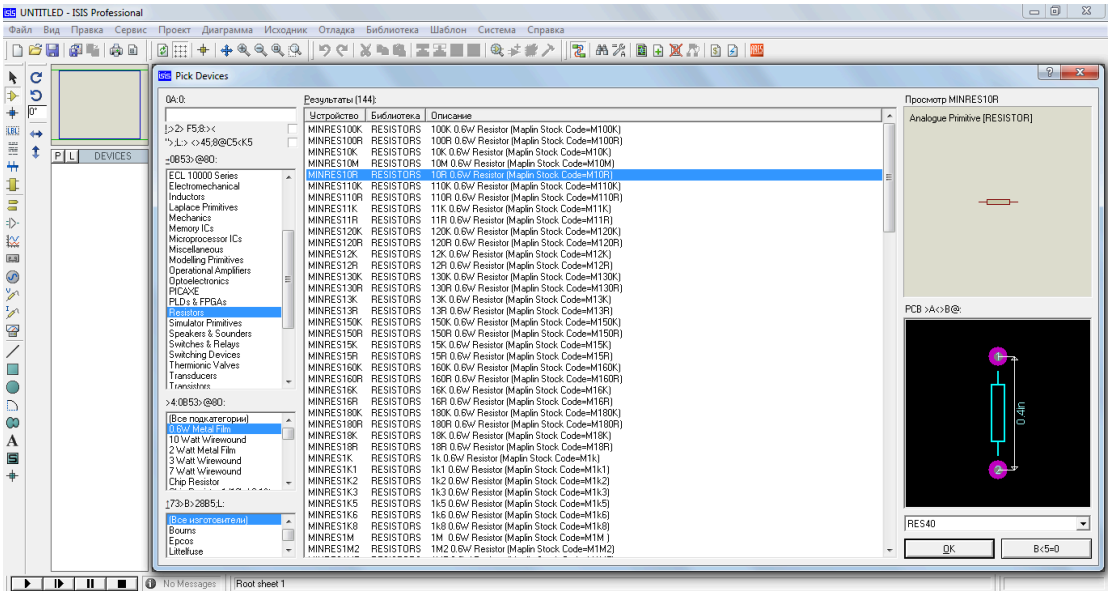


Рисунок 2.3 Вибирання резисторів в папці “ Resistors”

Вибираємо кнопки для керування керуючим автоматом з паки “ Switches & Relays → BUTTON” (див. рис 2.4)

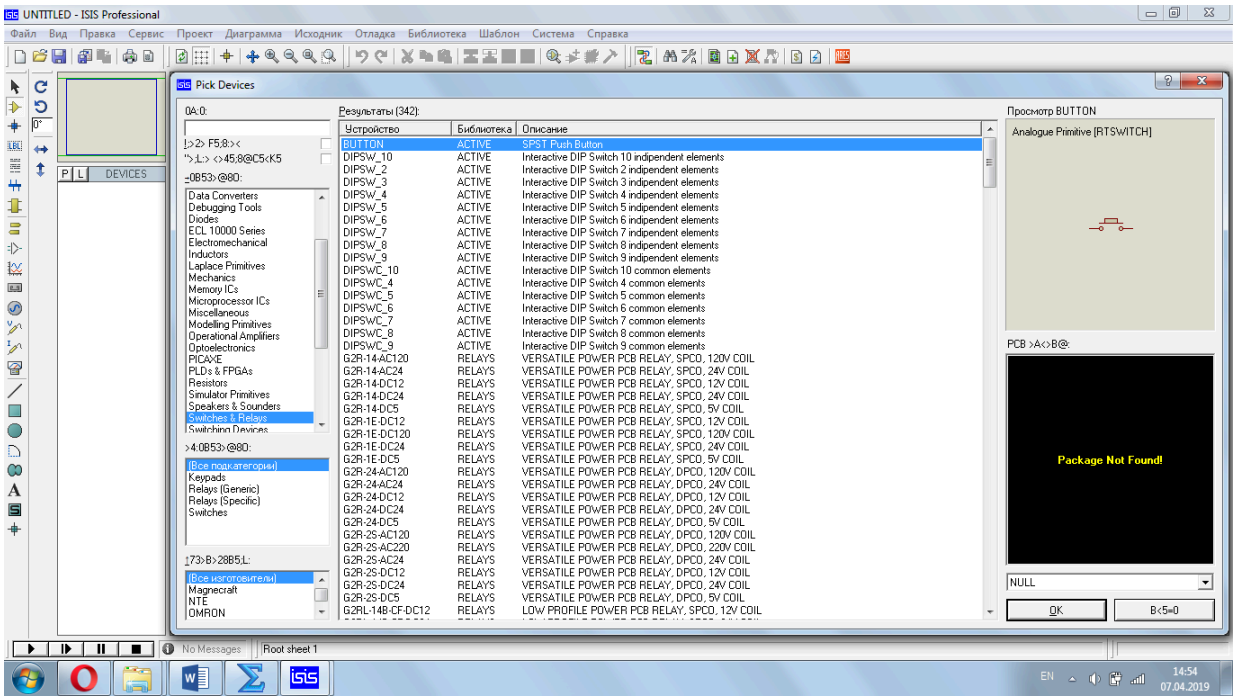
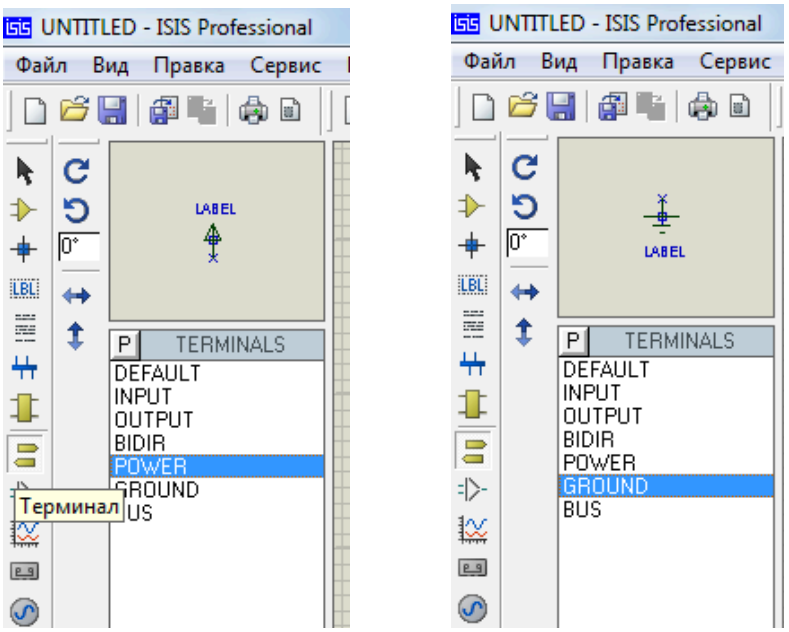


Рисунок 2.4 Вибирання кнопок керування в папці “Switches & Relays ”

Створюмо необхідну різницю потенціалів, для цього використовуємо інструмент “Терминал” (в лівій частині інтерфейса програми), вибираємо необхідні компоненти “POWER”, тобто джерело енергії (з напругою +5V) (рисунок 2.4 а) і “GROUND”, тобто заземлення (рисунок 2.4 б).



а *б*

Рисунок 2.5

На робочому полі програми створюємо симуляцію керуючого пульта (підключаємо від джерела енергії (POWER) до кнопок керування необхідну різницю потенціалів, що представлено на рисунку 2.6 (фактично рисунок 2.6 відповідає рисунку 1.1, тільки в схематичному вигляді).

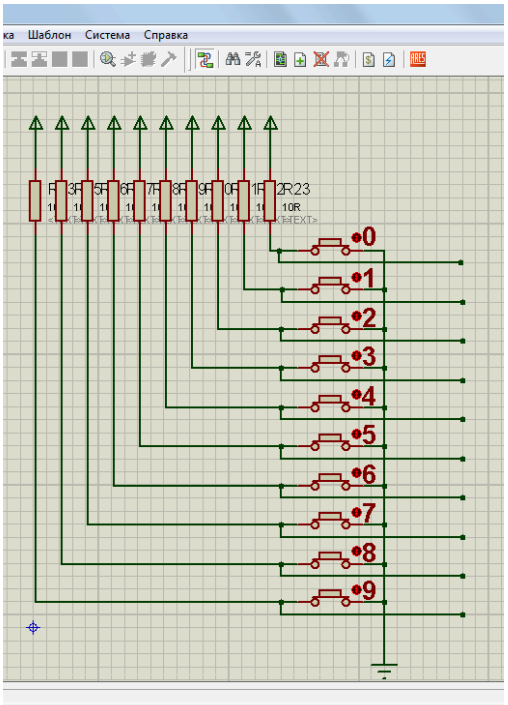


Рисунок 2.6

Складаємо комбінаційну схему, для цього вибираємо елемент H_1, (тобто першу неправильну кнопку – це буде елемент АБО, входи якого будуть всі кнопки крім першої правильної кнопки (кнопки 1), тобто вихід H_1 має бути інверсований), що представлено на рисунку 2.7

Вибираємо елемент H_2, (тобто другу неправильну кнопку – це буде елемент АБО, входи якого будуть всі кнопки крім другої правильної кнопки (кнопки 3), тобто вихід H_2), що представлено на рисунку 2.8.

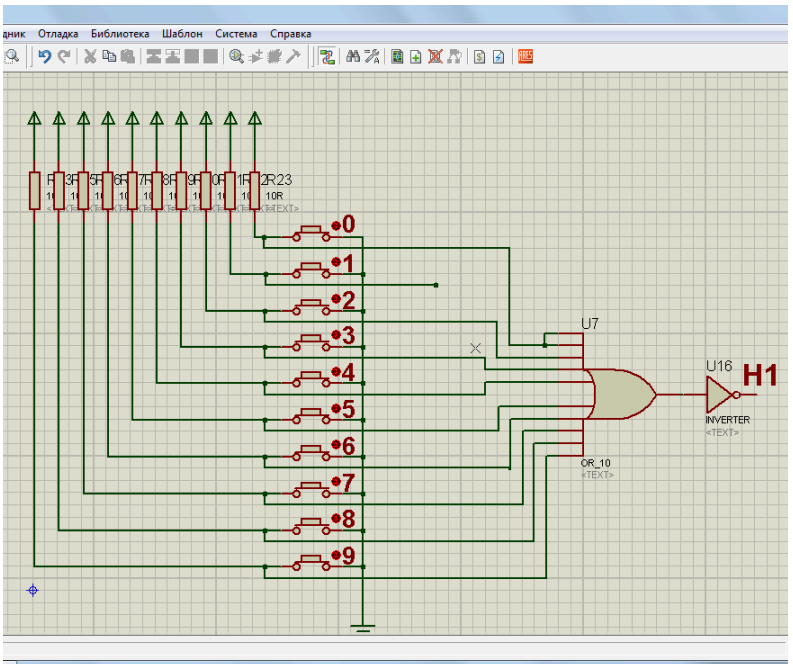


Рисунок 2.7

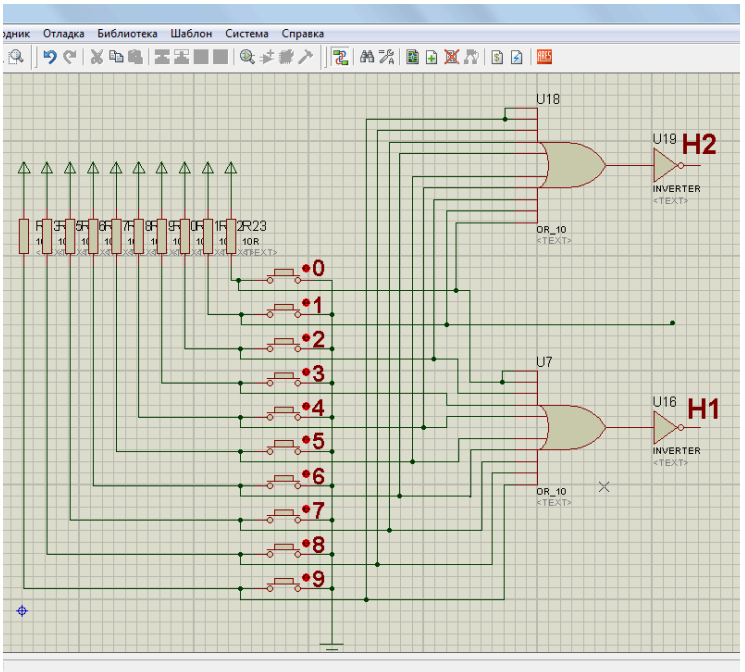


Рисунок 2.8

Для організації пам’яті скінченого автомата (як відомо раніше, в нас є чотири стани, то відповідно в двійковому коді це число 11), отже потрібно два RS трігери, які вибираємо використовуючи інструмент “Компоненти” (вибираючи з папки “Modeling Primitives → DTFF”) (рис. 2.9), та розміщуємо їх на робочому полі програми (рис.2.10)

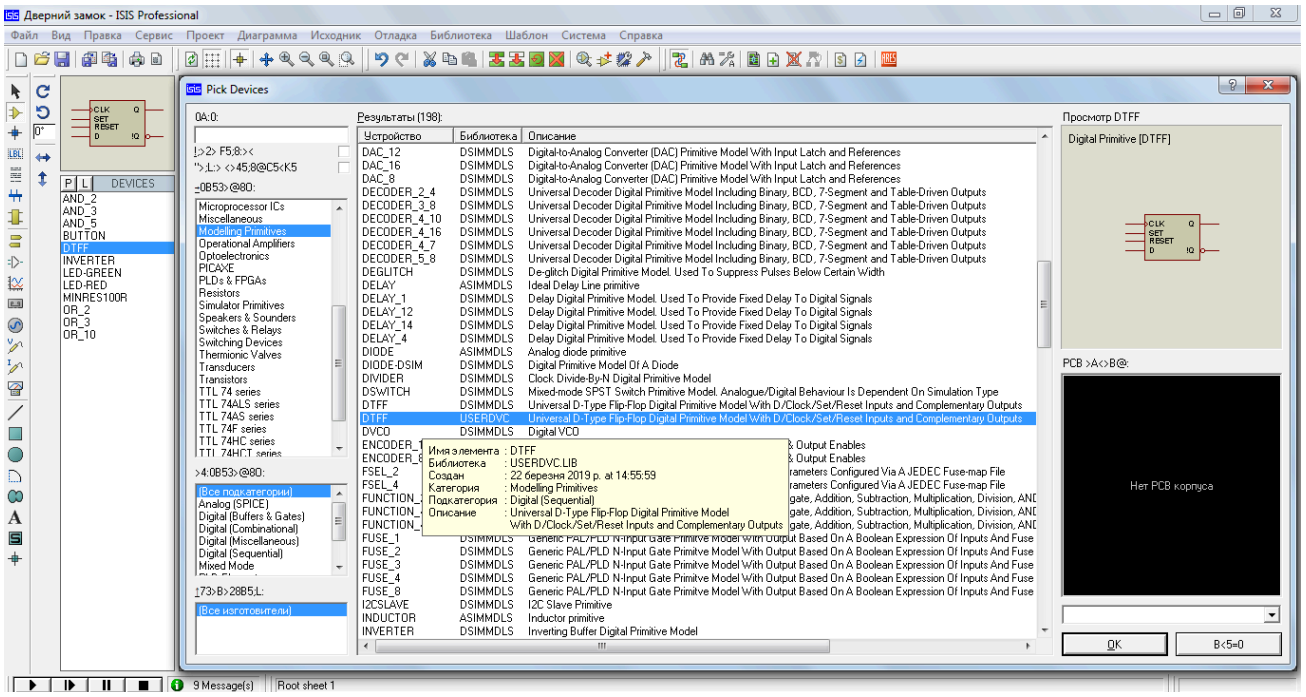


Рисунок 2.9

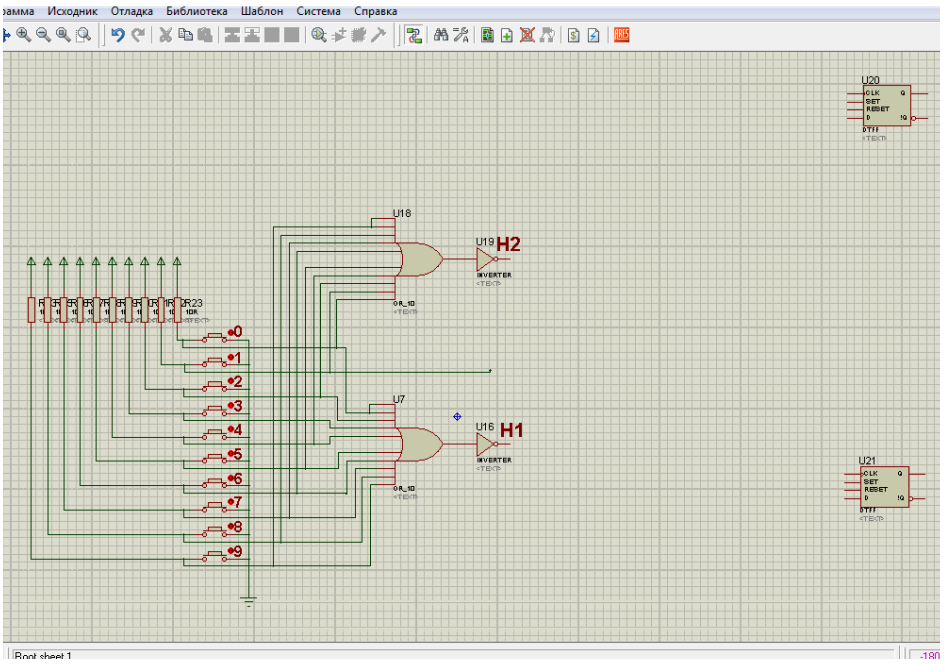


Рисунок 2.10

Для симуляції стану (Z_3), (коли можливо хтось відкрив двері з середини) використовуємо підключення окремої кнопки “ДВЕРІ” що представлено на рисунку 2.11.

Для встановлення першого RS трігера, як відомо з табл. 1.1 ($S_1 = \bar{Q}_1 \cdot \bar{Q}_2 \cdot \quad _1 \cdot \bar{\quad}_1 \cdot \quad \cdot$), використовуємо п’яти вхідний елемент “І”, який знаходиться в папці “Modeling Primitives → AND_5” (див. рис.2.12)

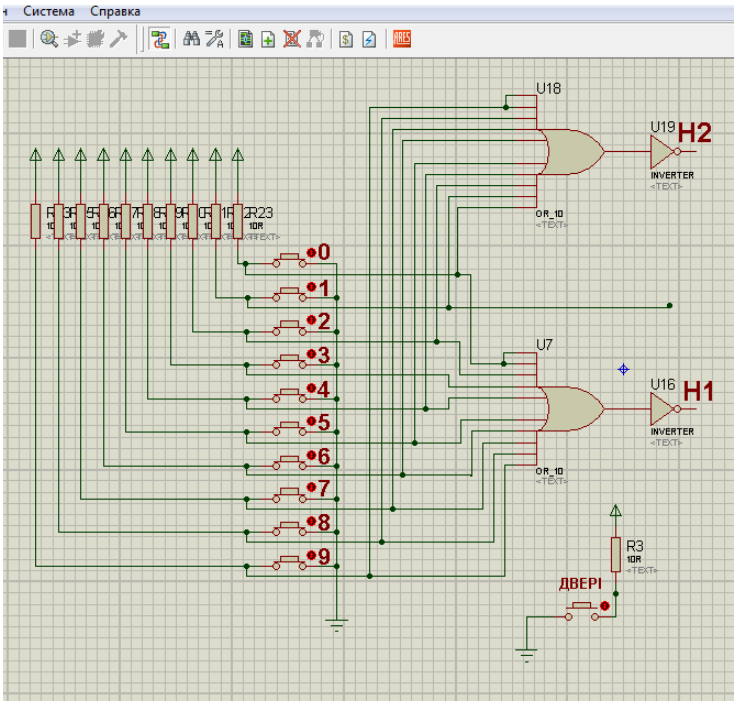


Рисунок 2.11

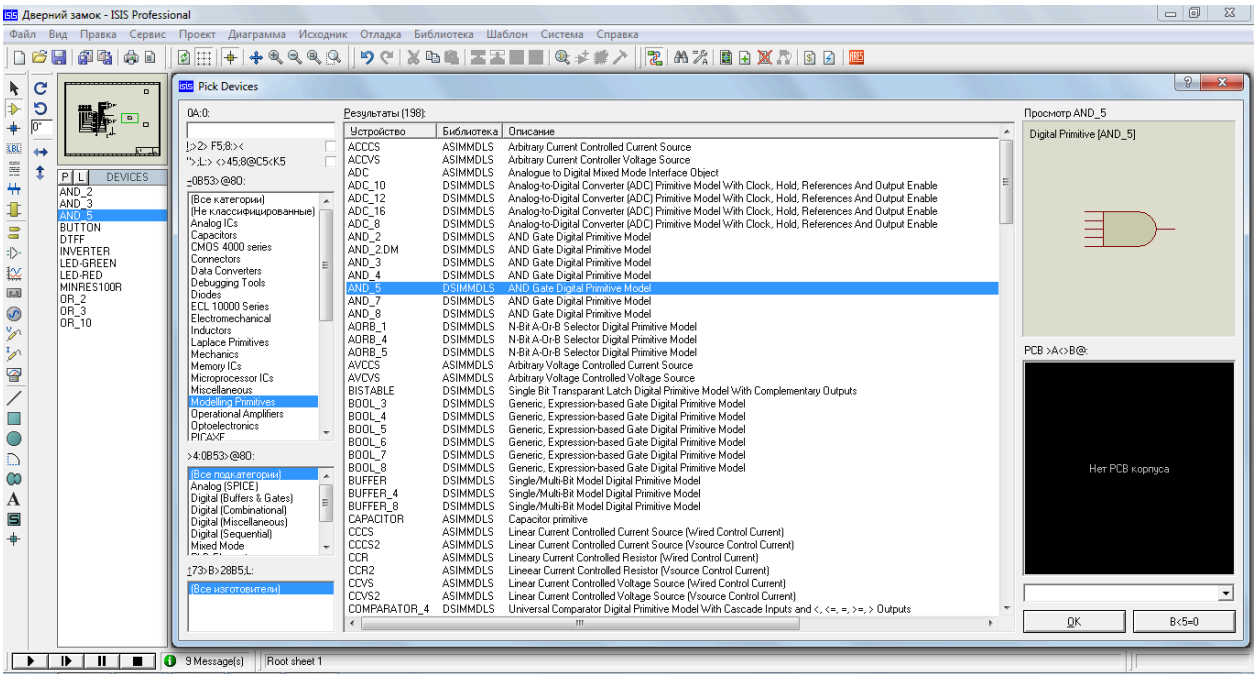


Рисунок 2.12

Підключаємо входи до елемента S_1 (“AND_5”) (рис.2.13)

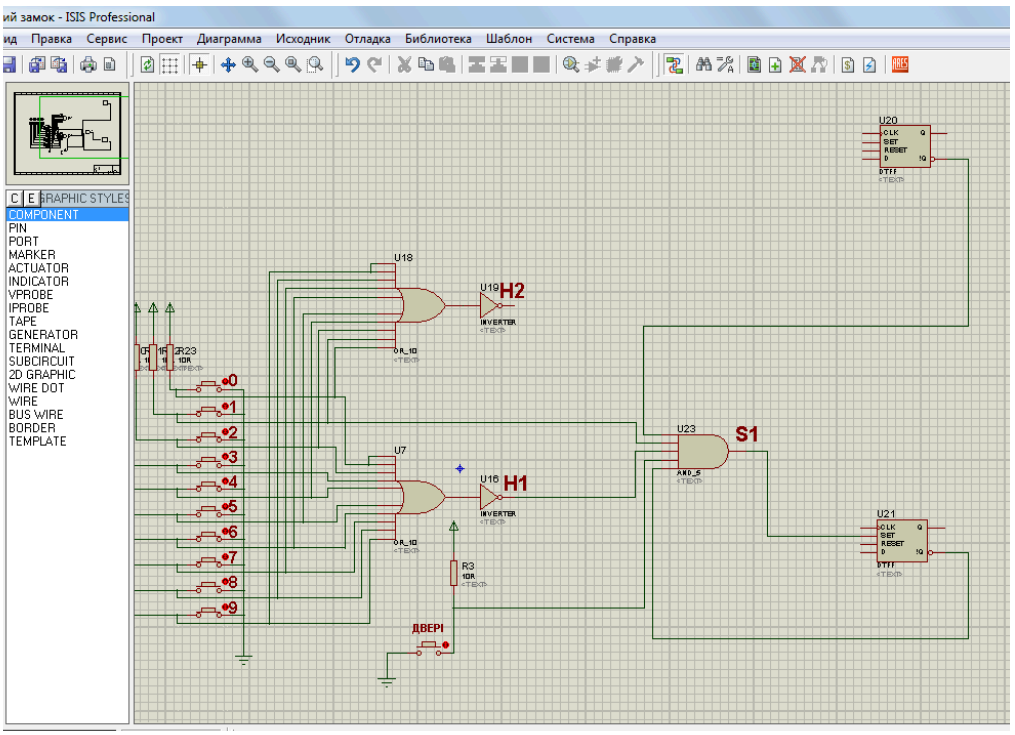


Рисунок 2.13

Для онулення першого RS трігера, як відомо з табл. 1.1 ($R_1 = H \cdot Q_1 \cdot \bar{Q}_2 \cdot \quad {}_2 \cdot \bar{\ }_2 \cdot H \vee Q_1 \cdot \bar{Q}_2 \cdot D_2 \vee \bar{Q}_1 \cdot Q_2 \cdot \bar{D} \vee Q_1 \cdot Q_2 \cdot \bar{\ } \cdot$), використовуємо

чотирьох вхідний елемент “АБО”, який знаходиться в папці “Modeling Primitives → OR_4” (див. рис.2.14), що складається з чотирьох елементів “І”:

$(Q_1 \cdot \bar{Q}_2 \cdot \Pi_2 \cdot \bar{H}_2 \cdot D)$ – перший елемент “ I_{R1} ”

$(Q_1 \cdot \bar{Q}_2 \cdot H_2)$ – другий елемент “ I_{R2} ”

$(\bar{Q}_1 \cdot Q_2 \cdot \bar{D})$ – третій елемент “ I_{R3} ”

$(Q_1 \cdot Q_2 \cdot \bar{D})$ – четвертий елемент “ I_{R4} ”

Повне представлення першого RS трігера з його входами (Reset і Setting) зображено на рисунку 2.15

Для встановлення другого RS трігера, як відомо з табл. 1.1 ($S_2 = Q_1 \cdot \bar{Q}_2 \cdot \quad {}_2 \cdot \bar{\quad}_2 \cdot \quad \cdot$), використовуємо п’яти вхідний елемент “І”, який знаходиться в папці “Modeling Primitives → AND_5” (див. рис.2.12)

Представлення підключення елемента S_2, до другого RS трігера зображено на рисунку 2.16

Для онулення другого RS трігера, як відомо з табл. 1.1 $(R_2 = Q_1 \cdot \bar{Q}_2 \cdot D_2 \vee \bar{Q}_1 \cdot Q_2 \cdot \bar{D} \vee Q_1 \cdot Q_2 \cdot \bar{\quad})$, використовуємо трьох вхідний елемент “АБО”, який знаходиться в папці “Modeling Primitives → OR_3” (див. рис.2.14), що складається з трьох елементів “І”:

$(Q_1 \cdot \bar{Q}_2 \cdot H_2)$ – перший елемент “ I_{2R1} ”

$(\bar{Q}_1 \cdot Q_2 \cdot \bar{D})$ – другий елемент “ I_{2R2} ”

$(Q_1 \cdot Q_2 \cdot \bar{D})$ – третій елемент “ I_{2R3} ”

Повне представлення підключення другого RS трігера з його входами (Reset і Setting) зображено на рисунку 2.19

Для того щоб передбачити стан Z_3 (хтось відкрив двері з середини), тобто повернутися в початковий стан Z_0 (ми маємо онулити всі RS трігери), тому підключаємо до кожного входу “RESET” з RS трігерів до інверсованого входу з кнопки “ДВЕРІ” (рис. 2.18)

Для виводу інформації застосуємо логічний елемент “І”, входами якого є вихід Q_2 і $\overline{Q}_1$, вихід елемента “І” підключимо до світлодіода зеленого кольору, який знаходиться в папці “Optoelektronics → LEDs → LED_GREEN” (див. рис. 2.20).

Повне представлення керуючого автомата дверним замком представлене на рисунку 2.21.

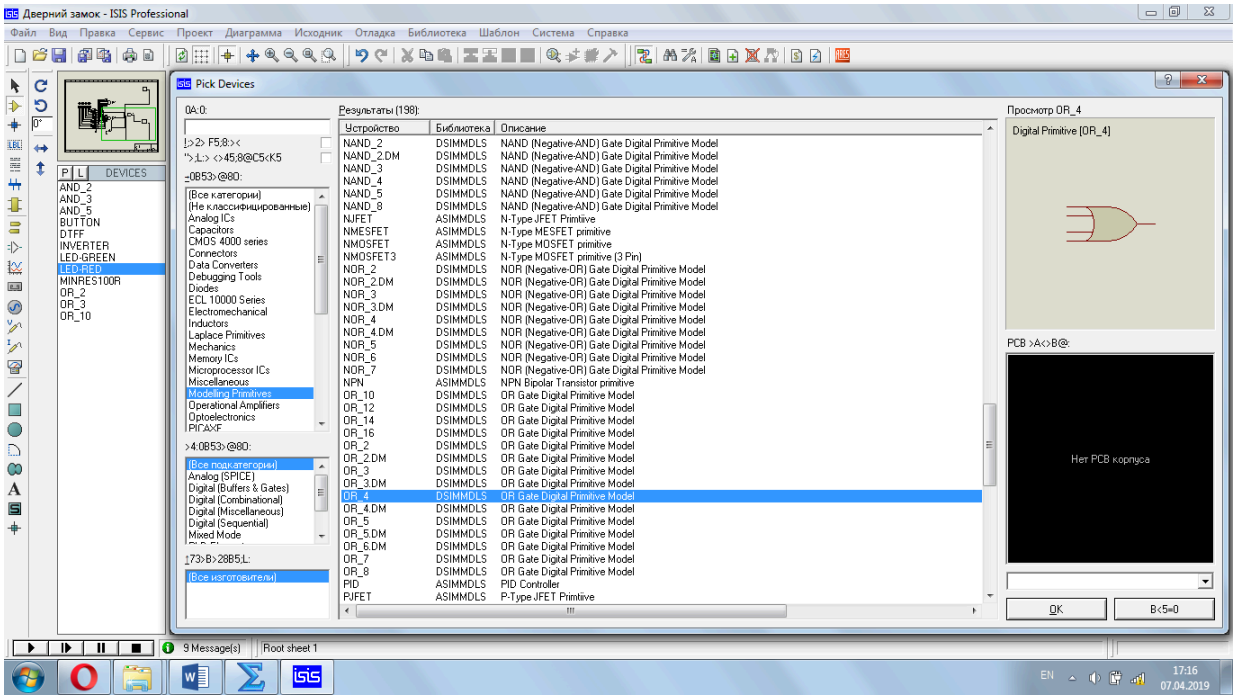


Рисунок 2.14

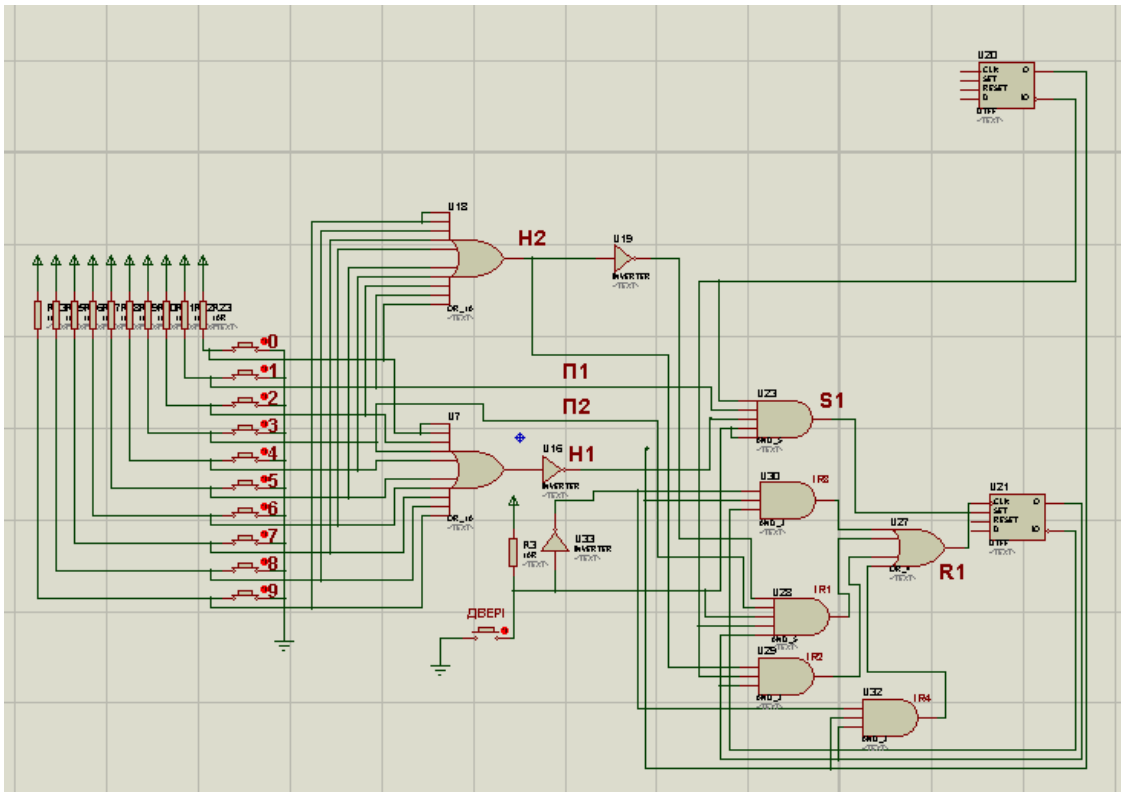


Рисунок 2.15

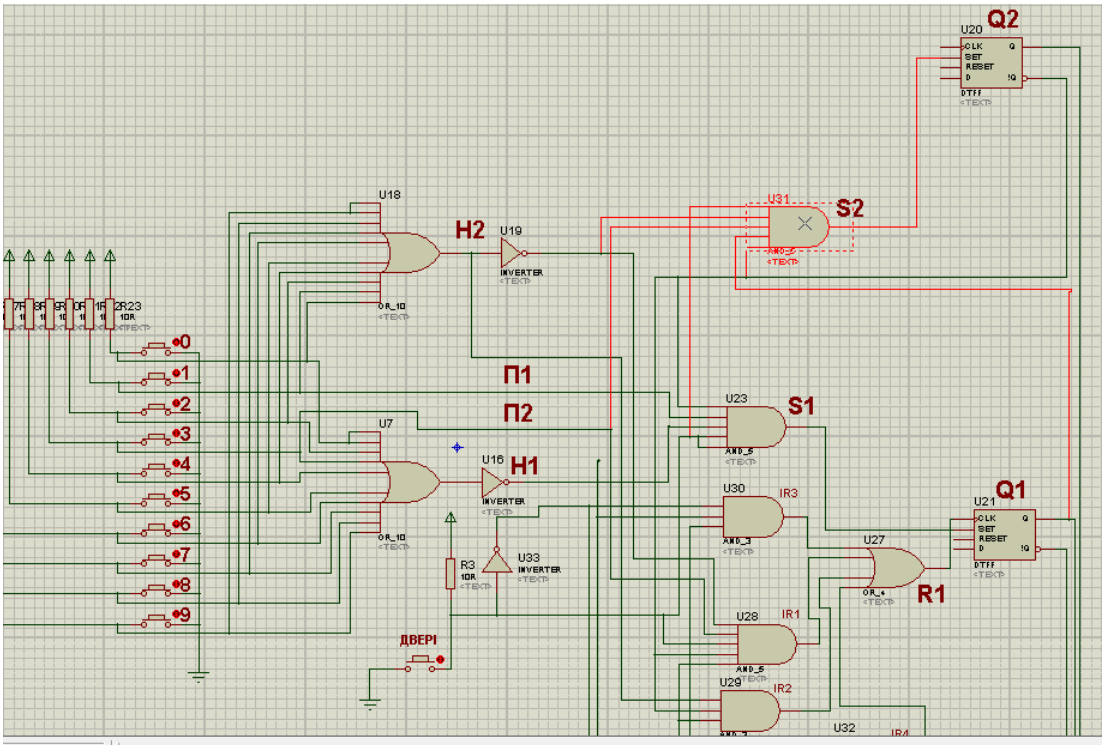


Рисунок 2.16

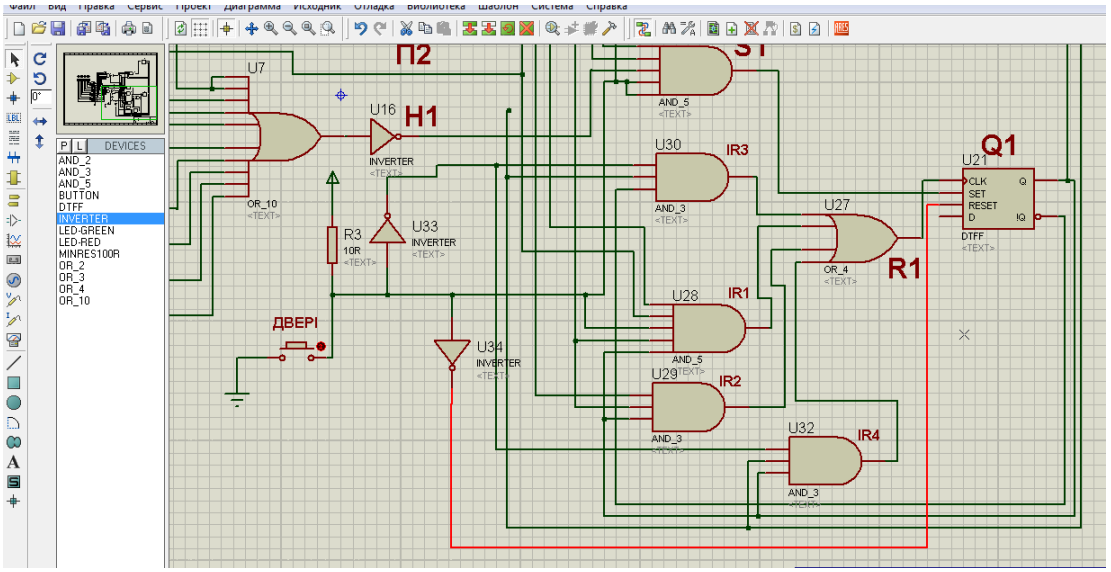


Рисунок 2.18

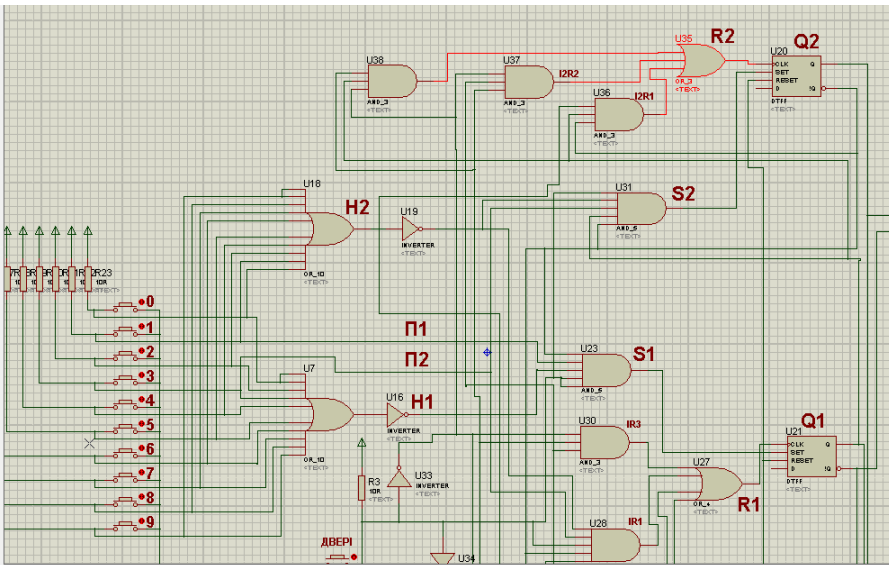


Рисунок 2.19

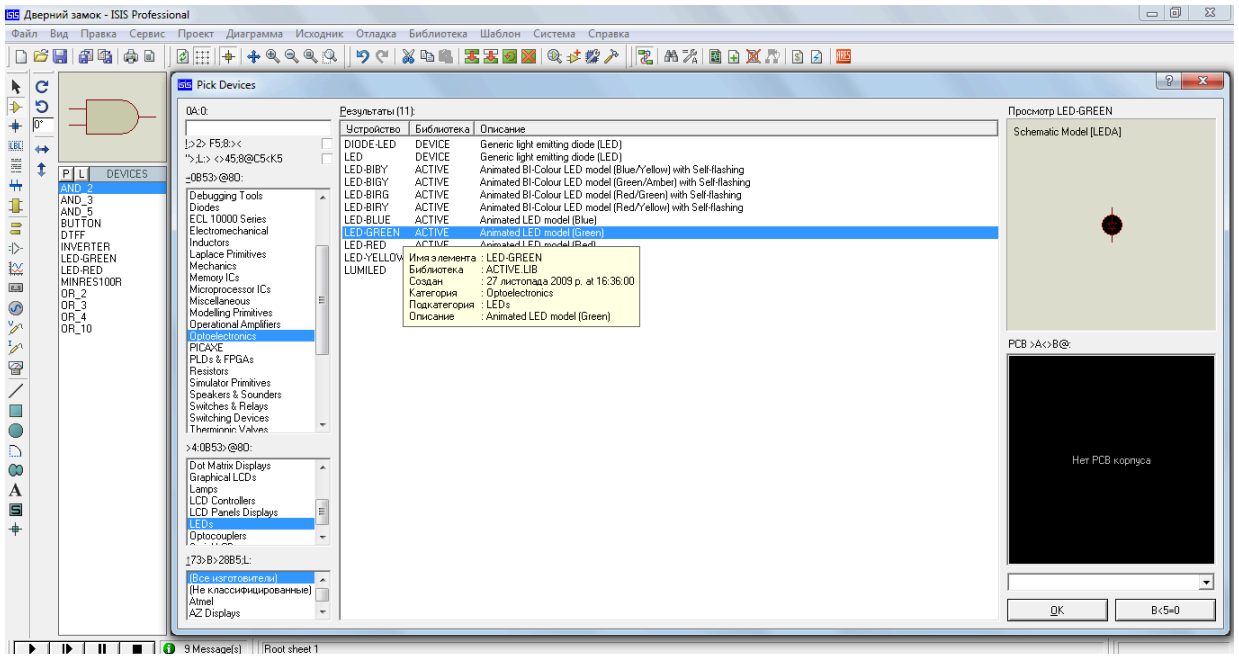


Рисунок 2.20

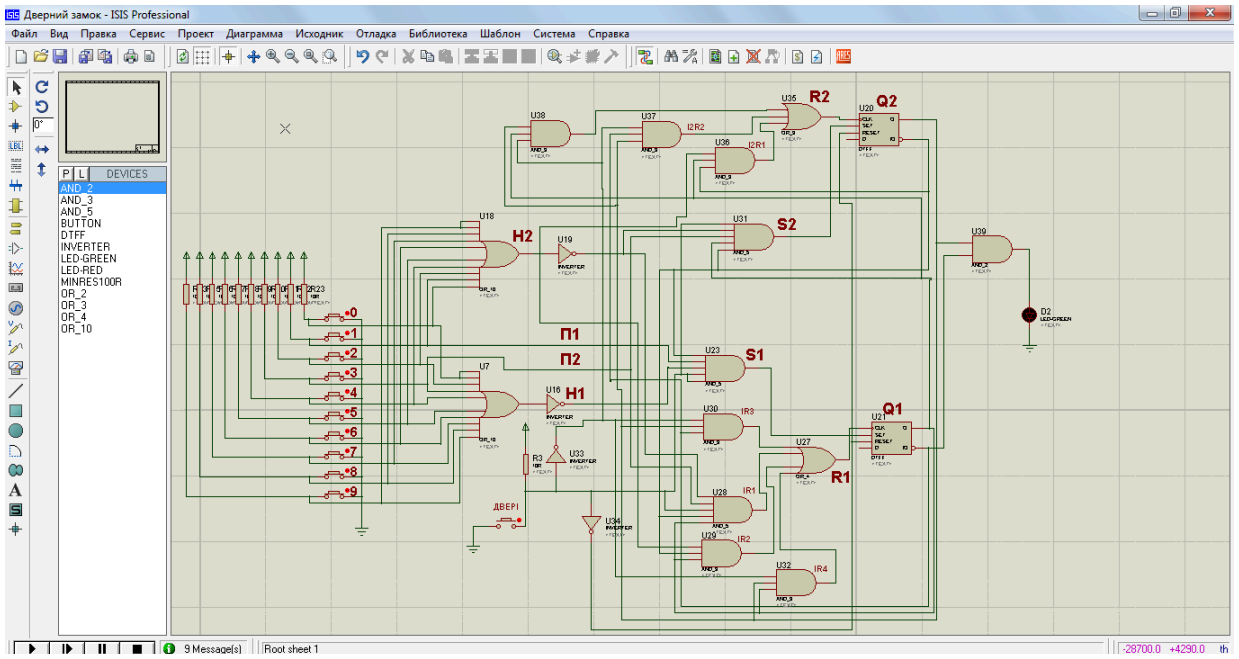


Рисунок 2.21