Datorteknik för civilingenjörer, HT18, DT509G
Lab 1: Digital Logic Circuits

Karl Eriksson & Simon Johansson
2018-09-26

Task 1: Breadboard setup and a NAND gate

The objective of task 1 was to test the NAND gate and see that it worked as expected with different output from the Arduino. We made a simple Arduino sketch that loop through every combination of the two output pins and at the end of each iteration we had a delay to see the outcome more clearly. The result worked as expected as shown below.

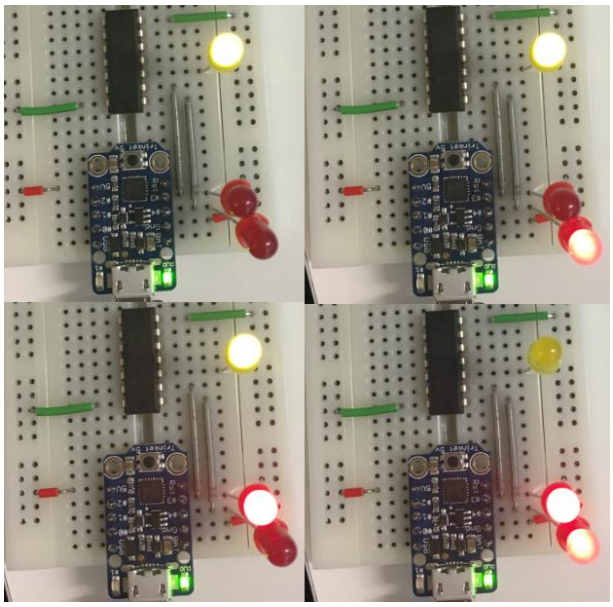


Fig 1. NAND-gate test. Input: red LED, output: yellow LED. (High=LED on)

Task 2: Half-adder

The problem was to implement a half-adder with a maximum of 4 NAND and one NOR gate. The half-adder circuit is an AND for the carry and an XOR for the sum. We found it quite easy to construct the AND part of the half-adder, two NAND-gates in sequence. But we could not find a way to create an XOR-gate with the remaining components so we chose to do it the other way.

Instead we started by creating the XOR with four NAND-gates and tried to find a way to make the AND part with the remaining NOR gate. To find the solution we made logic tables for all the parts of the XOR-gate. The idea was to find two components in the circuit that both gave zero (so the NOR would output one) only when input A and B were one. We found that component C and F satisfied the requirement as shown below in figure 2.

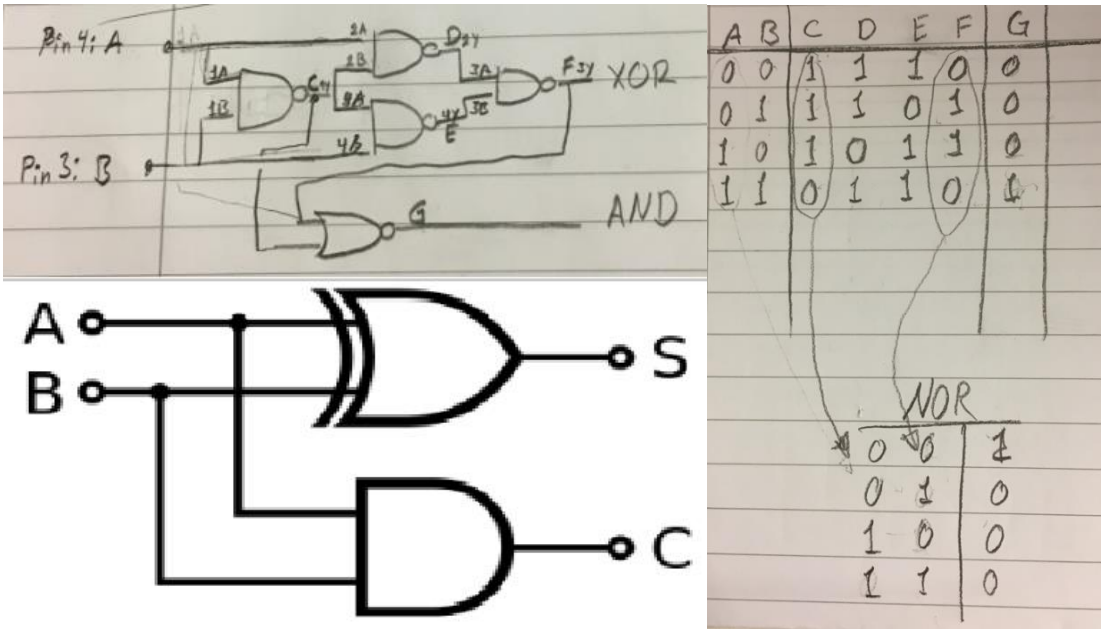


Fig 2. Half-adder schematic and logic table

The same sketch as in task 1 was used to test the new circuit. The half-adder should add two bits together. The sum is the left yellow LED and the carry is the right yellow LED. The two red LEDs are the two bits being added together.

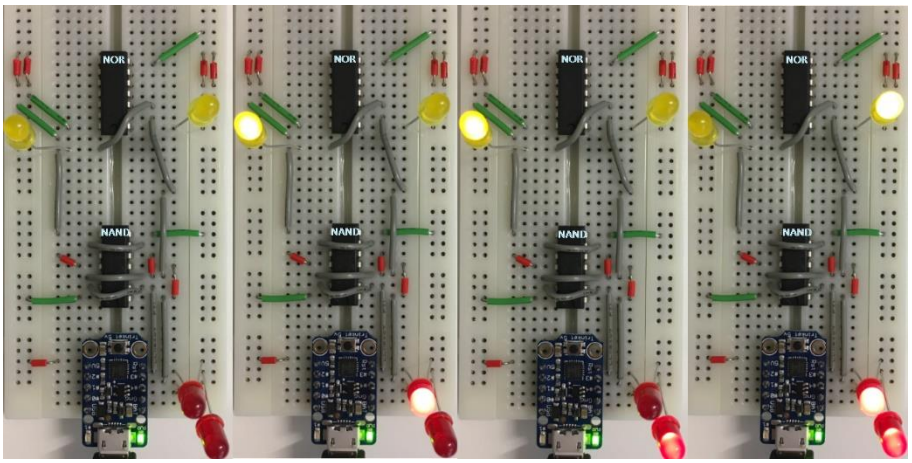


Fig 3. Half-adder circuit

Task 3: Full-adder and 2-bit adder circuit

The full-adder is made of two half-adders after each other, the sum of the first half-adder goes in to the second half-adder as well as a carry-in from a previous stage. The carry from the first half-adder and the carry from the second half-adder are OR-ed together as the carry out from the full adder. To make the OR-gate we used two NOR-gates in sequence.

To make the two-bit adder, the half-adder from task 2 and the full-adder from task 3 are put in sequence by connecting the carry from the half-adder to the carry-in of the full-adder. Four inputs are now required instead of two. Two for the half-adder inputs and two for the full-adder inputs. The final circuit is shown in figure 4.

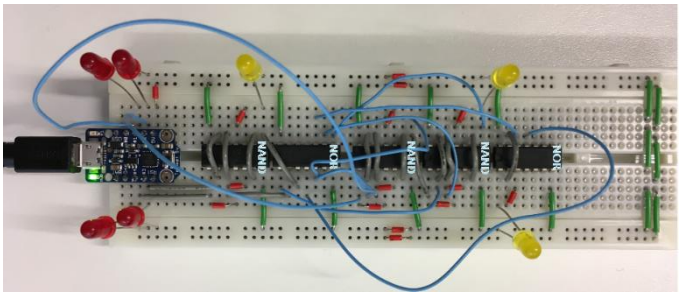
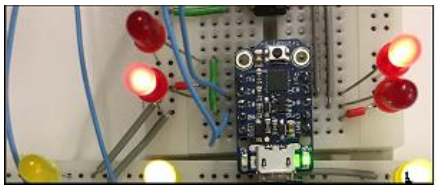


Fig 4. Two-bit adder circuit

The Arduino sketch from task 1 was extended to loop through the combinations of four inputs instead of two. To demonstrate the two-bit adder we moved the LEDs to make it more like normal binary adding as shown in figure 5. The entire result can be seen in figure 6.



	0		1
+	1		0
0	1		1

Fig 5. Binary addition

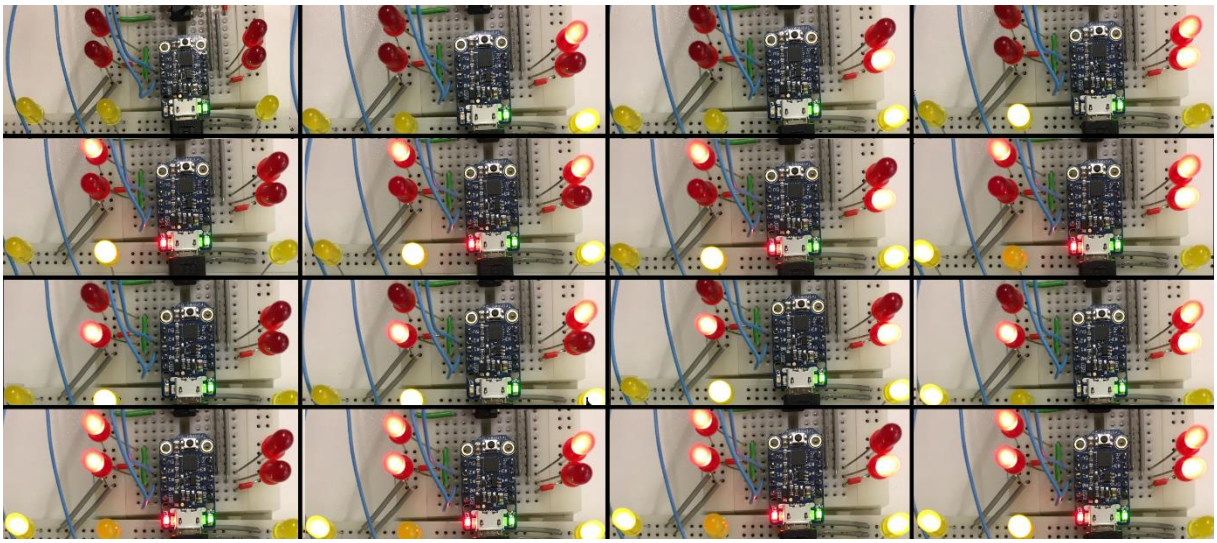


Fig 6. Two-bit adder result

Code for two bit adder:

```
int A = 4;
int B = 3;
int C = 2;
int D = 1;


void setup() {

                pinMode(A, OUTPUT);
                pinMode(B, OUTPUT);
                pinMode(C, OUTPUT);
                pinMode(D, OUTPUT);

}


void loop() {

                digitalWrite(A, LOW);
                digitalWrite(B, LOW);
                digitalWrite(C, LOW);
                digitalWrite(D, LOW);
                delay(1000);
                digitalWrite(A, LOW);
                digitalWrite(B, HIGH);
                digitalWrite(C, LOW);
                digitalWrite(D, LOW);
                delay(1000);
                digitalWrite(A, HIGH);
                digitalWrite(B, LOW);
                digitalWrite(C, LOW);
                digitalWrite(D, LOW);
                delay(1000);
                digitalWrite(A, HIGH);
                digitalWrite(B, HIGH);
                digitalWrite(C, LOW);
                digitalWrite(D, LOW);
                delay(1000);

                digitalWrite(A, LOW);
                digitalWrite(B, LOW);
                digitalWrite(C, LOW);
                digitalWrite(D, HIGH);
                delay(1000);
                digitalWrite(A, LOW);
                digitalWrite(B, HIGH);
                digitalWrite(C, LOW);
                digitalWrite(D, HIGH);
                delay(1000);
                digitalWrite(A, HIGH);
                digitalWrite(B, LOW);
                digitalWrite(C, LOW);
                digitalWrite(D, HIGH);
                delay(1000);
```

```
    digitalWrite(A, HIGH);
    digitalWrite(B, HIGH);
    digitalWrite(C, LOW);
    digitalWrite(D, HIGH);
    delay(1000);

    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, HIGH);
    digitalWrite(D, LOW);
    delay(1000);
    digitalWrite(A, LOW);
    digitalWrite(B, HIGH);
    digitalWrite(C, HIGH);
    digitalWrite(D, LOW);
    delay(1000);
    digitalWrite(A, HIGH);
    digitalWrite(B, LOW);
    digitalWrite(C, HIGH);
    digitalWrite(D, LOW);
    delay(1000);
    digitalWrite(A, HIGH);
    digitalWrite(B, HIGH);
    digitalWrite(C, HIGH);
    digitalWrite(D, LOW);
    delay(1000);

    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, HIGH);
    digitalWrite(D, HIGH);
    delay(1000);
    digitalWrite(A, LOW);
    digitalWrite(B, HIGH);
    digitalWrite(C, HIGH);
    digitalWrite(D, HIGH);
    delay(1000);
    digitalWrite(A, HIGH);
    digitalWrite(B, LOW);
    digitalWrite(C, HIGH);
    digitalWrite(D, HIGH);
    delay(1000);
    digitalWrite(A, HIGH);
    digitalWrite(B, HIGH);
    digitalWrite(C, HIGH);
    digitalWrite(D, HIGH);
    delay(1000);

}
```