# Assembly Language Report:

Table of Contents

[Assembly Language Report: 1](#_Toc513921097)

[Flowchart: 2](#_Toc513921098)

[LED flowchart: 2](#_Toc513921099)

[NOP flowchart: 3](#_Toc513921100)

[The theory behind the code: 4](#_Toc513921101)

[How the LED’s work: 4](#_Toc513921102)

[The phases would be as follows: 4](#_Toc513921103)

## Flowchart:

## LED flowchart: A close up of text on a white background Description generated with very high confidence

## NOP flowchart:

A picture containing photo, indoor, sky, wall

Description generated with high confidence

## The theory behind the code:

### Delay\_ms:

The first part was figuring out and getting the delay function to work. To do this I started by finding out what 1 nop (no operation), which was 1 clock cycle in this instance. Knowing that the blink code runs in 2000ms + timing overheads (approx. 17ms) and needed to get it to 2017 (+/- 0.05%). With a quick google search I found that one cycle is equivalent to 62.5ms. Next, I added 5 nop’s to see how close to 2017 I got; 5 nop’s equating to 312.5ms. So with this I calculated that with by using 12 nop’s it would get me into that 2017 with +/-0.05% accuracy.

A screenshot of a map

Description generated with very high confidence

### How the LED’s work:

Each bit of the binary number represents one of the LEDS, so if the is 1 then the LED would turn on and for 0 it would turn off. Therefore, 111111 would turn on all the LED’s and 000000 would turn all the LED’s off.

Pin 13 = GreenN ; Bit 5 = 100000

Pin 12 = AmbeN ; Bit 4 = 010000

Pin 11 = RedN ; Bit 3 = 001000

Pin 10 = GreenE ; Bit 2 = 000100

Pin 9 = AmberE ; Bit 1 = 000010

Pin 8 = RedE ; Bit 0 = 000001

Furthermore, to turn on more than one light on you would have to add those to together, for example; for an antiphase light system you would want one green light on and the other off, therefore, Pin 13 (green LED on the north / south direction) and Pin 8 (red LED on the east / west direction) you would do; 100000 + 100000 = 100001.

### The phases would be as follows:

One light cycle {

* Pin 13 + Pin 8; Bit 5 & 0
* Pin 12 + Pin 8; Bit 4 & 0
* Pin 11 + Pin 9 + Pin 8; Bit 3 & 1 & 0
* Pin 11 + Pin 9; Bit 3 & 1
* Pin 11 + Pin 10; Bit 3 & 2
* Pin 11+ Pin 9; Bit 3 & 1
* Pin 11 + Pin 12 + Pin 8; Bit 3 & 4 & 0

}

One binary cycle {

* 100000 + 000001 = 100001
* 010000 + 000001 = 010001
* 001000 + 000010 + 000001 = 001011
* 001000 + 000010 = 001010
* 001000 + 000100 = 001100
* 001000 + 000010 = 001010
* 001000 + 010000 + 000001 = 011001

}

\*The Arduino board uses hex code, therefore, I got my hex numbers from: <https://www.rapidtables.com/convert/number/binary-to-hex.html>

A screenshot of a cell phone

Description generated with very high confidence

## Reflection:

Setting up a flowchart first was definitely something I should have done for the other 2 assignments as it made it much much clearer of how I should layout and the process of my code. Assembly is extremely challenging and involved a lot more than I had anticipated.

## Problems encountered:

My first problem was understanding what on earth a nop was, how they worked and how the code worked. Furthermore, how changing the delay and what the cycles did. After that it was all the math with the binary, bits and hex’s.