

# Cara Kerja 4 Bit Fisherman

By Nathanael Rachmat, NIM 13523142

Disimulasikan dengan Digital Logic Sim

Dalam Digital Logic Sim, semua input dari semua komponen berada di kiri, dan semua output dari semua komponen berada di kanan.

Untuk menggunakan program simulasi komputer, download aplikasi dari [Download Digital Logic Sim by Sebastian Lague - itch.io](#). Pilih sesuai OS Laptop dan perhatikan lokasi penyimpanan project digital logic sim masing2 OS:

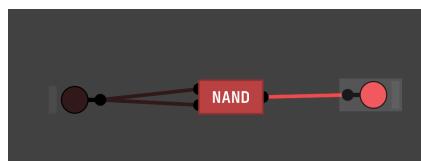
```
MAC: /Users/<your-username>/Library/Application  
Support/com.DefaultCompany.2DProject/Projects/  
Windows:  
C:\Users\<your-username>\AppData\LocalLow\SebastianLague\Digital-Logic-Sim\Projects\  
Linux: /home/<your-username>/.config/unity3d/Sebastian Lague/Digital Logic Sim/Projects/
```

Di lampiran (bawah), telah disediakan file sebuah project digital logic sim. Bawa zip file tersebut ke lokasi direktori yang tercantum di atas dan extract di dalam folder Projects. Ketika sudah ada folder "4 bit" dalam folder Projects, maka Aplikasi Digital Logic Sim sudah siap untuk dijalankan. Pilih open project dan pilih "4 bit". Click kanan "Computer" pada menu bawah dengan melakukan scroll dan pilih open. Simulasi komputer yang telah dibuat siap untuk dicoba.

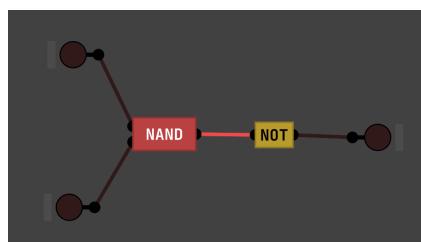
Di bawah ini merupakan dokumentasi setiap komponen penting yang dipakai untuk merancang simulasi komputer ini dan penjelasannya.

## Logical Gate Dasar:

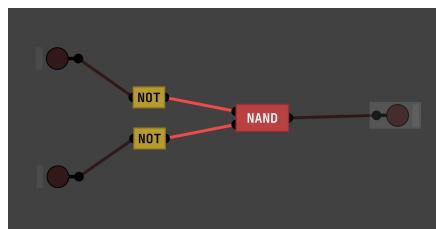
1. NAND (Paling dasar)
2. NOT



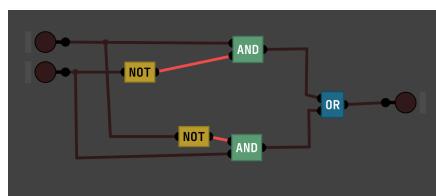
3. AND



4. OR

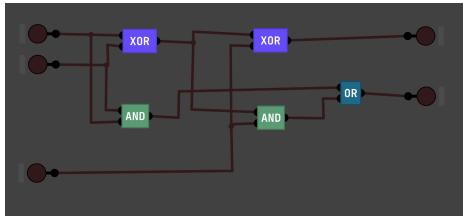


5. XOR

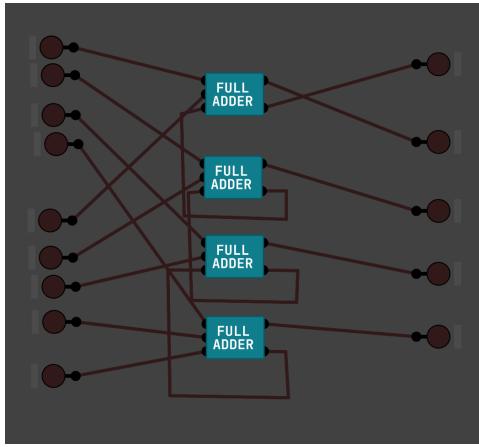


# Pertambahan Dasar:

## 1. Full Adder

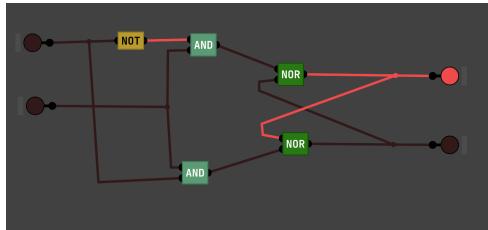


## 2. Ripple Carry Adder

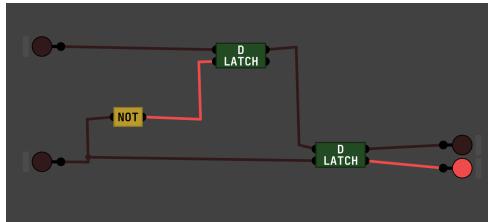


# Penyimpanan Memori Bit Dasar:

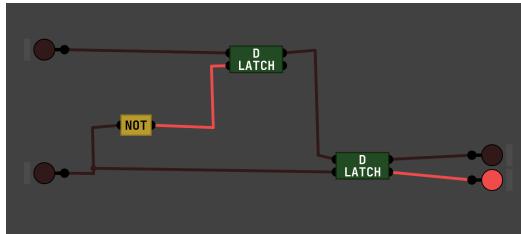
## 1. D-Latch



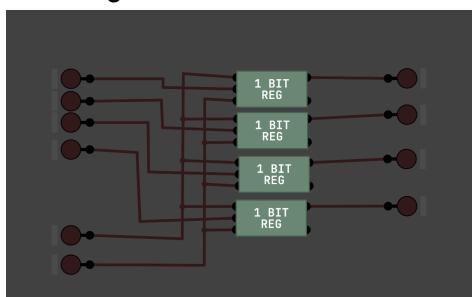
## 2. D Flip Flop



## 3. 1 Bit Register

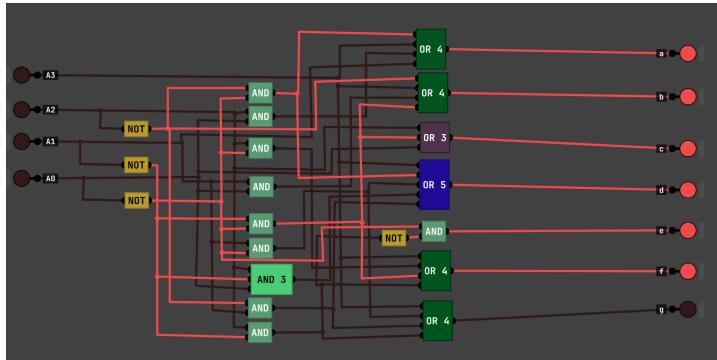


## 4. 4 Bit Register

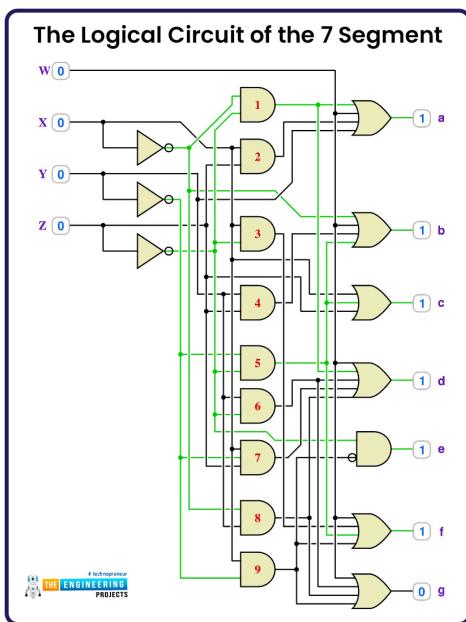


# 7 Segment:

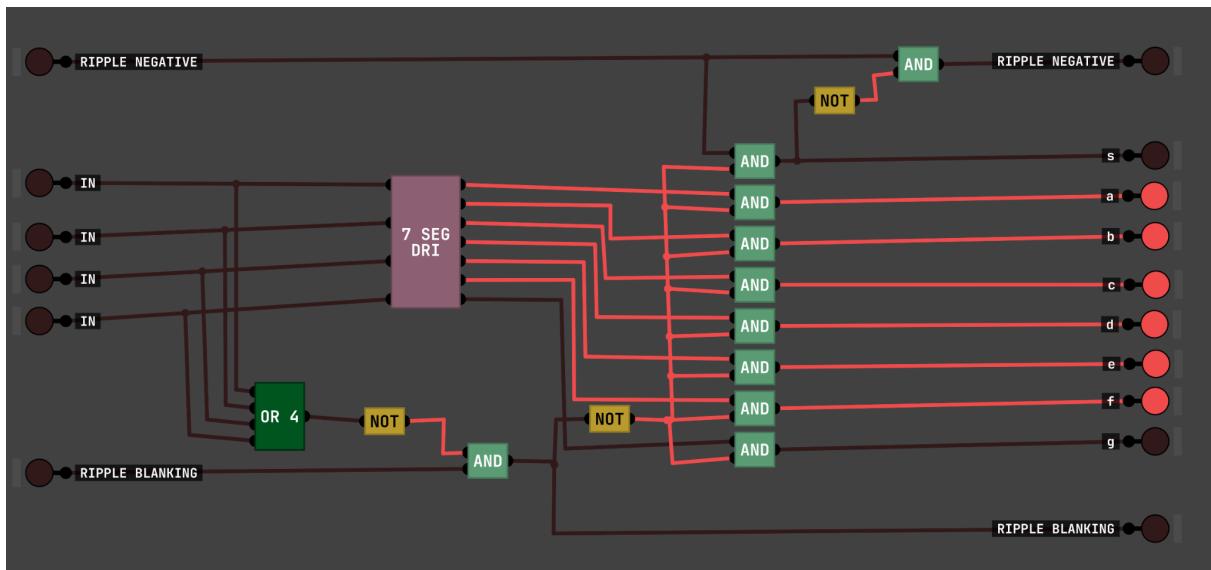
## 1. 7 Segment Driver



Cara kerja 7 segment driver di atas mengikuti skema logic gate di bawah ini:

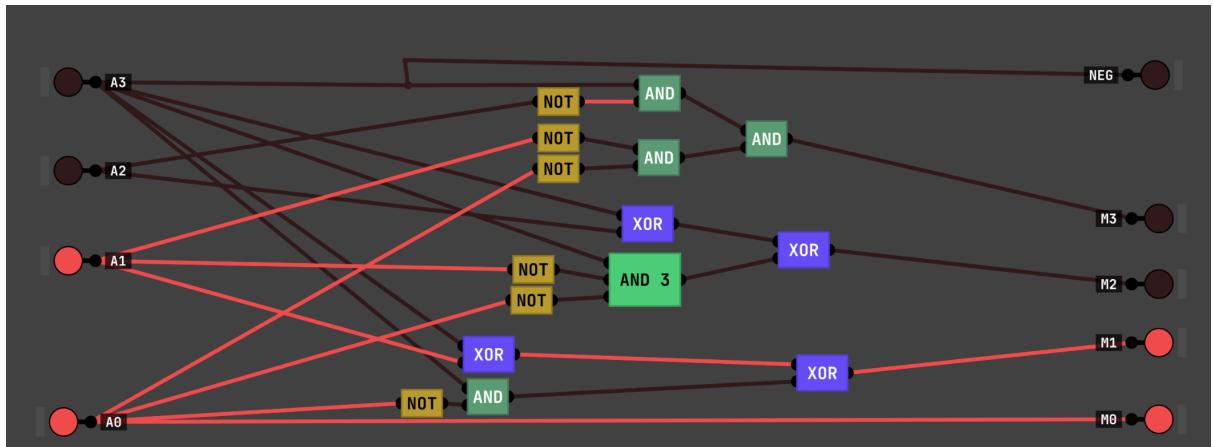


## 2. Blanking Negative Drive



Komponen ini dibuat dengan fitur tambahan blanking (tidak menyala ketika bernilai nol), dan ripple negative untuk petanda minus

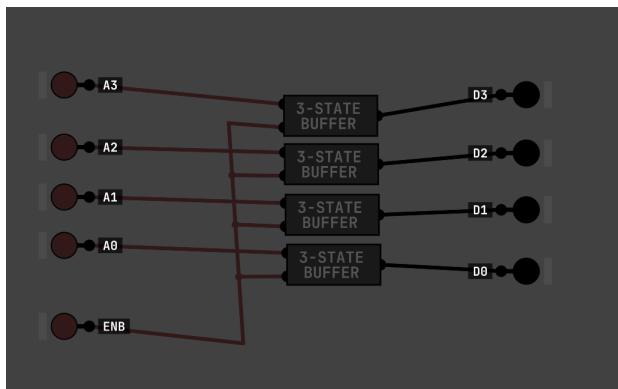
### 3. Signed converter



Mengubah dari unsigned integer menjadi versi signed dari integer. Output “Neg” menunjukkan bahwa angka itu negatif.

## BUFFER

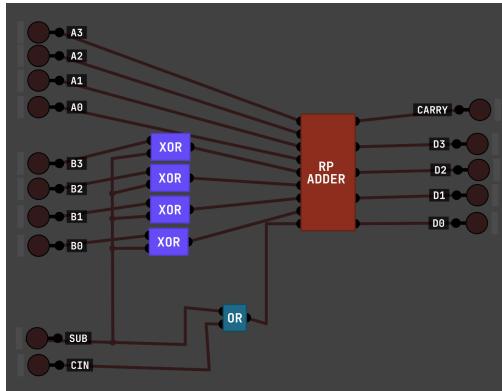
1. 3 State Buffer (Paling dasar)
2. Buffer 4 bit



Digunakan untuk menentukan apakah signal yang mengalir lewat input itu memiliki output atau tidak.

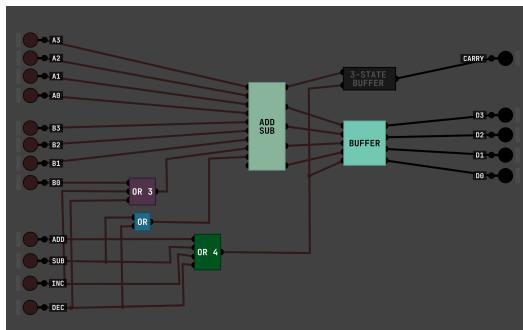
# Aritmatika

## 1. Pertambahan dan pengurangan (Add Sub)



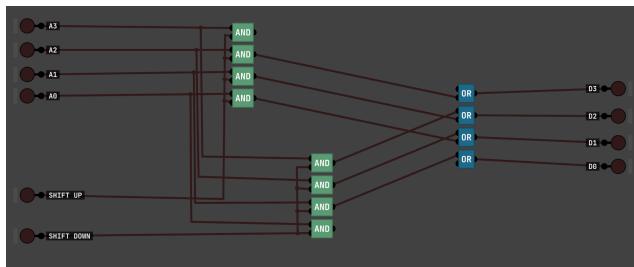
Dalam komponen ini, dapat dipilih untuk menambah atau mengurangi dua angka 4 bit.

## 2. Komponen Aritmatika (Arithmetic Unit)

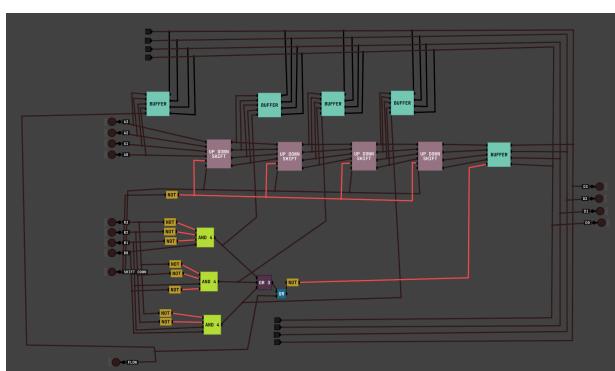


Komponen ini sama seperti Add Sub tapi dengan adanya kontrol aliran data melalui buffer

## 3. Shift Atas/Bawah (Left/Right)



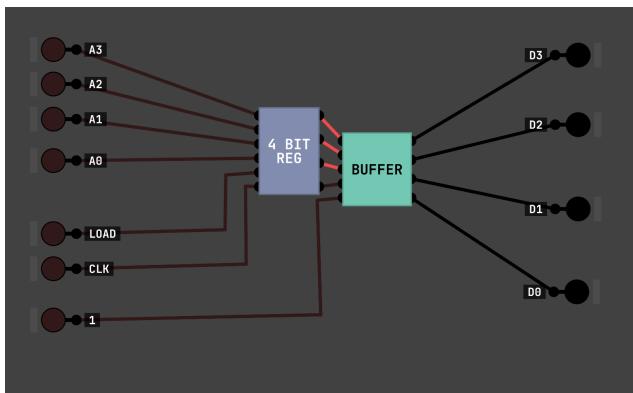
## 4. Shift x kali



Shift right or left dengan input berapa kali di shift.

# Penyimpanan Memori 4 Bit

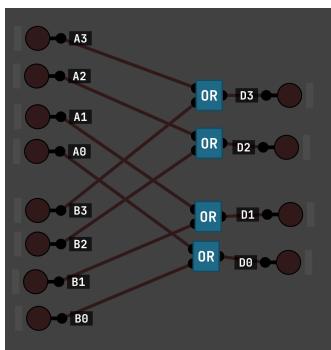
## 1. 1x 4 bit RAM



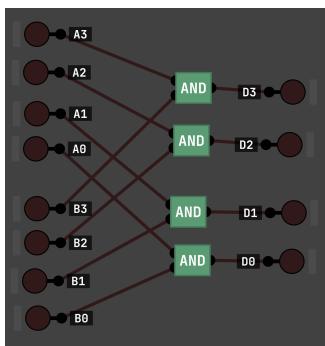
RAM digunakan untuk menyimpan memori dengan kontrol aliran data.

# Operasi Bitwise

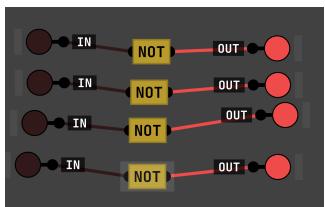
## 1. Or Bitwise



## 2. And Bitwise

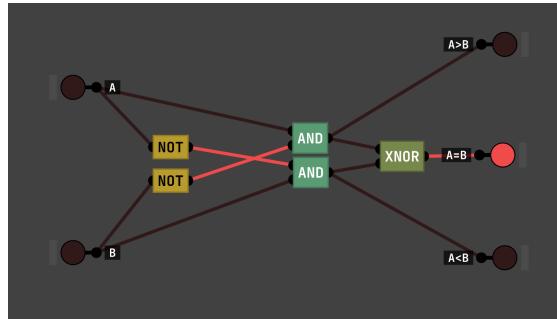


## 3. Not Bitwise

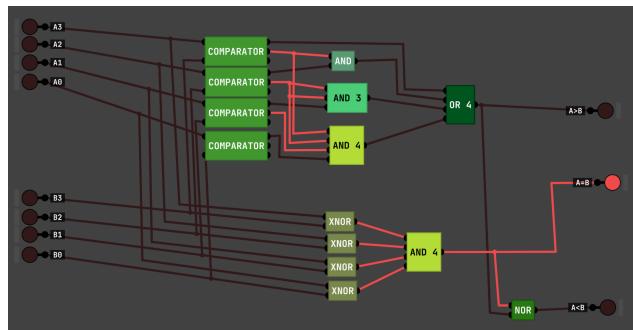


# Pembanding

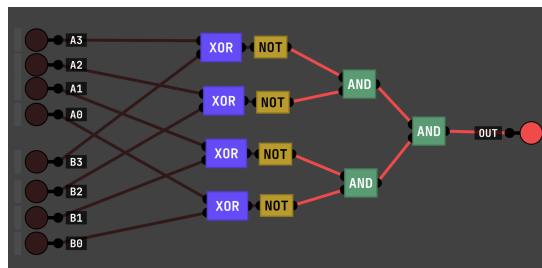
## 1. Comparator 1 Bit



## 2. Comparator 4 Bit

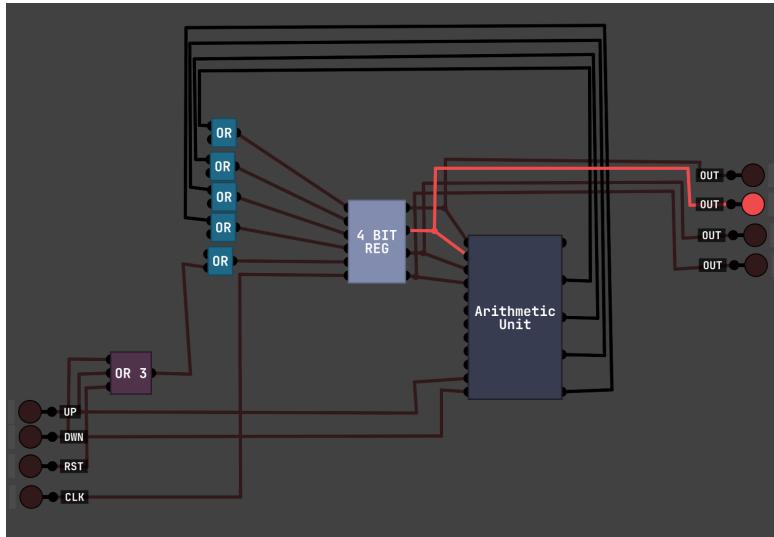


## 3. Persamaan 4 Bit

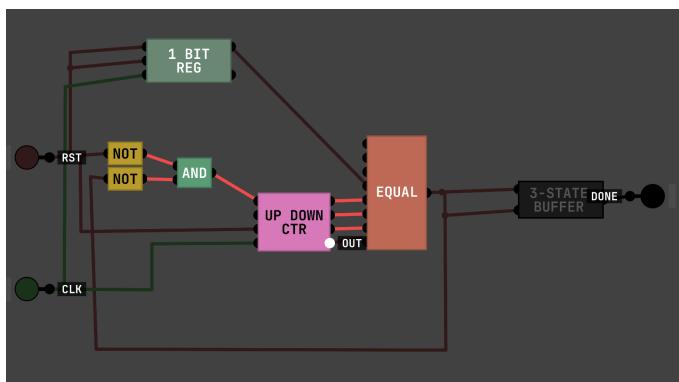


# Perulangan

## 1. Up and Down counter

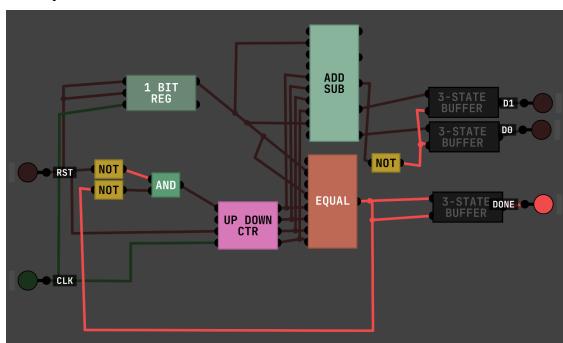


## 2. Loop sekali



Sekali clock nyala, output akan terus bernilai 1

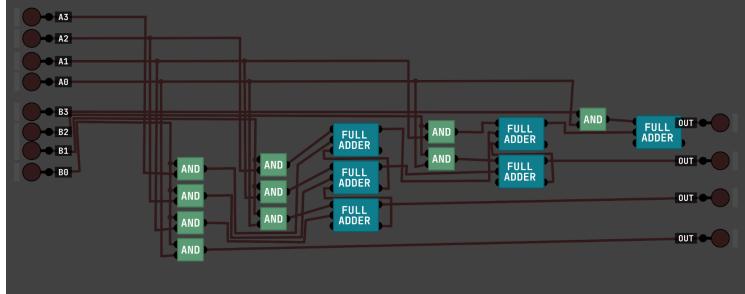
## 3. Loop 5x



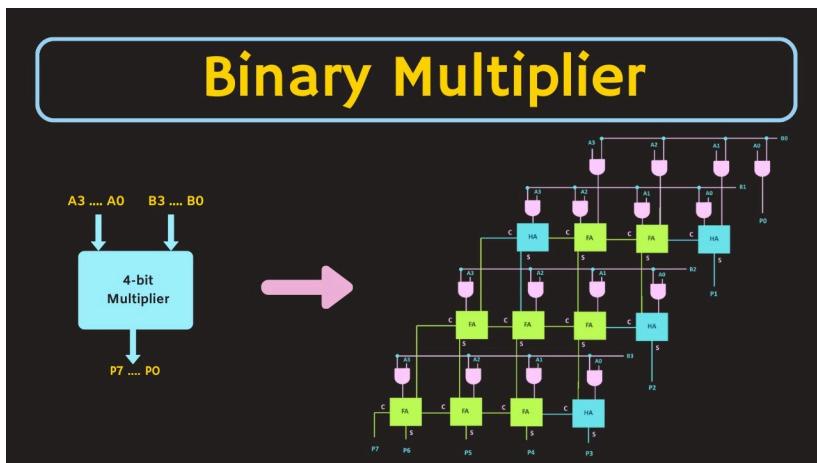
Clock akan nyala mati hingga 5 kali agar output bernilai 1 (yang bawah), dengan tambahan output 4 dikurang index (3 hingga 0)

# Perkalian dan Pembagian

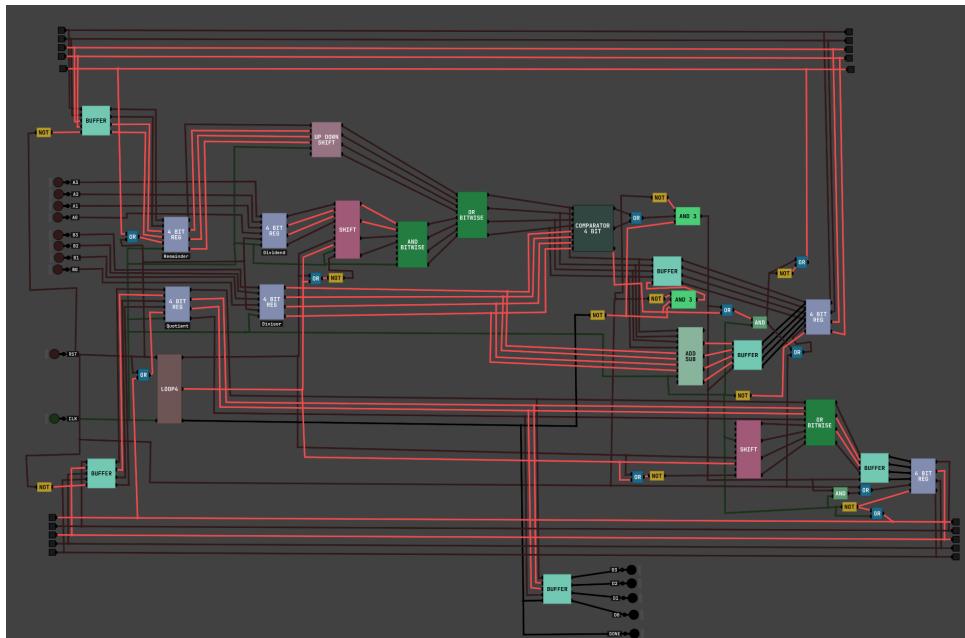
## 1. Perkalian 4 bit



Komponen perkalian di atas menggunakan skema yang sama seperti di bawah ini, namun dibatasi sampai 4 bit saja



## 2. Pembagian



Komponen di atas memanfaatkan clock untuk menyimpan ulang yang telah dioperasikan ke register secara berulang, agar dapat mensimulasikan looping. Cara kerja pembagian ini mengikuti cara kerja pembagian di bahasa c secara bitwise:

```

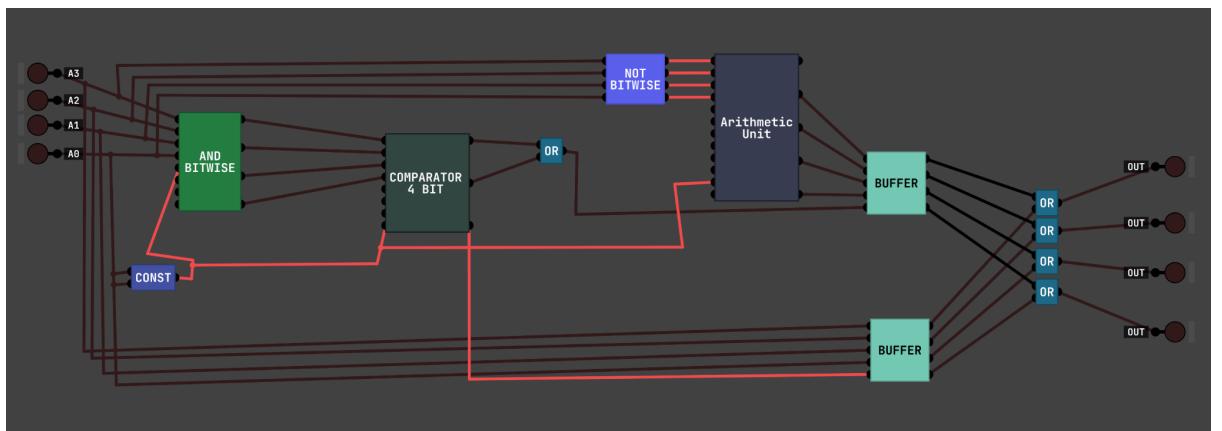
void divide4(unsigned dividend, unsigned divisor,
            unsigned *quotient, unsigned *remainder) {
    unsigned q = 0;
    unsigned r = 0;

    for (int i = 3; i >= 0; i--) {
        r = (r << 1) | ((dividend >> i) & 1);
        if (r >= divisor) {
            r -= divisor;
            q |= (1 << i);
        }
    }

    *quotient = q;
    *remainder = r;
}

```

### 3. Absolut



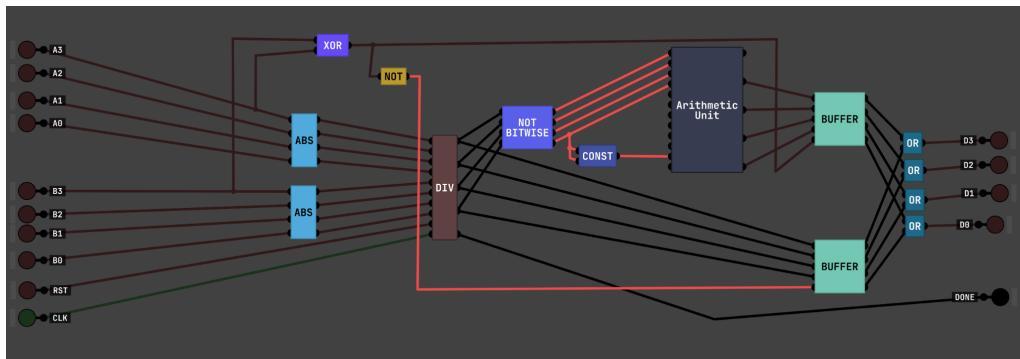
Cara kerja komponen absolut mengikuti cara kerja absolut di bahasa c dengan operasi bitwise:

```

unsigned abs4(unsigned x) {
    if (x & 0x8) {
        return ((~x) + 1) & 0xF;
    } else {
        return x & 0xF;
    }
}

```

### 4. Signed Division



Komponen diatas mengikuti cara kerja signed division di bahasa C menggunakan operasi bitwise:

```

void divide4_signed(unsigned dividend, unsigned divisor,
                    unsigned *quotient) {
    unsigned sign_dividend = (dividend >> 3) & 1;
    unsigned sign_divisor = (divisor >> 3) & 1;

    unsigned mag_dividend = abs4(dividend);
    unsigned mag_divisor = abs4(divisor);

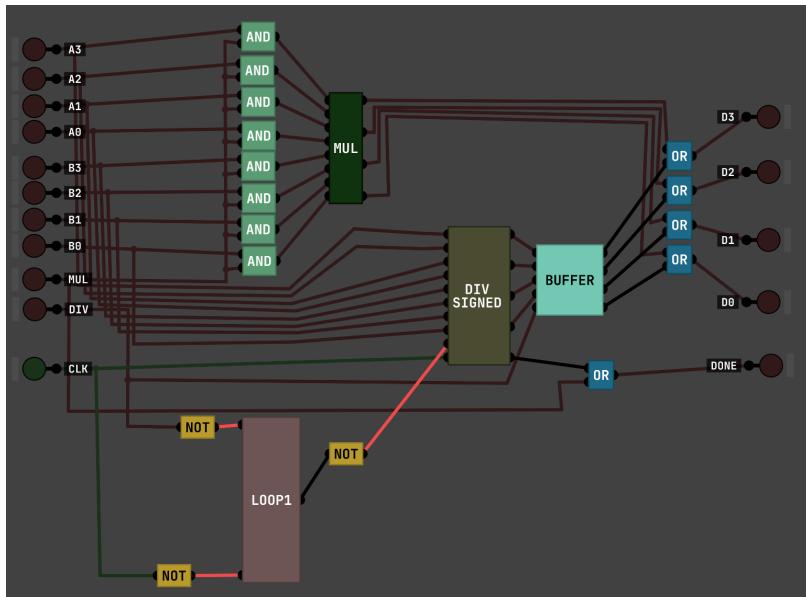
    unsigned q, r;
    divide4(mag_dividend, mag_divisor, &q, &r);

    if (sign_dividend ^ sign_divisor) {
        q = ((~q) + 1) & 0xF; // two's complement 4-bit
    }

    *quotient = q & 0xF;
}

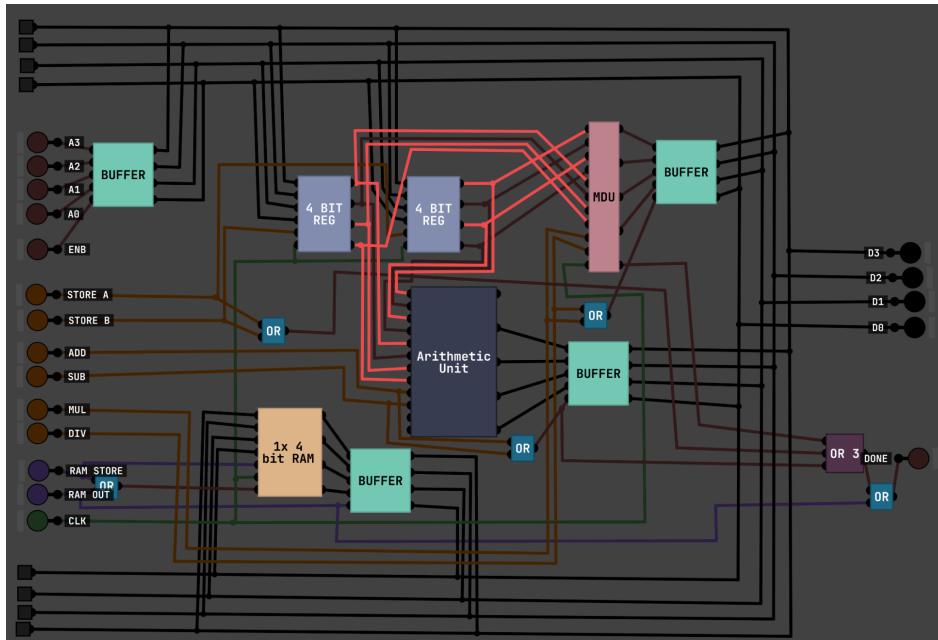
```

##### 5. Perkalian dan pembagian (MDU):



Hasil gabungan komponen multiply dan division signed.

# Arithmetic Logic Unit (with MDU and Memory Unit)



Cara kerja komponen ini sederhana. Pertama, ditentukan apa yang menjadi input dan apa yang menjadi output. Input dibagi menjadi input data, dan input kontrol:

## Data Inputs:

- D0–D3 (4-bit literal) : Data yang ingin disimpan/dioperasikan
- Data ENB : Meng-ENaBLE data input (D0-D3) ke bus buffer

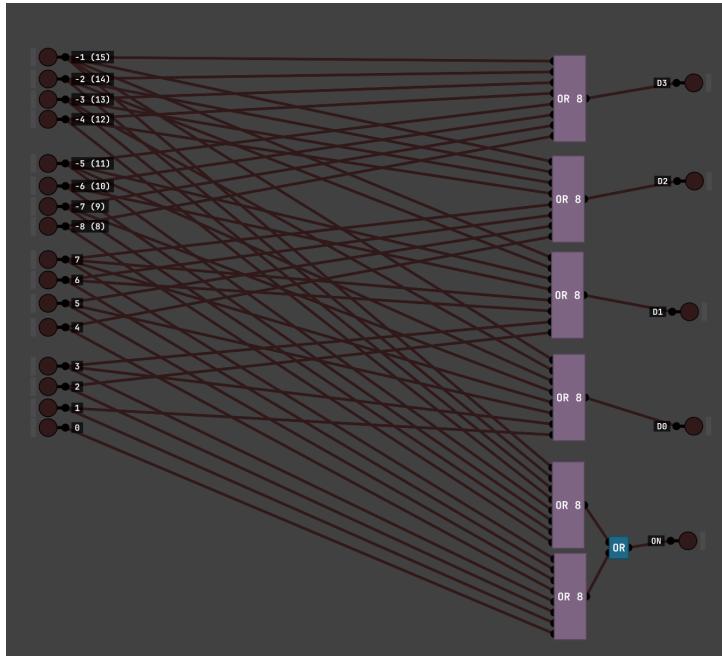
## Control Inputs:

- STORE A : Menyimpan data yang berada di bus buffer ke register A.
- STORE B : Menyimpan data yang berada di bus buffer ke register B.
- ADD : Menambahkan data yang berada di register A dan B dan hasilnya di-enable ke bus buffer.
- SUB : Mengurangi data yang berada di register A dengan data yang berada di register B dan hasilnya di-enable ke bus buffer.
- MUL : Mengalikan data yang berada di register A dan B dan hasilnya di-enable ke bus buffer.
- DIV : Membagi data yang berada di register A dengan data yang berada di register B dan hasilnya di-enable ke bus buffer.
- RAM STORE : Menyimpan data yang sedang di-enable di bus buffer ke RAM.
- RAM OUT : Meng-enable data yang disimpan dalam RAM ke bus buffer.
- CLOCK : Pulse yang digunakan untuk sinkronisasi penyimpanan memori

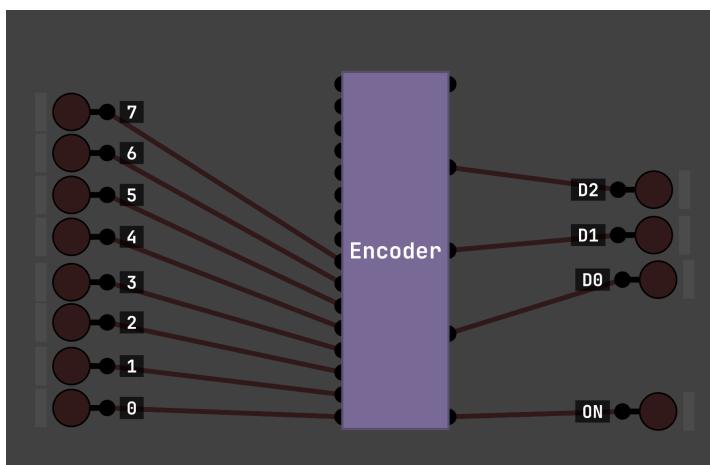
Untuk outputnya itu sendiri, lumayan sederhana hanya binary yang sedang berada di bus saat ini dan flag “Done”. Flag “Done” dipakai (hanya untuk division) untuk menandakan apakah data di bus sekarang siap untuk ditampilkan di output nantinya. Yaitu 7 segment display.

# Digital Encoder

## 1. 4 Bit Encoder:



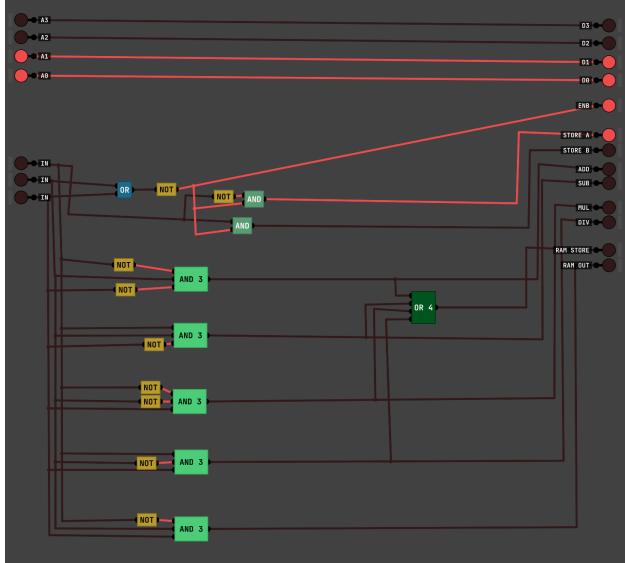
## 2. 3 Bit Encoder



Menggunakan 4 bit encoder yang sudah ada

# Control Unit, CPU, dan Komputer

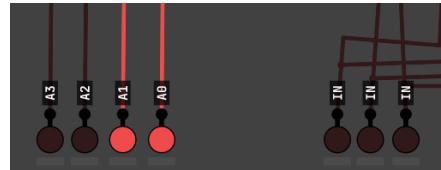
## 1. Control Unit



Komponen ini dibuat sedemikian sehingga mengikuti instruksi berikut:

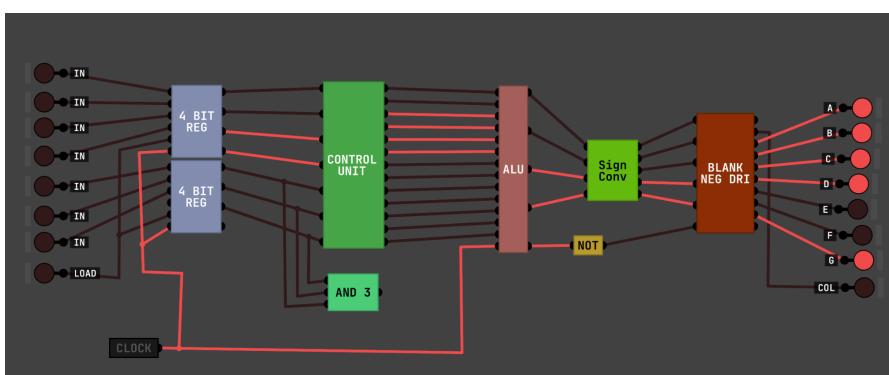
### CU INSTRUCTIONS :

```
xxxx 000 → STORE literal ke Reg A  
xxxx 001 → STORE literal ke Reg B  
---- 010 → ADD (A+B → RAM)  
---- 011 → SUB (A-B → RAM)  
---- 100 → MUL (A×B → RAM)  
---- 101 → DIV (A÷B → RAM)  
---- 110 → OUTPUT RAM (Display)  
---- 111 → NOP
```



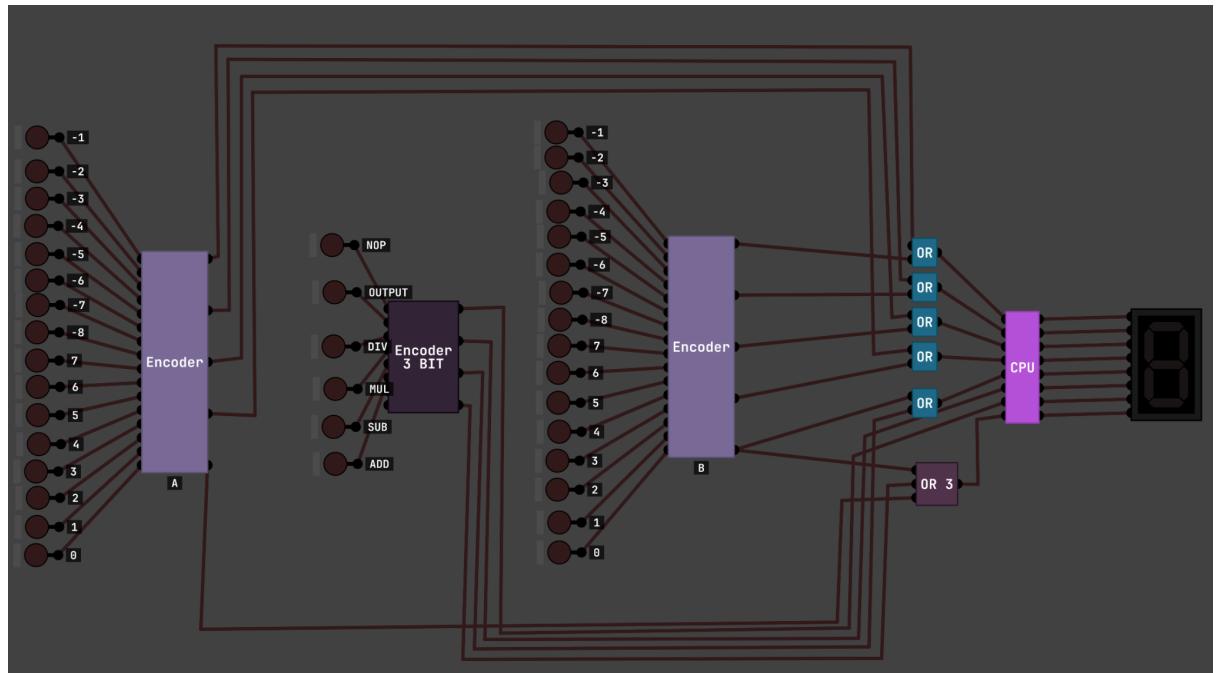
Output yang dihasilkan akan otomatis menjalankan sebuah fungsi di komponen ALU.

## 2. CPU



Input berisi 7 bit instruksi CU dan 1 bit load sebagai kontrol aliran data. Clock mensinkronisasikan seluruh penyimpanan memory. Control Unit dipasangkan dengan ALU, dan hasil ALU jika sudah selesai (done flag) akan di teruskan ke output yang sudah sesuaikan dengan format 7 segment display melalui blanking negative driver (yang memiliki 7 segment driver di dalam nya)

### 3. Komputer Sederhana untuk Menghitung 4 Bit



Sudah jadi

## Lampiran

Demonstrasi:

<https://drive.google.com/file/d/1WqgiNf75YEQbR1kzG8741JGI4BqvTbD0/view?usp=sharing>

Catatan: Warna biru pada 7 segment display menunjukkan tanda negatif.

File terkait:

[https://drive.google.com/file/d/1UfzzF57MCJ8b5y\\_r93KAt7ryxuZTmXeI/view?usp=sharing](https://drive.google.com/file/d/1UfzzF57MCJ8b5y_r93KAt7ryxuZTmXeI/view?usp=sharing)