

# 4bitCPU俯瞰

ComputerScience集会#12 @VRChat 2023-11-07

夜鍋ヨナ-yonabeyona <<http://x.com/yonabeyona>>

# CS 集会所

# # 12

# 設計中のCPU

# 俯瞰しよう

# 自己紹介

- 名前：夜鍋 ヨナ(よなべ よな)
- X(Twitter) : yonabeyona, yonabeyona\_sub
- Discord : yona\_47
- その他
  - ComputerScienceが好き
  - 数学勉強中
  - 物理も勉強中
  - ComputerScienceの中でも、ComputerArchitectureが好き



# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- PC(Program Counter)
- デコーダ
- セレクタ

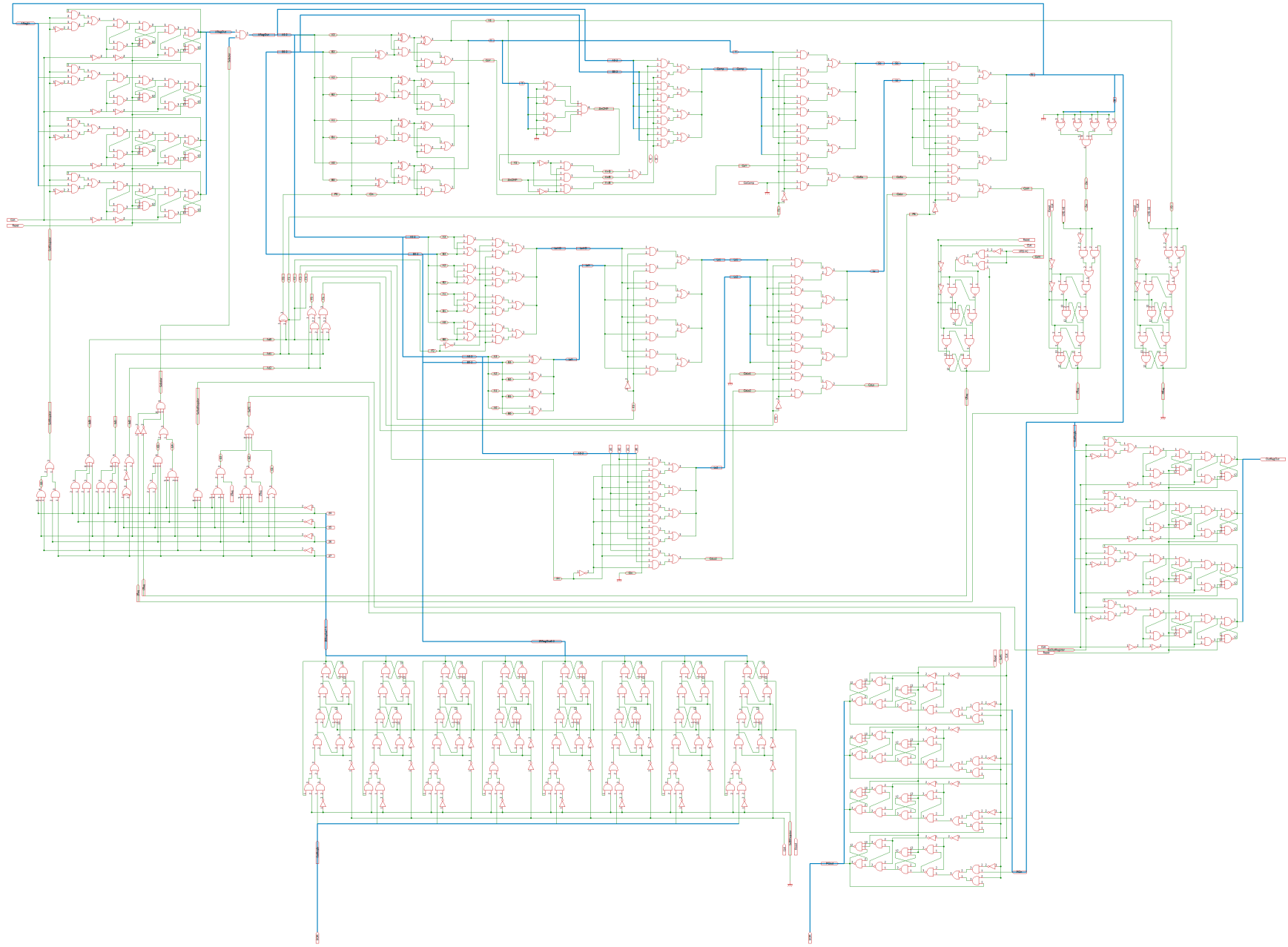
# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- PC(Program Counter)
- デコーダ
- セレクタ

# 俯瞰する

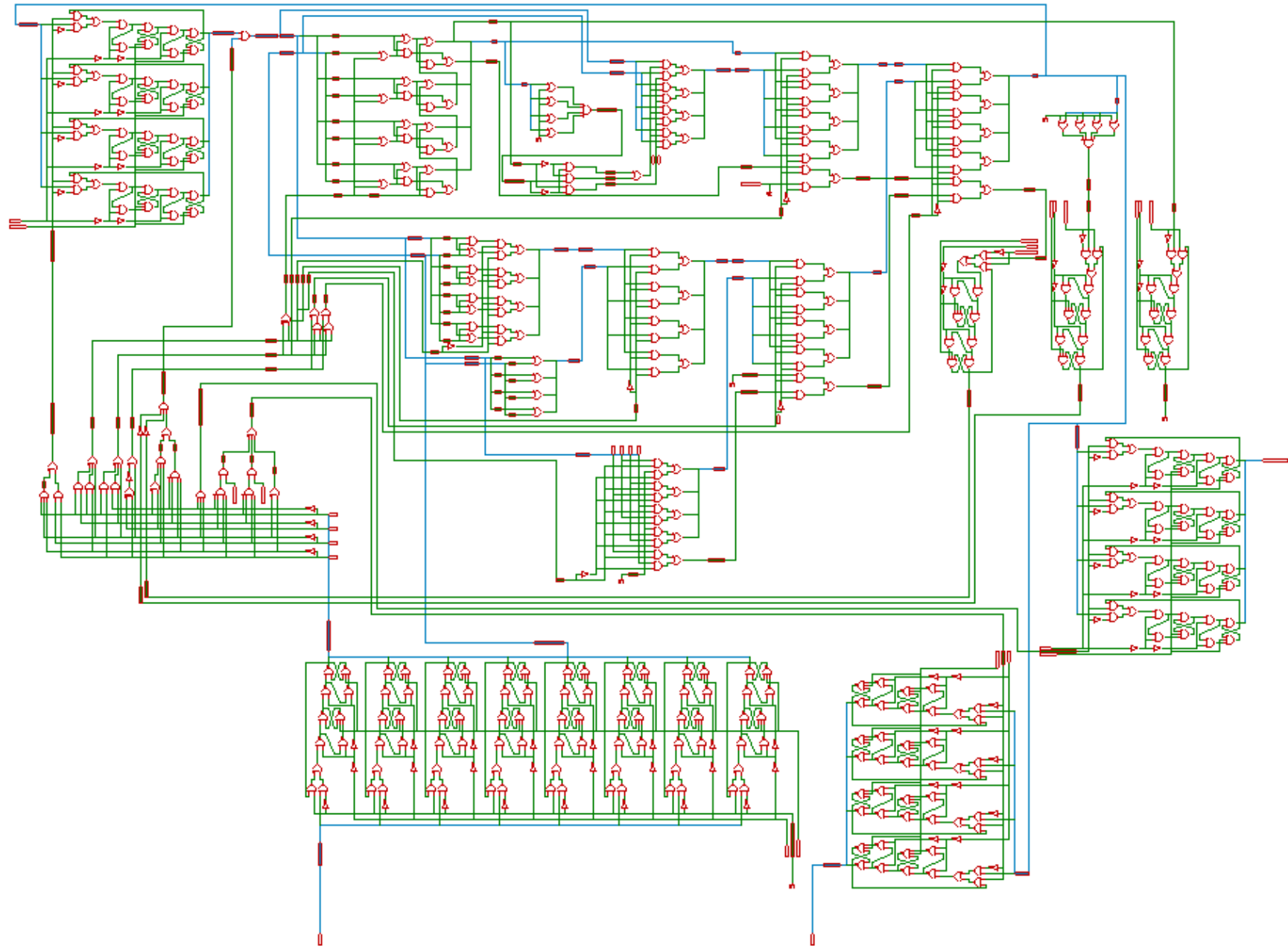
これ覚えてますか

# 俯瞰





# 俯瞰



# 名称

ALU:演算器

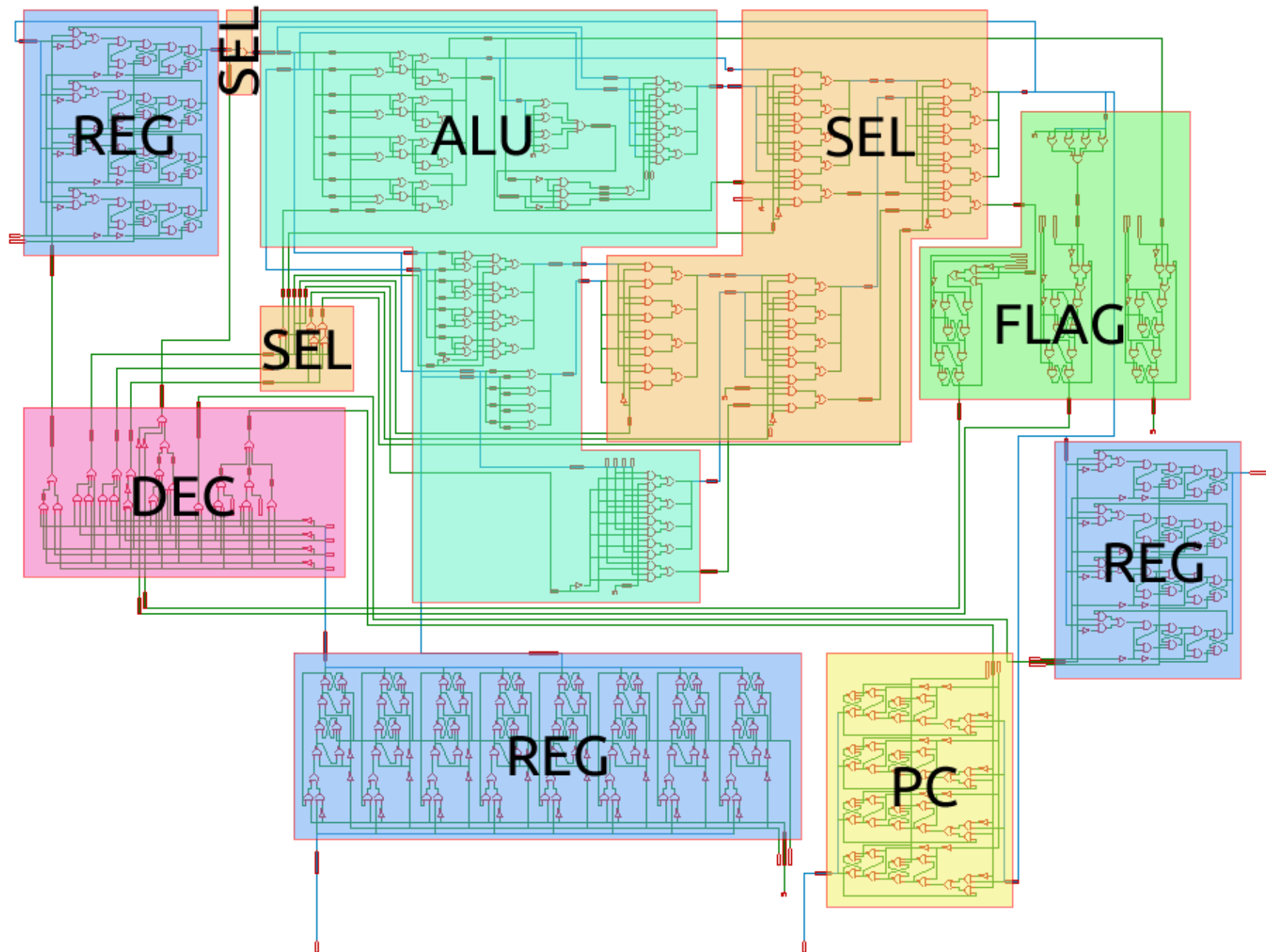
REG:レジスタ

FLAG:フラグ

PC:プログラム  
カウンタ

SEL:セクタ

DEC:デコーダ

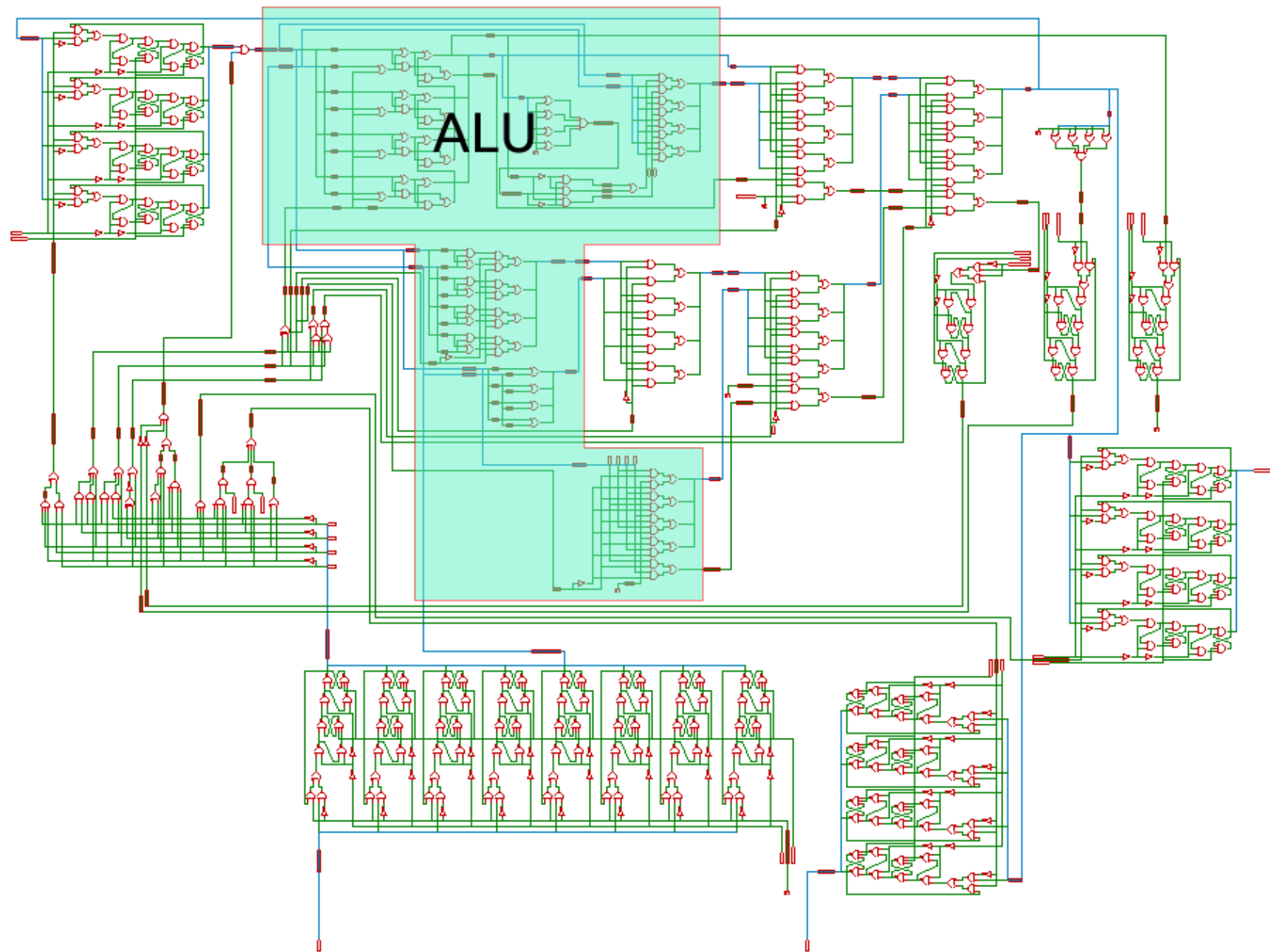


# 今日やること

- 4bitCPUを俯瞰しよう
- **演算器**
- メモリ
- PC(Program Counter)
- デコーダ
- セレクタ

# 演算器 (ALU)

俯瞰で言うと  
この辺



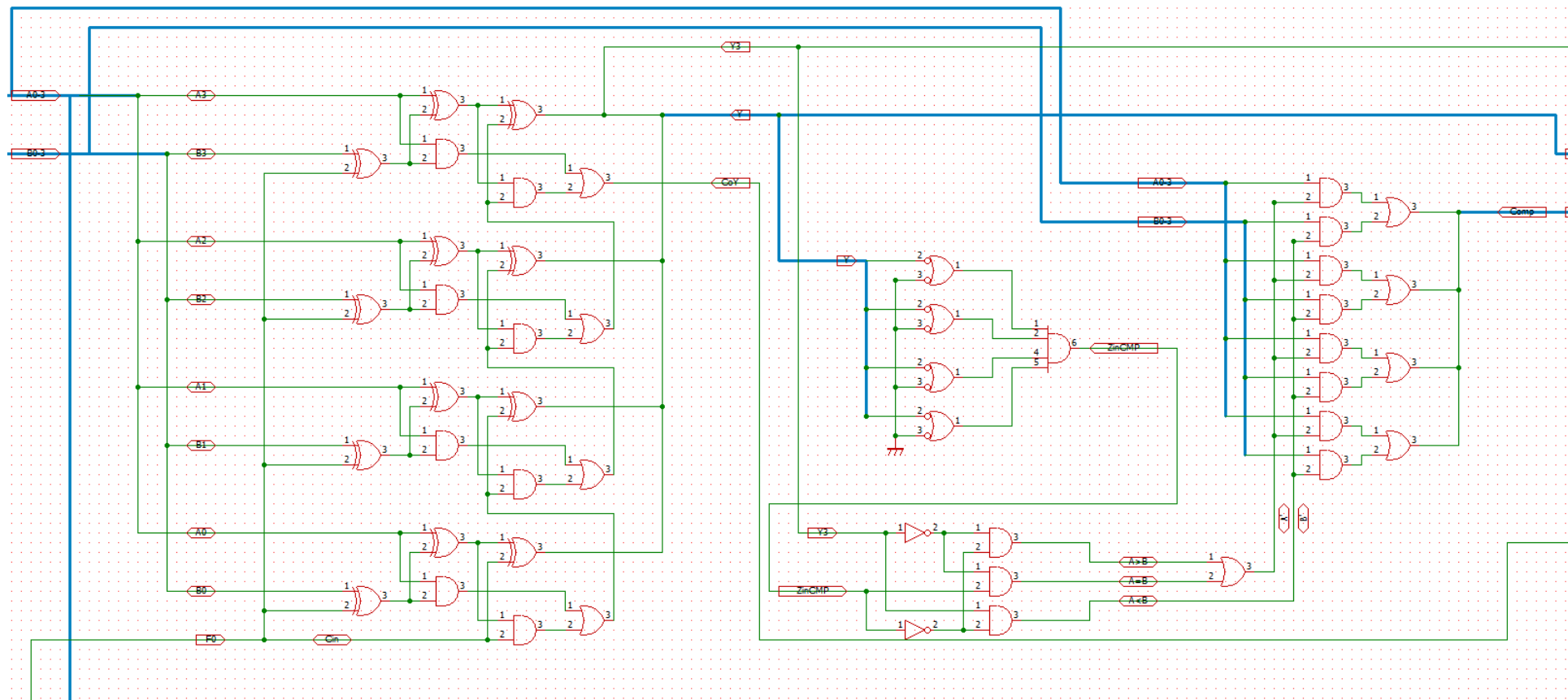
# 演算器(ALU)

- Arithmetic Unit(算術演算機)
  - 全加算器
  - 比較器
- Logic Unit(論理演算器)
  - and
  - or
  - xor
  
  - r-shift
  - l-shift

# 演算器 (算術)

左半分:  
全加算器

右半分:  
比較器



# 演算器(ALU)

- Arithmetic Unit(算術演算機)
  - 全加算器
  - 比較器
- Logic Unit(論理演算器)
  - and
  - or
  - xor
  
  - r-shift
  - l-shift

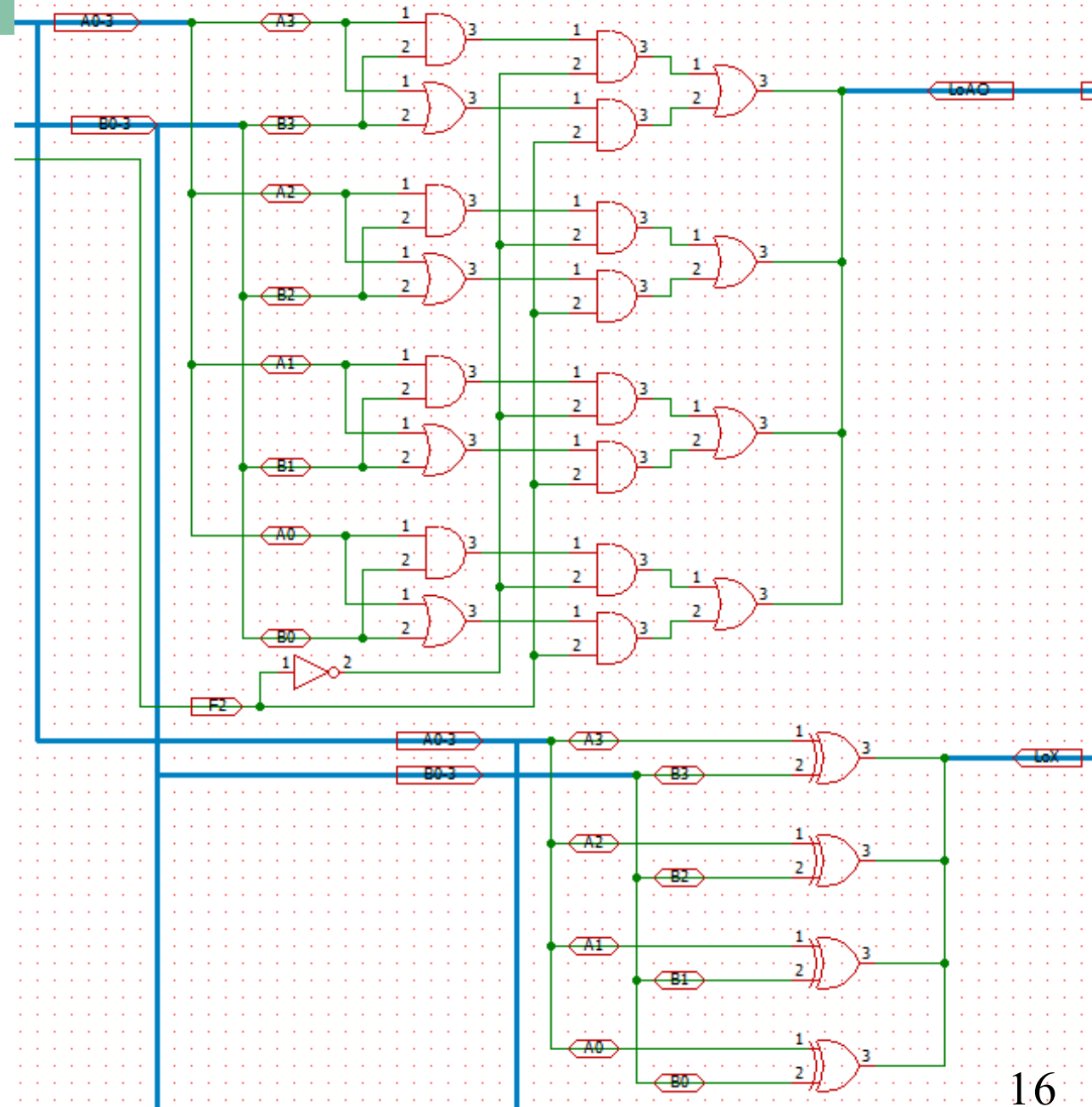
# 演算器(論理)

上半分:

- and
- or

下半分:

- xor

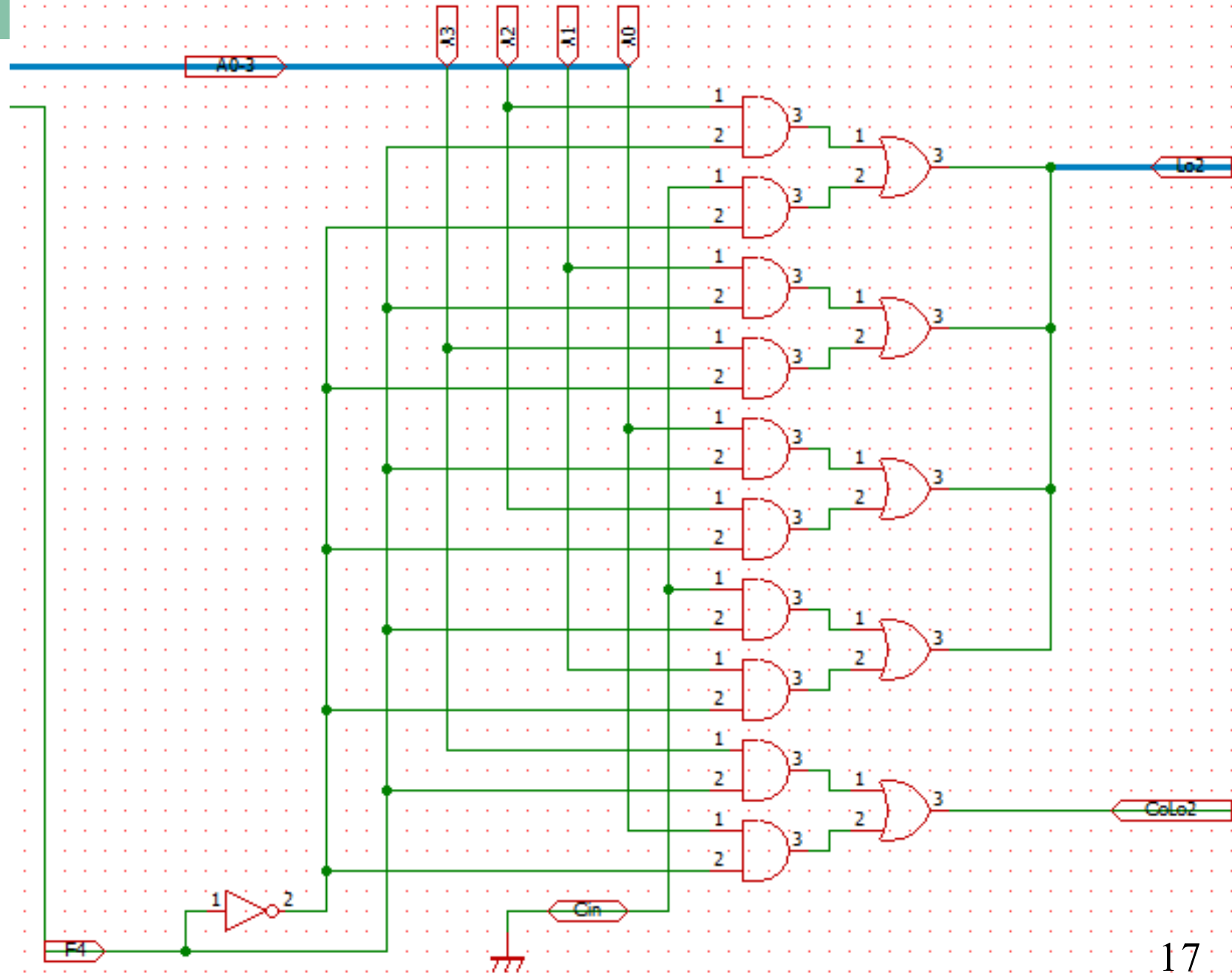




# 演算器(論理)

シフト演算:

- r-shift
- l-shift

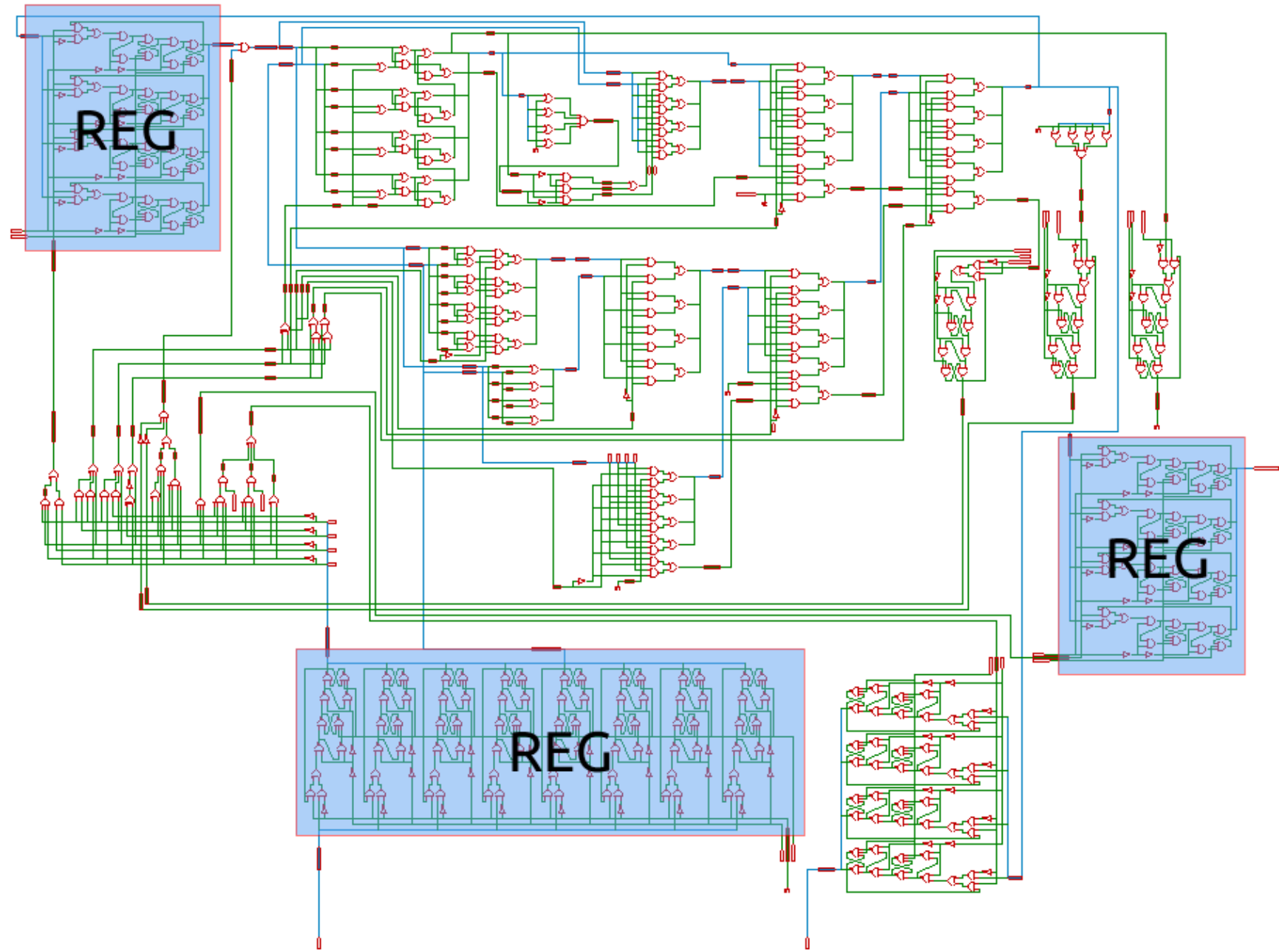


# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- **メモリ**
- PC(Program Counter)
- デコーダ
- セレクタ

# メモリ

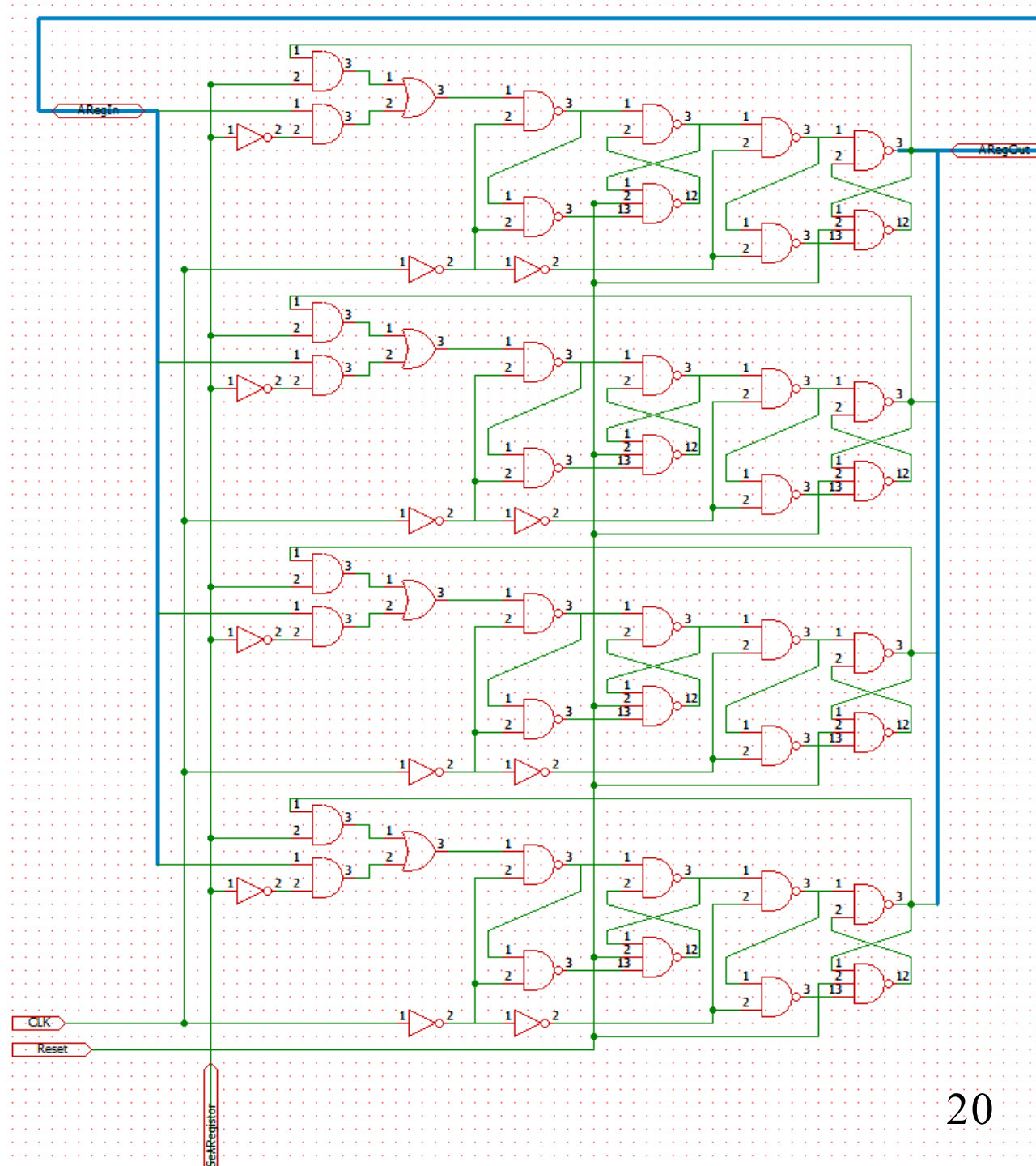
俯瞰で言うと  
この辺



# メモリ

アップで見るとこんな感じ

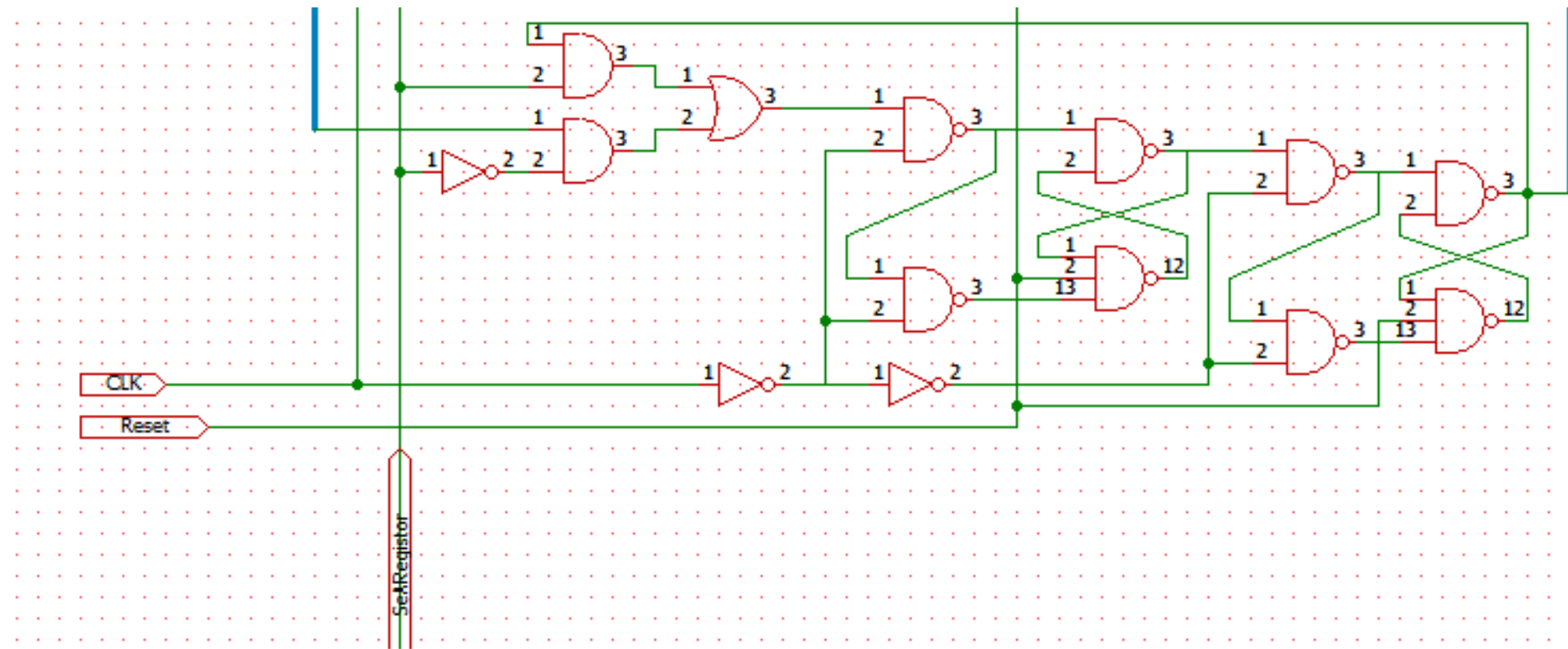
- 4bit



# メモリ

アップで見るとこんな感じ

- 1bit

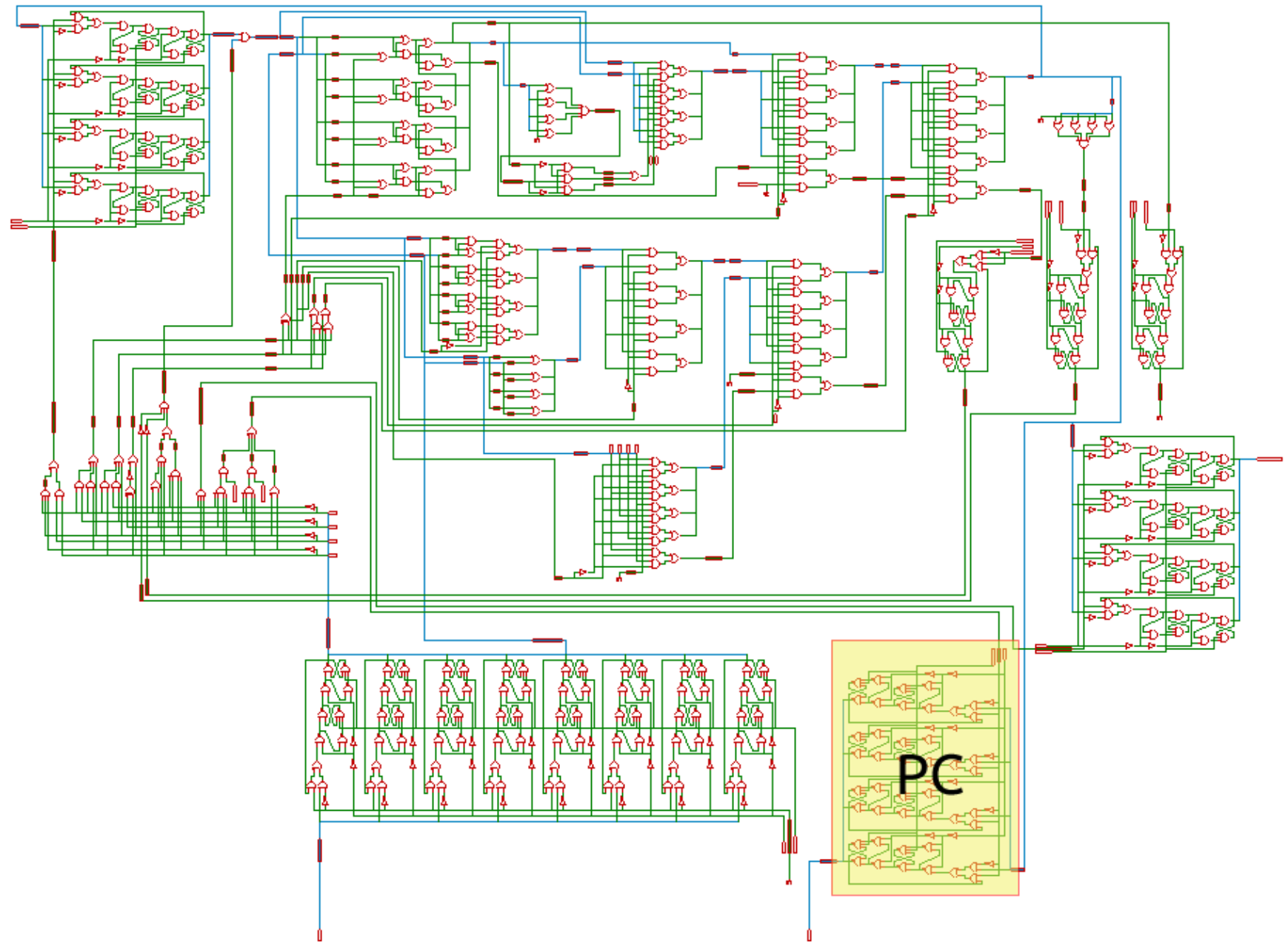


# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- **PC(Program Counter)**
- デコーダ
- セレクタ

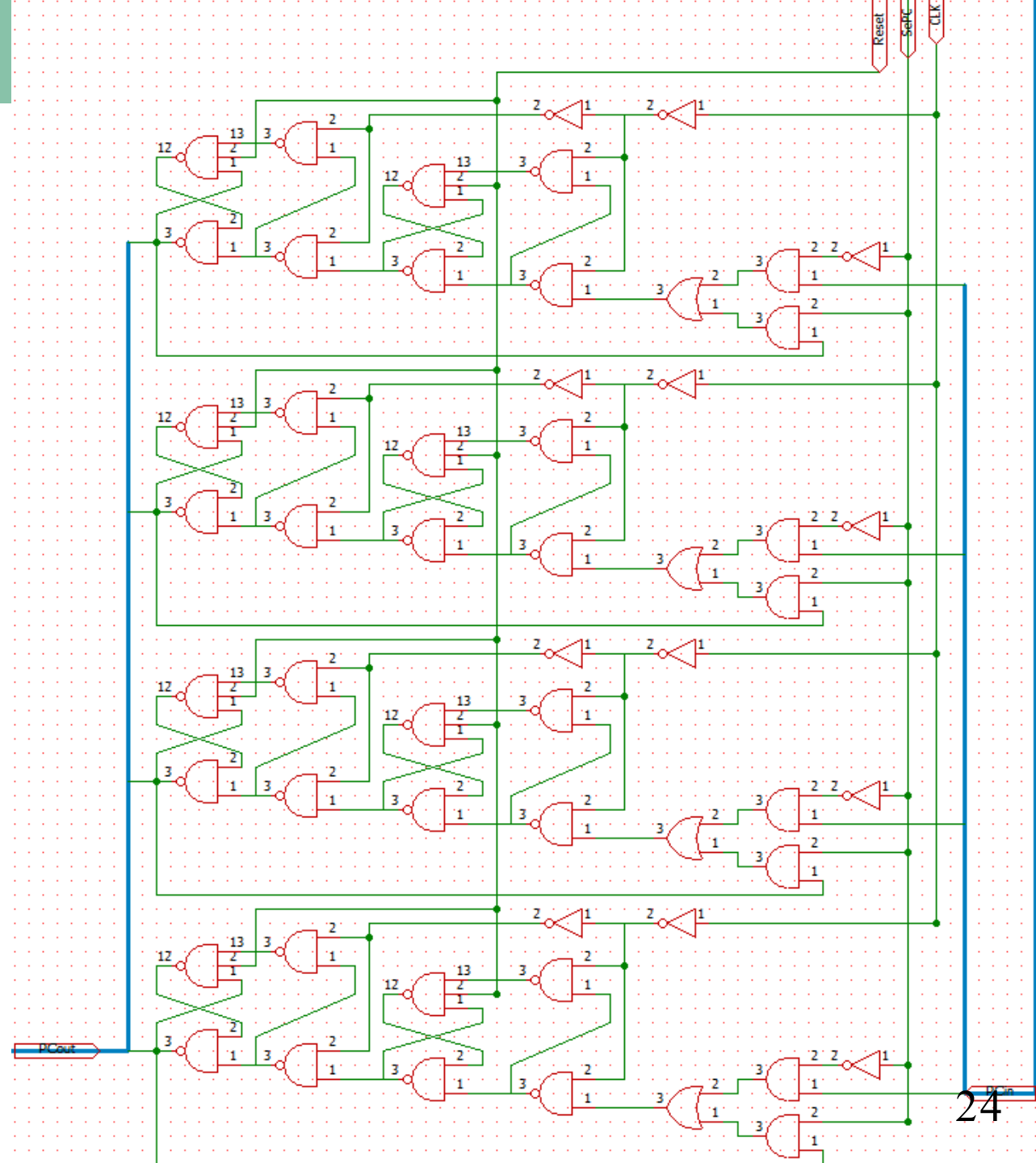
# PC

俯瞰で言うと  
この辺



# PC

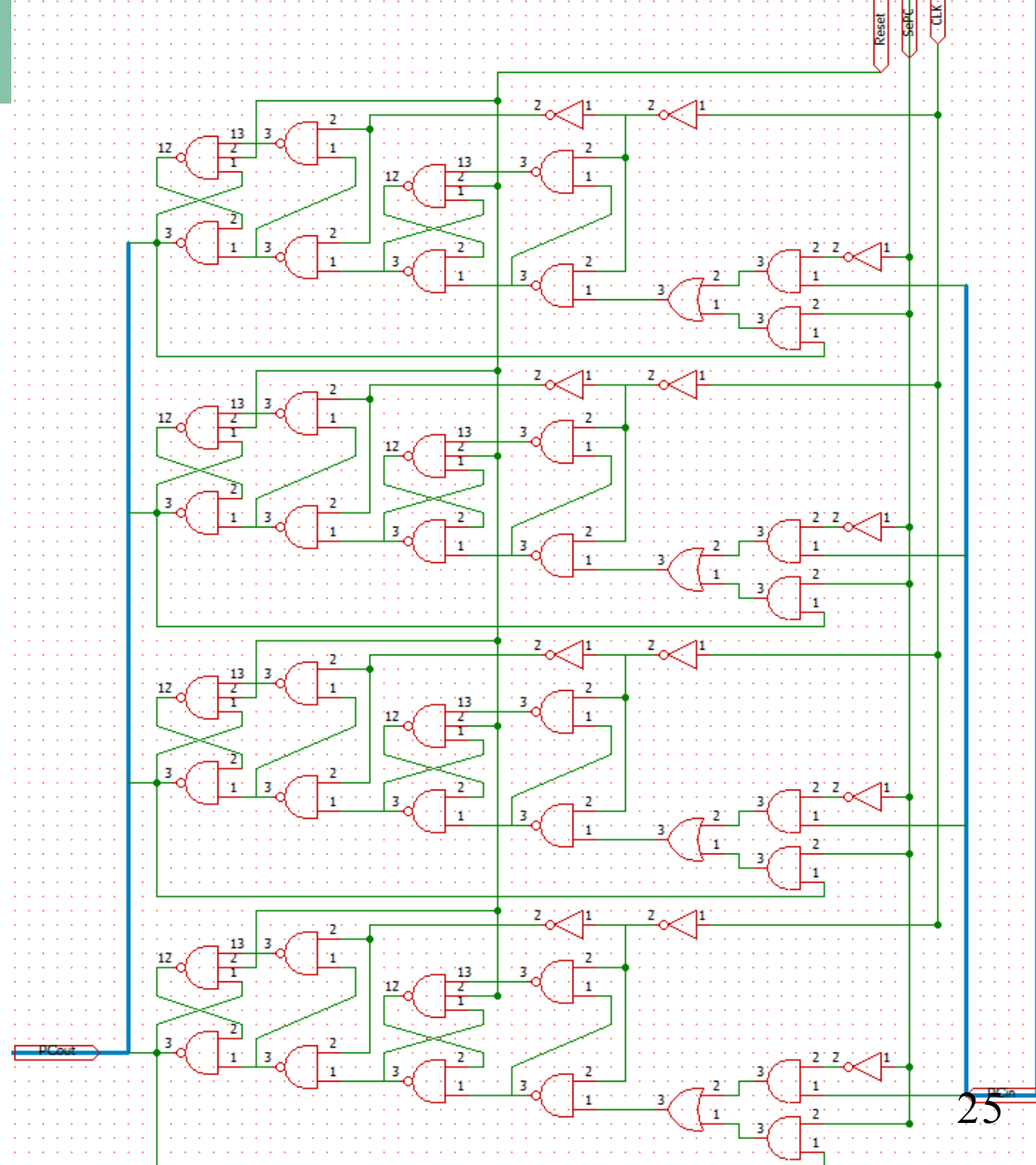
アップで見るとこんな感じ





# PC

皆さん気が付きましたか?

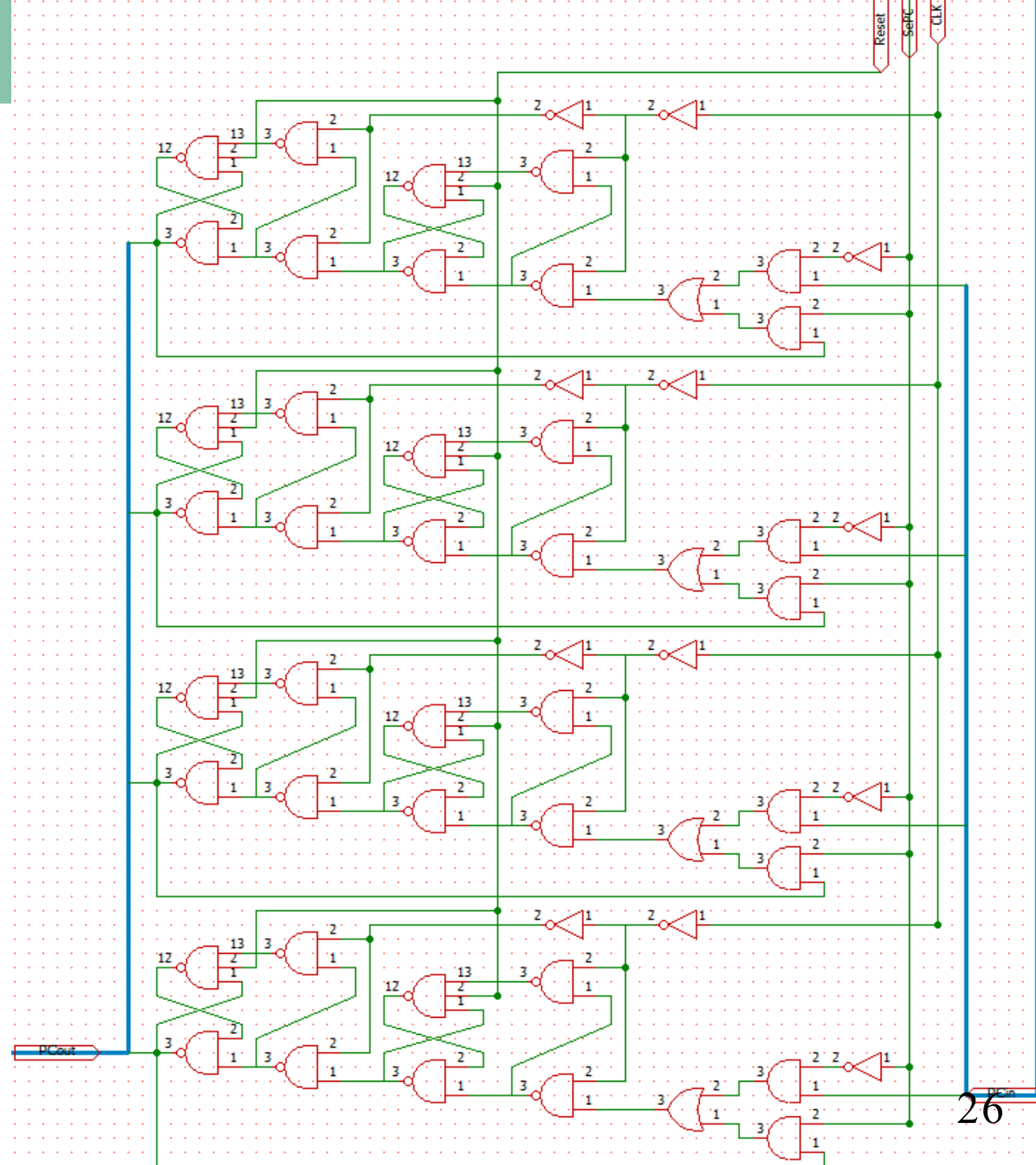


# PC

皆さん気が付きましたか?

Q1. PCとレジスタって同じ回路なんですか?

A1. いいえ



# PC

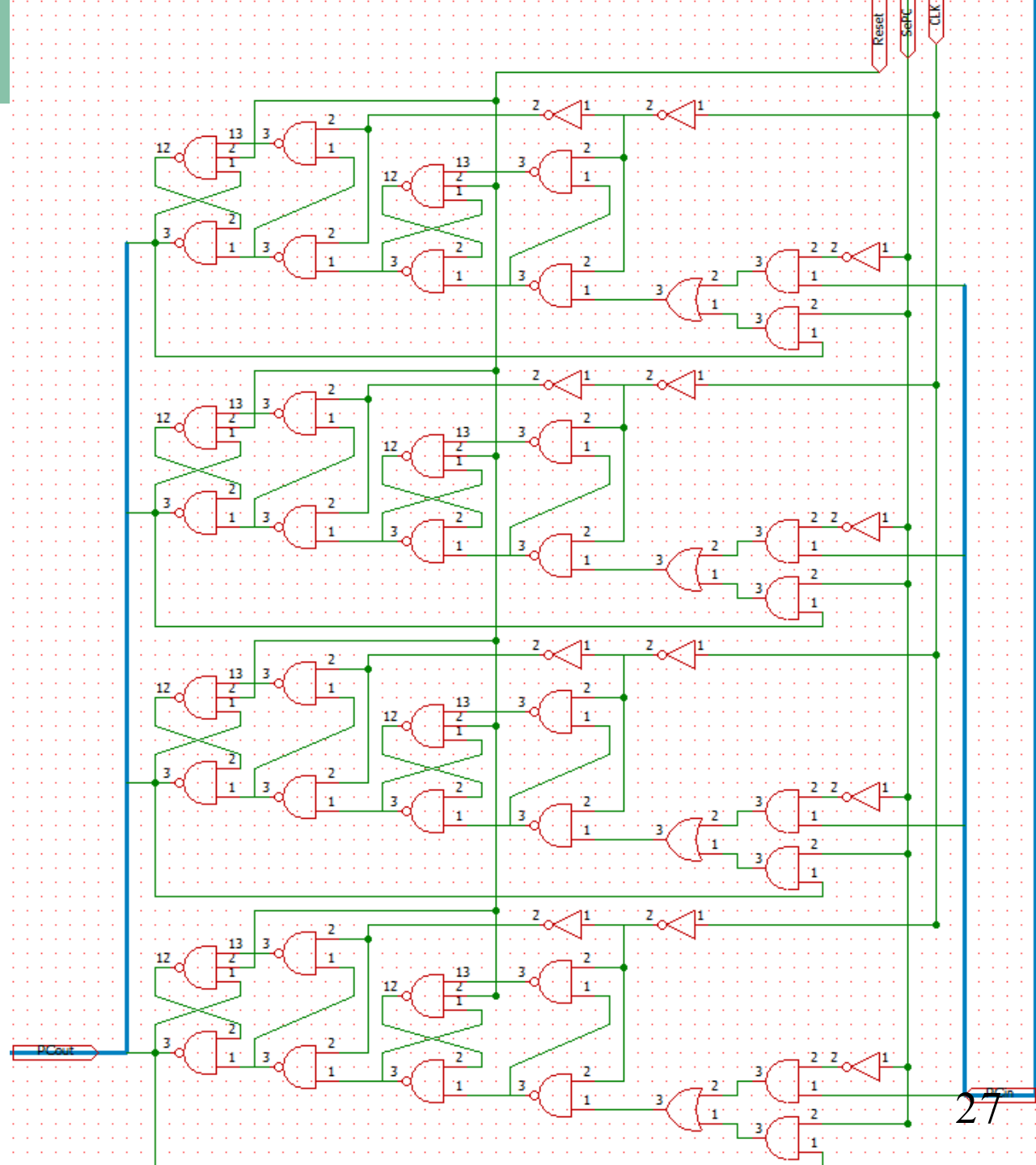
皆さん気が付きましたか？

Q1. PCとレジスタって同じ回路なんですか？

A1. いいえ

Q2. ただのレジスタになってるでは？

A2. Exactly



# PC

皆さん気が付きましたか？

Q1. PCとレジスタって同じ回路なんですか？

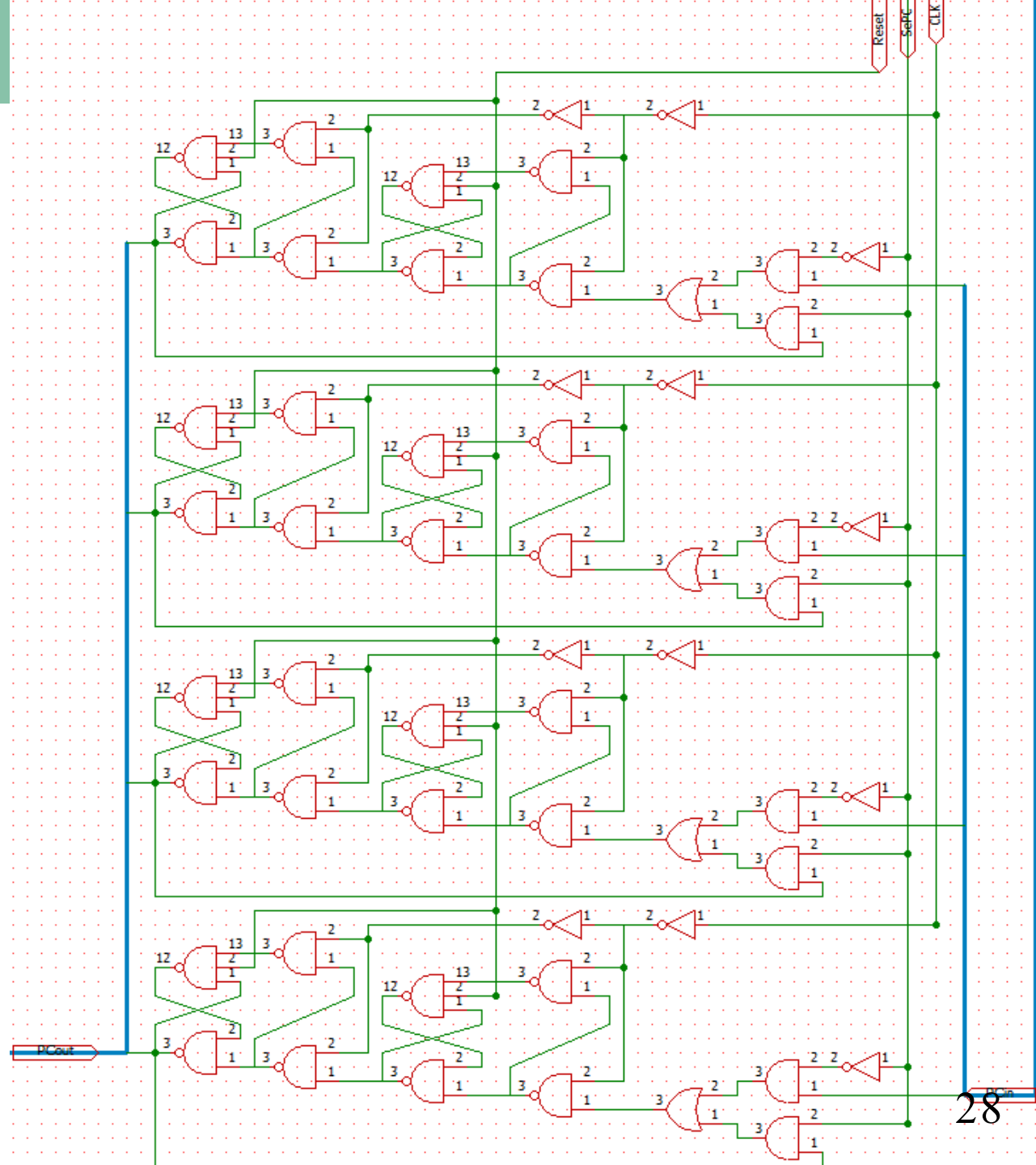
A1. いいえ

Q2. ただのレジスタになってるでは？

A2. Exactly

Q3. ミスった？

A3. 10年越しに気付いた



# PC

皆さん気が付きましたか?

Q1. PCとレジスタって同じ回路なんですか?

A1. いいえ

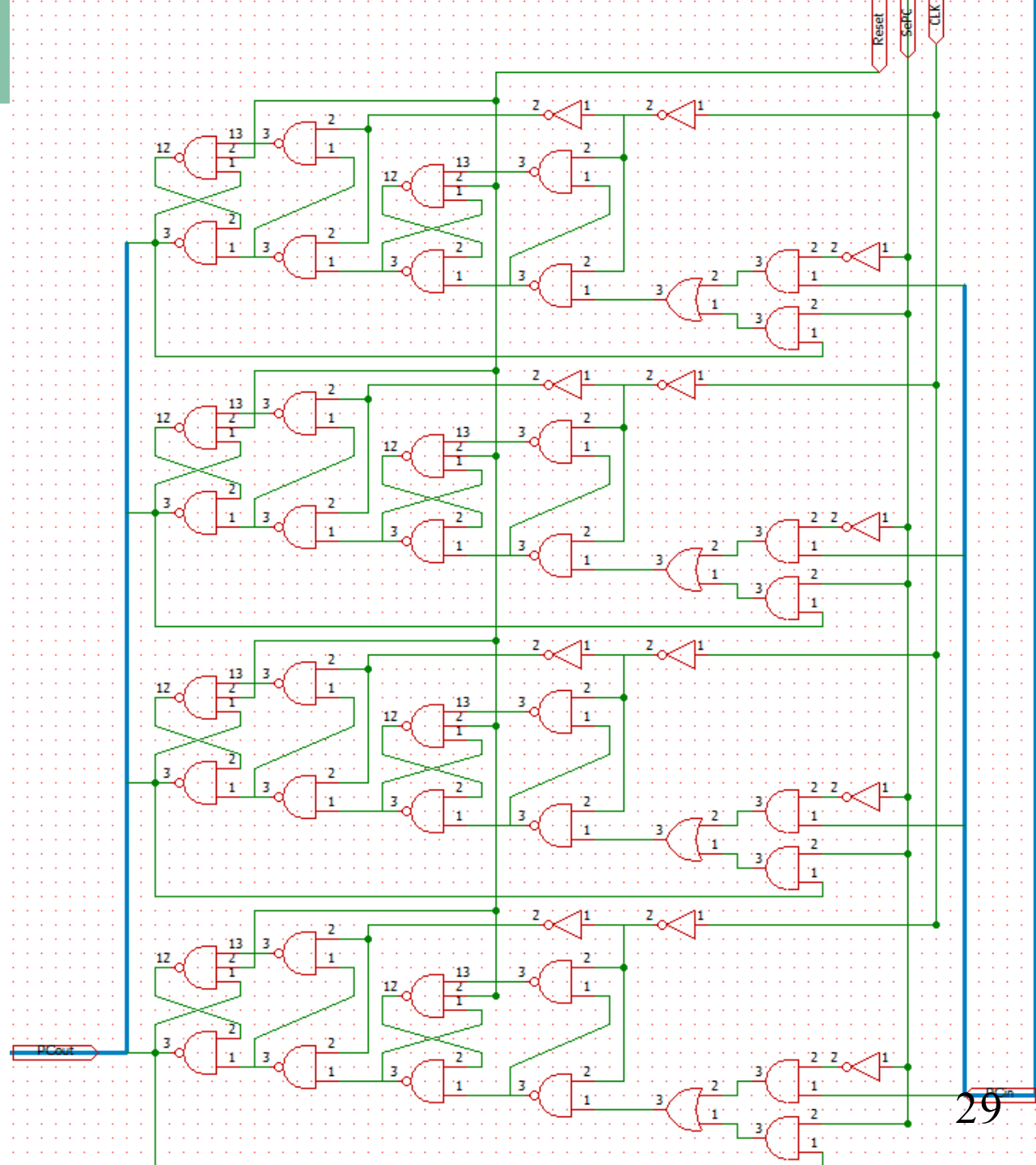
Q2. ただのレジスタになってるでは?

A2. Exactly

Q3. ミスった?

A3. 10年越しに気付いた

カウントアップ回路になってなくてただのレジスタになってる  
→ 後日作り直します



# 謝罪(元気があればやる)

母校のK先生へ。お元気ですか？

僕は10年前に提出した課題のミスを発見しました。

でも、もう単位と学位もらったので見なかったことにします。

どうかこのまま気づかないでください。

※(業務連絡)編集班の方へ、ここのボケの件カットしてください。

# 謝罪(元気があればやる)

母校のK先生へ。お元気ですか？

僕は10年前に提出した課題のミスを最近発見しました。

でも、もう単位と学位もらったので見なかったことにします。

どうかこのまま気づかないでください。

あと、自分の課題のレポート読み返してて恥ずかしくなってきたので、僕の班のレポートを課題のベースモデルとして後輩に見せびらかすのそろそろ止めてください。

※(業務連絡)編集班の方へ、ここのボケの件カットしてください。

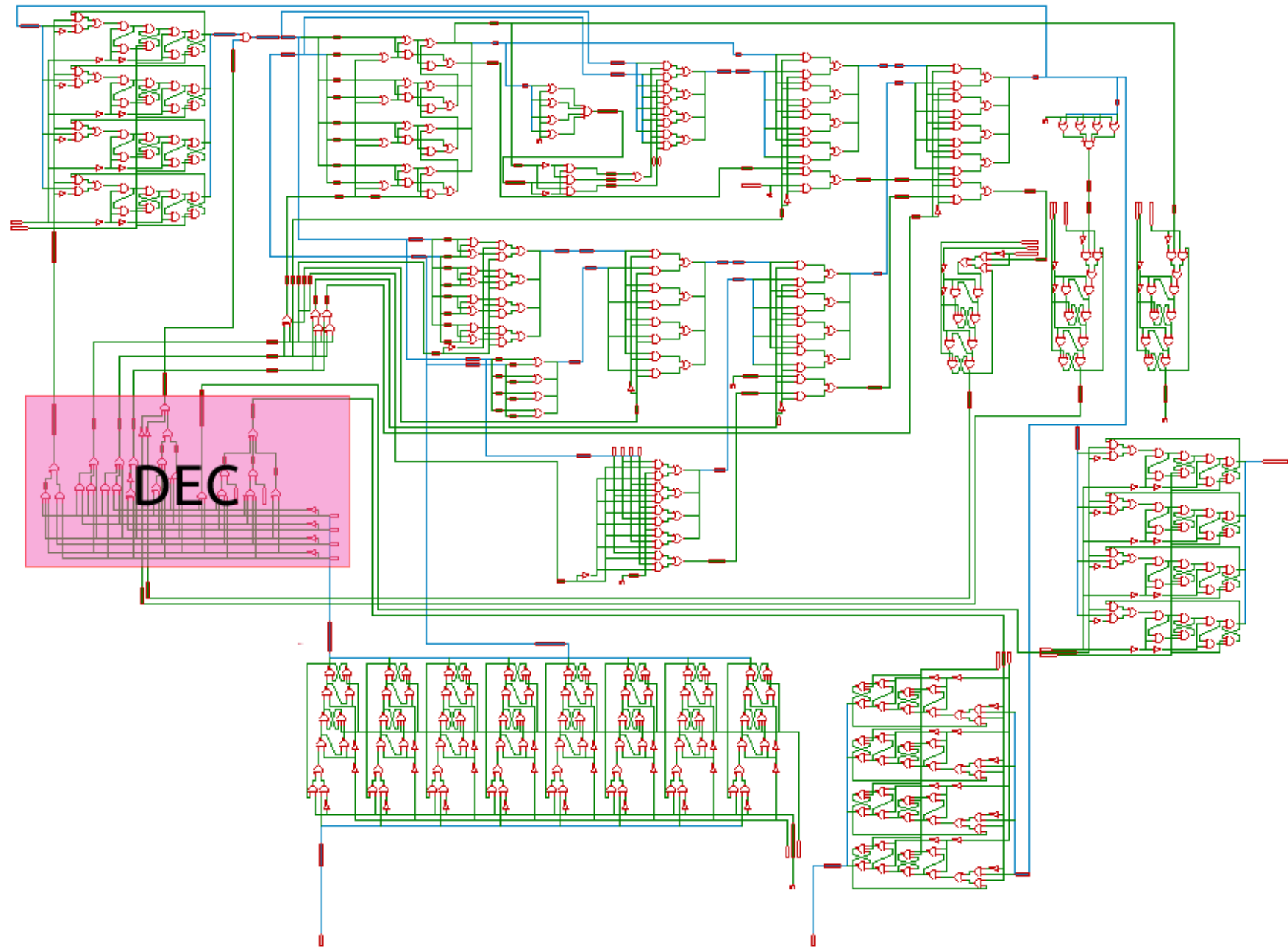
# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- PC(Program Counter)
- **デコーダ**
- セレクタ



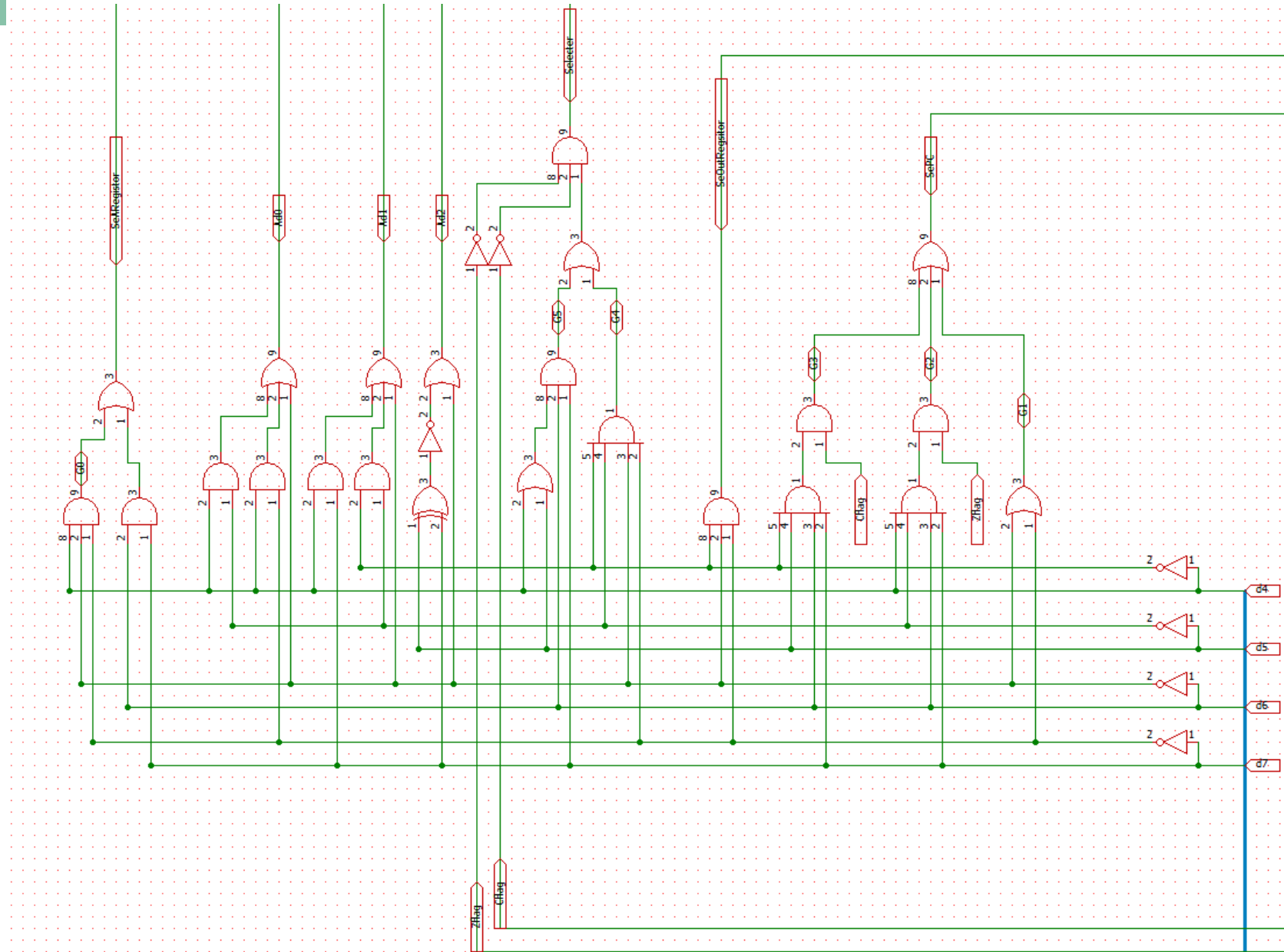
# デコーダ

俯瞰で言うと  
この辺



# デコーダ

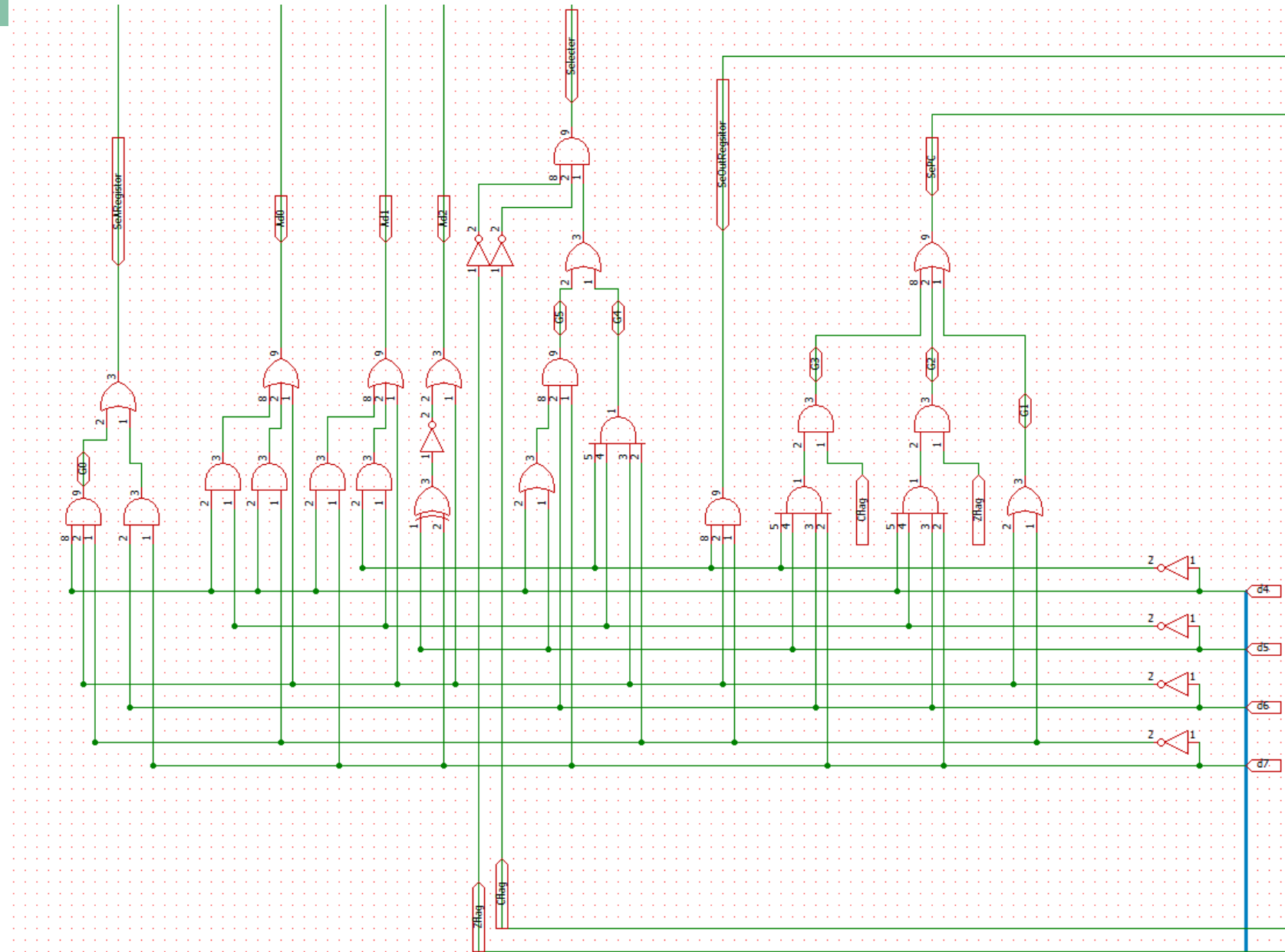
アップで見るとこんな感じ



# デコーダ

左から

- AレジスタWrite
- ALUセクタ\*3
- #0 出力選択  
(Aレジスタ  $\overline{Read}$ )
- Outレジスタ Write
- PC Write

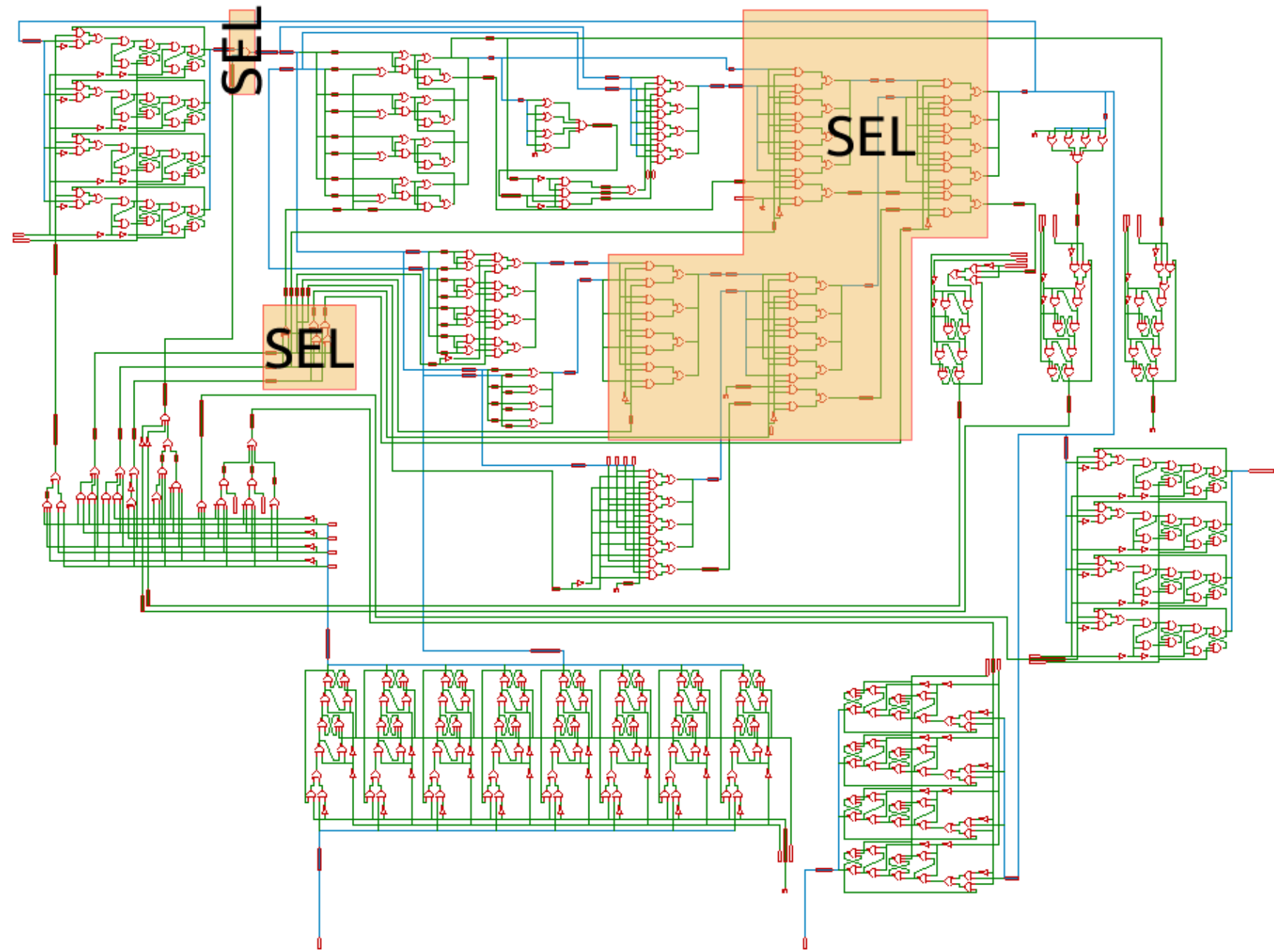


# 今日やること

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- PC(Program Counter)
- デコーダ
- **セレクタ**

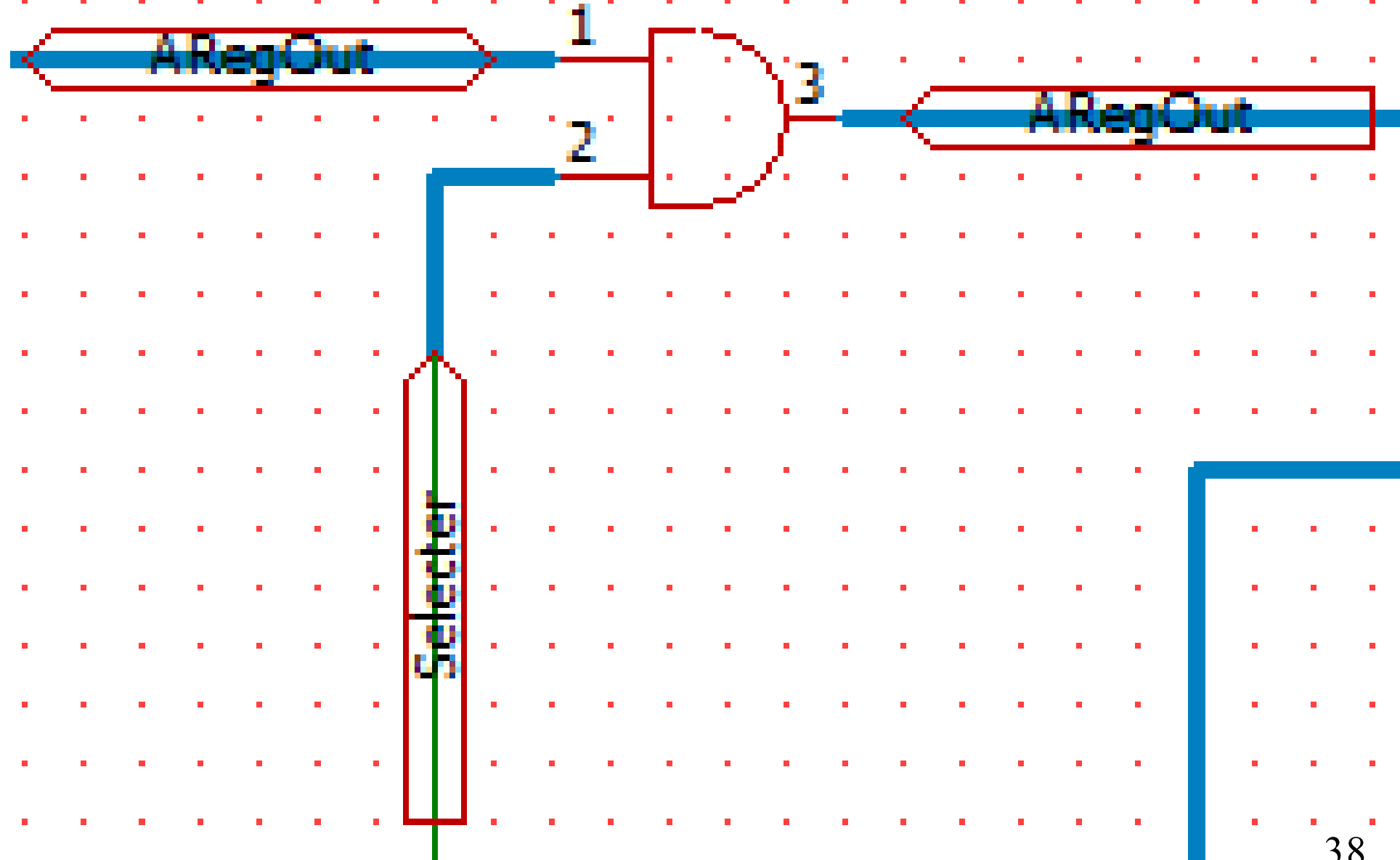
# セレクタ

俯瞰で言うと  
この辺



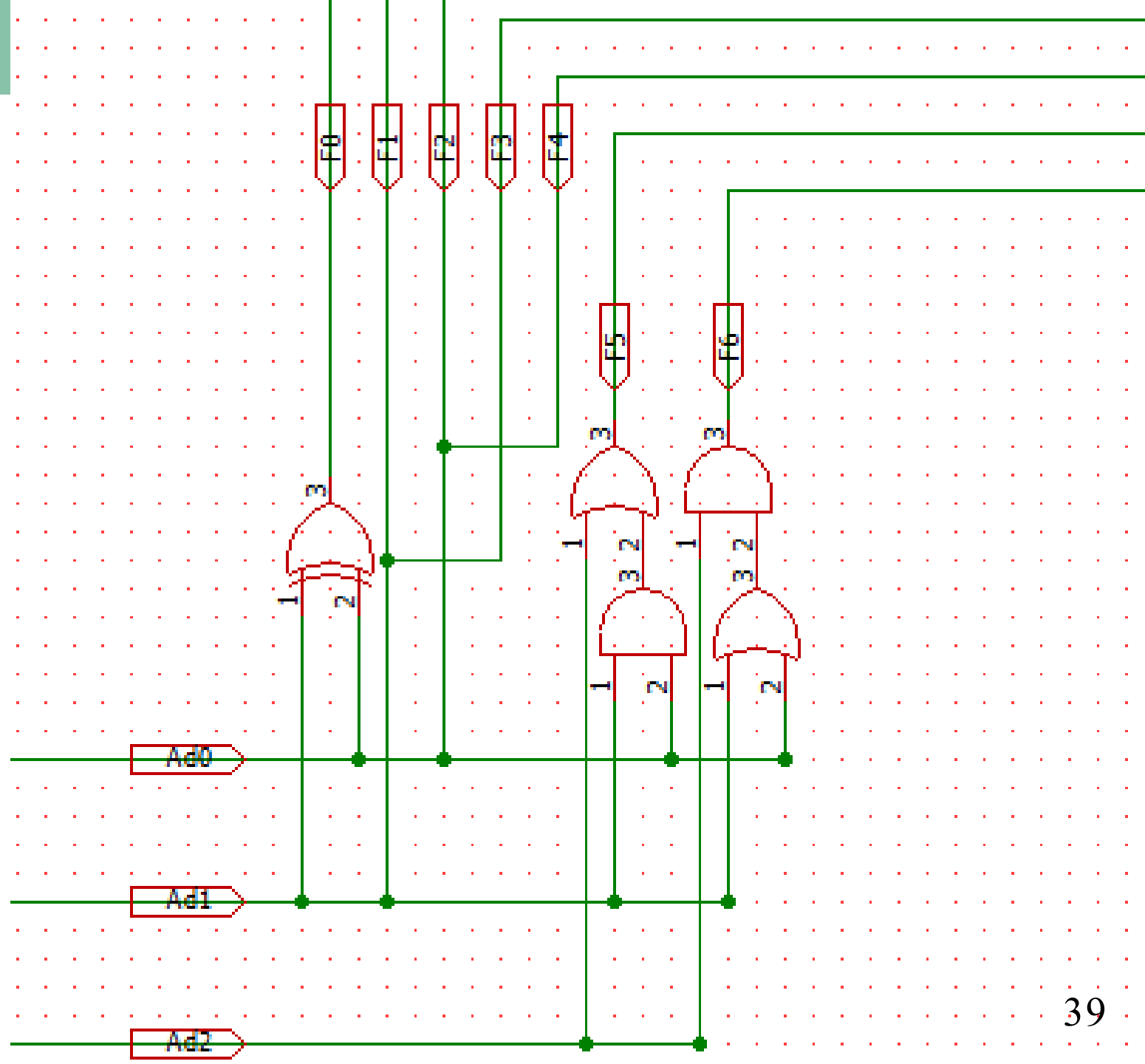
# セレクタ

Aレジスタ



# セレクタ

演算選択

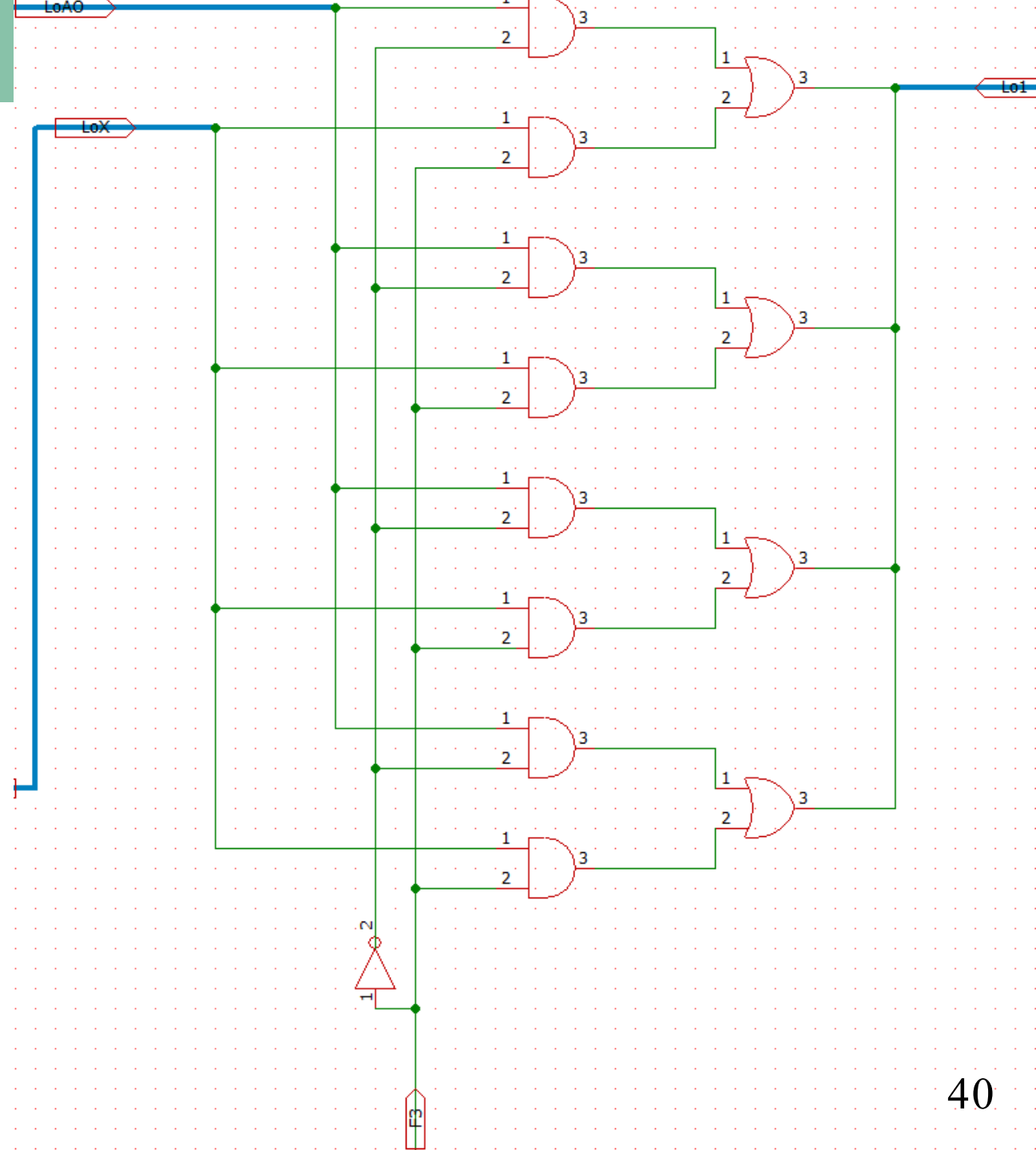


# セレクタ

## 出力する演算の選択

このアーキテクチャはすべての算術論理演算を並列に行い、デコーダの情報を元に出力したい結果だけ選択的に取り出してる。

→ 入力部分で封をしたり、使わない演算器はDisableにすることで消費電力を削減できる可能性





# 今日やったこと

- 4bitCPUを俯瞰しよう
- 演算器
- メモリ
- PC(Program Counter)
- デコーダ
- セレクタ

# 次回以降やること

- 全加算器で減算できるようにする
- シフト回路による乗算除算