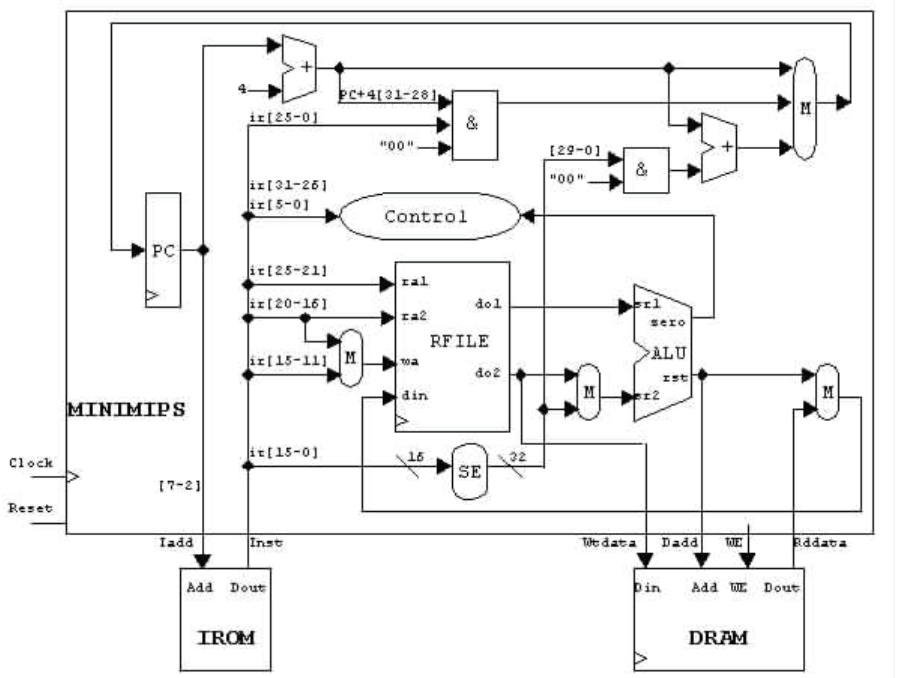
簡単な4ビットALUの設計

ALUとは何か

- ALU とは Arithmetic Logic Unit で略で、算術論理演算器とも言われる。
- 通常 ALU はマイクロプロセッサの内部で使用され、データに対して加算、引き算、論理和、論理積などの種々の算術論理演算を行う。
- 基本的に ALU は単なる組み合わせ回路である。
- 下記図は Nintendo64 や PlayStation そして多数のレーザプリンター等で用いられている RISC 型 MPU の MIPS プロセッサを単純にしたものであり、図の右下に ALU がある。



今回、以下の仕様のALUの設計方法をみる。

- 入力および出力信号

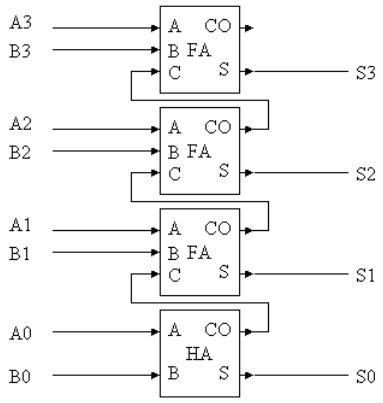
入力信号	4ビットデータ入力１	A=(A3,A2,A1,A0)
	4ビットデータ入力２	B=(B3,B2,B1,B0)
	5ビット制御信号	F=(F4,F3,F2,F1,F0)
出力信号	4ビットデータ出力	Y=(Y3,Y2,Y1,Y0)

- 機能

制御信号					動作	説明
F4	F3	F2	F1	F0		
0	0	0	1	0	Y <= A + B	加算
0	0	0	1	1	Y <= A − B	減算、B の2の補数と A の加算で実現する
0	1	0	0	0	Y <= A and B	各桁ごとに and 演算を取る
0	1	1	0	0	Y <= A or B	各桁ごとに or 演算を取る
0	0	0	0	0	Y <= shl A	A を左へ（上位へ）1ビットシフト
1	0	0	0	0	Y <= shr A	A を右へ（下位へ）1ビットシフト

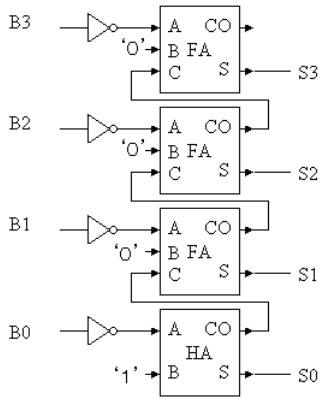
加算器はどう作るか？

- 各桁の加算で桁上げを考慮して、半加算器と全加算器で作れる。
- このような加算器をリップルキャリー型加算器という。（CO 出力がすべての FA を伝わってゆくので）
- リップルとは水面を伝わる波

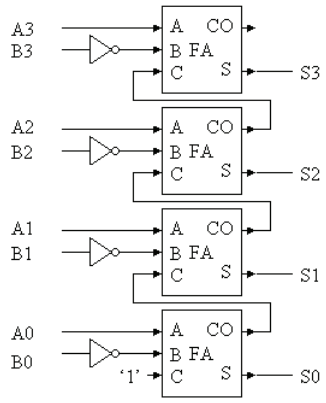


減算器は加算器を用いてどうつくるか？

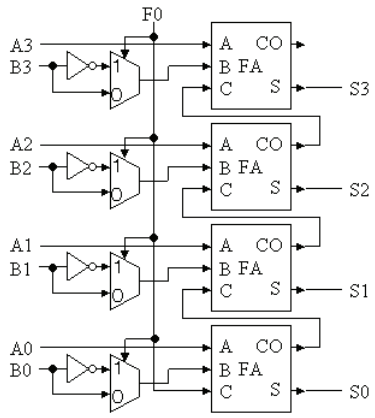
- A−B なる引き算は A と B の2の補数の加算であった。（思い出すべし！！！！）
- したがって、2の補数器と加算器で減算器が作れる。
- 2の補数＝1の補数＋1であった。（思い出すべし！！！！）
- 1の補数は各桁を反転したもの
- 2の補数器は以下のようになる。



- 上記2の補数器の出力（S3,S2,S1,S0）を加算器の（B3,B2,B1,B0）入力へほりこめば、減算器ができる。すなわち、HA2個、FA6個を用いて作れる。
- しかし、ちょっと賢い人なら、以下のようにできることに気が付く。この場合、FA4個である。
  なぜなら、引き算 Y <= A − B = A + B' +1 だから

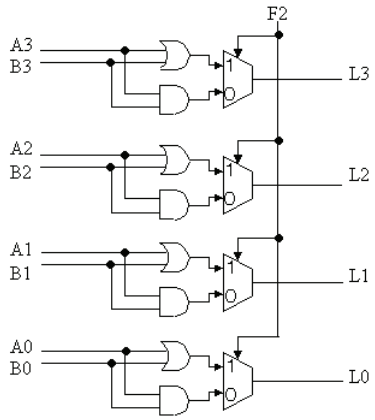


したがって、加減算器はマルチプレクサを用いて以下のようになる。

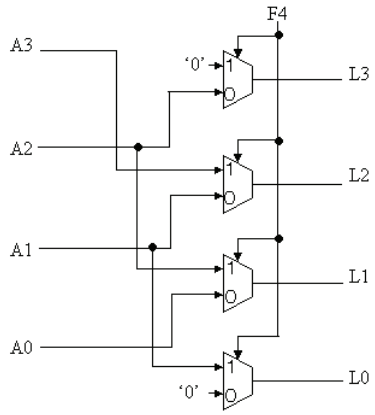


各桁ごとにANDもしくはORをとる回路はどうなるか？

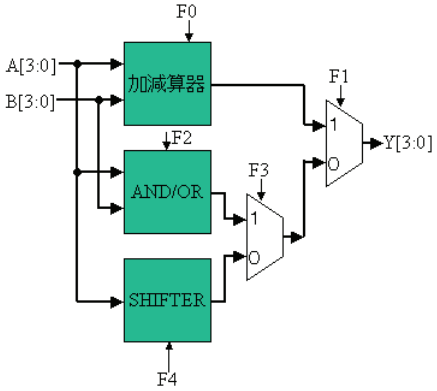
- これは非常に簡単で



シフト動作はどう実現するか？



全体をまとめて、ALUを作ると



- ここで太めの線は複数の信号の束（4ビットのバス配線）を意味する。

- **不思議なことに、今まで勉強してきた、論理ゲート（AND,OR,NAND,NOR,NOT）や、それを用いて作った基本的な組み合わせ回路（半加算器、全加算器、マルチプレクサなど）を用いるだけで、割と大きなALUが設計できてしまう。**
- **もっと複雑な回路も結局、このように、いろんな機能の組み合わせで作ることができる。**

ここまでが、中間テストの範囲です。教科書、ノートなどの持込は OK です。

以上