Project #2

1 ALU Architecture

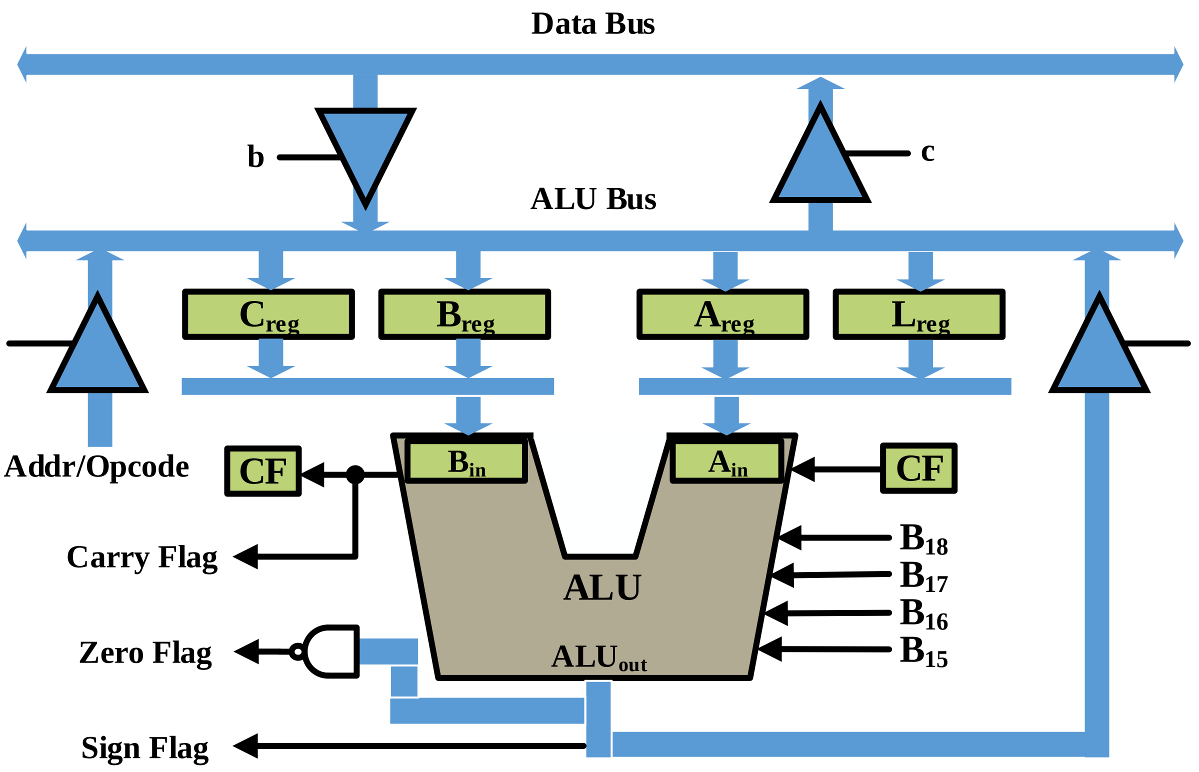

Fig 2.1: ALU Architecture

Table 2.1: Microcode layout

23~22	21~19	18~15	14~13	12	11~9	8	7~0
ALU dest	ALU src	ALU ops	CF	reg in	BOP	DB	Branch Addr.
A(default) B C L	A(default) B C L AB AC LB LC	ADD SUB AND OR XOR PASS(def) NOT SHLCF SHRCF INCR DECR CLR	NC SET CLR Carry	ALU DB	Next(def) BRA BRDEN BROP BRZ BRP BRC BRCF	IN OUT	

Table 2.2: Microcode Field Definition

Bit	Definition
Bit 23-22	ALU A, B, C or L register. Default is register A.
Bit 21-19	ALU input definitions, it must correspond with ALU unary or binary operations. For unary operation sources are A, B, C or L For binary operation sources are AB, AC, LB, LC
Bit 18-15	ALU operation definitions: ADD -- Ain plus Bin Plus CF; SUB -- Ain minus Bin minus Borrow; Borrow=not CF; AND -- Ain logical and with Bin; logical means bit by bit ops. OR – Ain logical or with Bin; XOR – Ain logical xor with Bin; For the unary ops ALUin = Ain or Bin; PASS – ALUin is pass through; This is the default. NOT – ALUin is complemented; SLCF – Rotate ALUin and CF left; SRCF -- Rotate ALUin and CF right; INCR – Increase ALUin DECR – Decrease ALUin CLR – Clear ALUin; This is used for clearing the registers.

Bit 14-13	Carry Flipflop operations;
	NC – No change;
	SET – Set flipflop;
	CLR – Clear flipflop;
	CRY – Loads ALU carry out;

Bit 12	Controls the input to the registers. ALU – ALU output goes to the dest reg. DB – Data bus goes to the dest reg.

Bit 11-9	Controls the microprogrammed sequencing.
	NXT – Next instruction;
	BRA – Branch Always;
	BRDBN – Branch on Data Bus Not Ready;
	-- Waiting for the data bus to be ready,
	-- Use this branch to itself while waiting and go to next instruction.
	BROP – Branch of Opcode Ready;
	-- Waiting for the opcode,
	-- Use this branch to itself while waiting and go directly to the opcode.
	BRZ – Branch on ALU out = zero;
	BRP – Branch on ALU msb = 0;
	BRC – Branch on ALU carry out;
	BRCF – Branch on Carry Flipflop = 1;

Bit 8	Controls the Data bus; IN – input from data bus; OUT – Output to data bus;
Bit 7-0	Branch address;

Table 2.3: Data direction

Bit 12	Bit 8	Tri-state Enables				Direction
Reg in	DB	a	b	c	d	
ALU	IN	disable	disable	disable	enable	ALU → Reg.
ALU	OUT	disable	disable	enable	enable	ALU → DB
DB	IN	disable	enable	disable	disable	DB → Reg.
DB	OUT	enable	disable	disable	disable	Op → Reg.

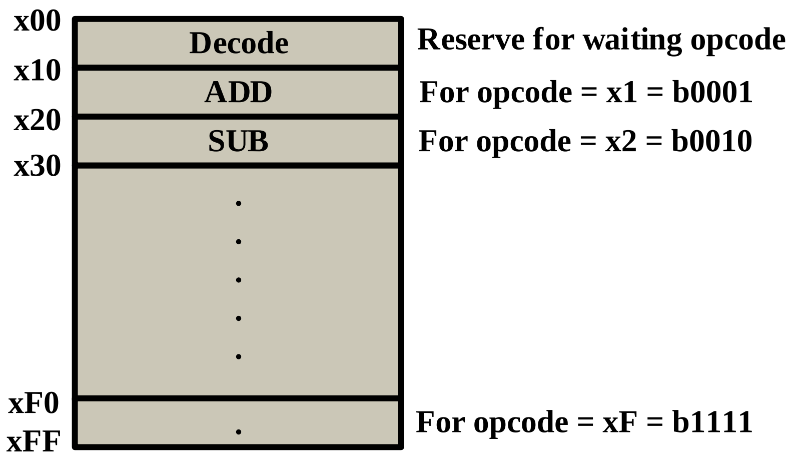

Fig. 2.2 Branch address

Table 2.4: Opcode Micro Programming

Arithmetic Ops	
ADD	Signed, Unsigned, 16-bit, 32-bit
SUB	
MULT	
DIV	
INCR	
DECR	
CLR	
Logical Ops	
AND	16-bit, 32-bit
NAND	
OR	
NOR	
NOT	
XOR	
XNOR	
COMP	
Shift Ops	
Logical	16-bit, 32-bit
Arithmetic	Signed, Unsigned, 16-bit, 32-bit
Rotate	16-bit, 32-bit
Memory Ops	
Read	16-bit, 32-bit
Write	

2 ALU Design Hints

ALU design is a parallel of all the alu operations required with tri-stated output to the alu bus. The tri-state driver is enabled by the alu instruction decoder of bits 18-15.

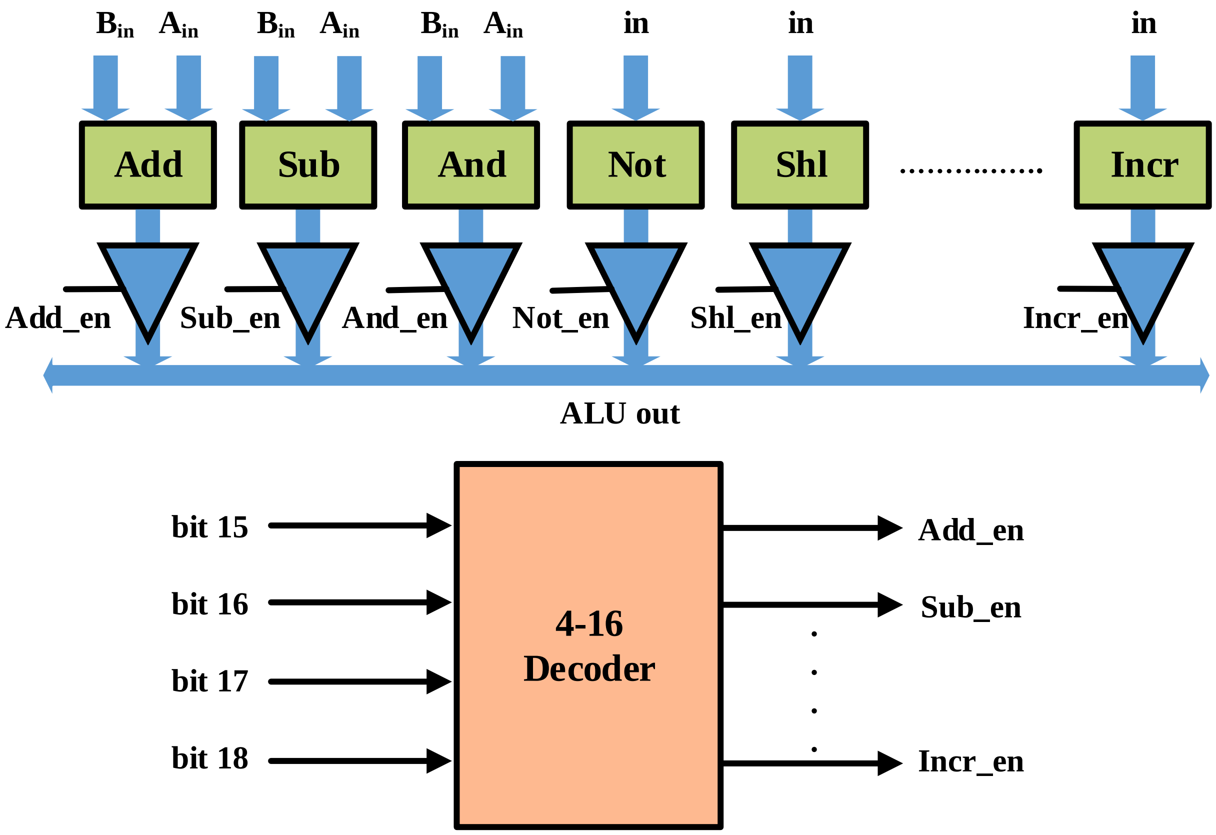

Fig. 2.3 ALU design

The shift left, shift right and pass operations requires no logics except wires.

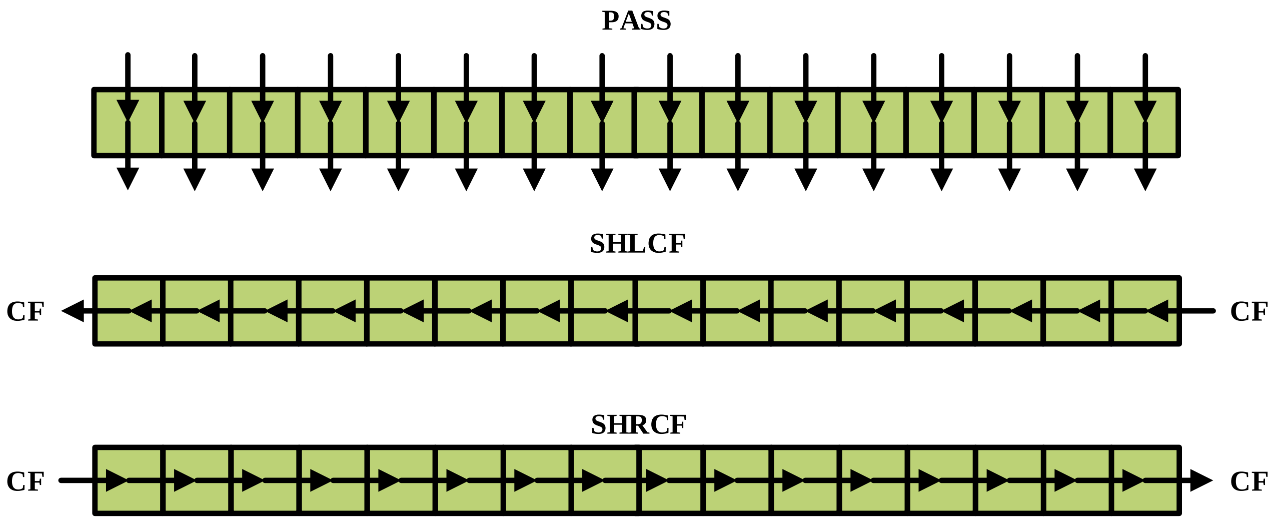

Fig. 2.4 SHLCF, SHRCF and PASS operations

The carry in and carry out are required on all operations except the logical operations and the carry flipflop will be set according to the CF ops (bit 14-13).

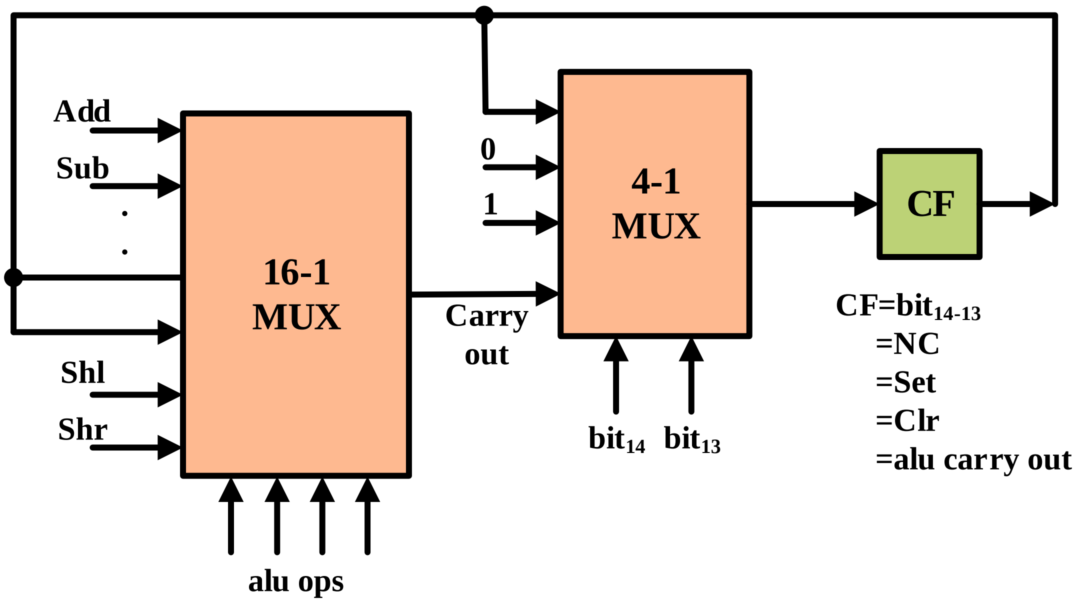

Fig. 2.5 CF operations

The Ain, Bin, and in are output of MUX depending on alu src (bit 21-19)

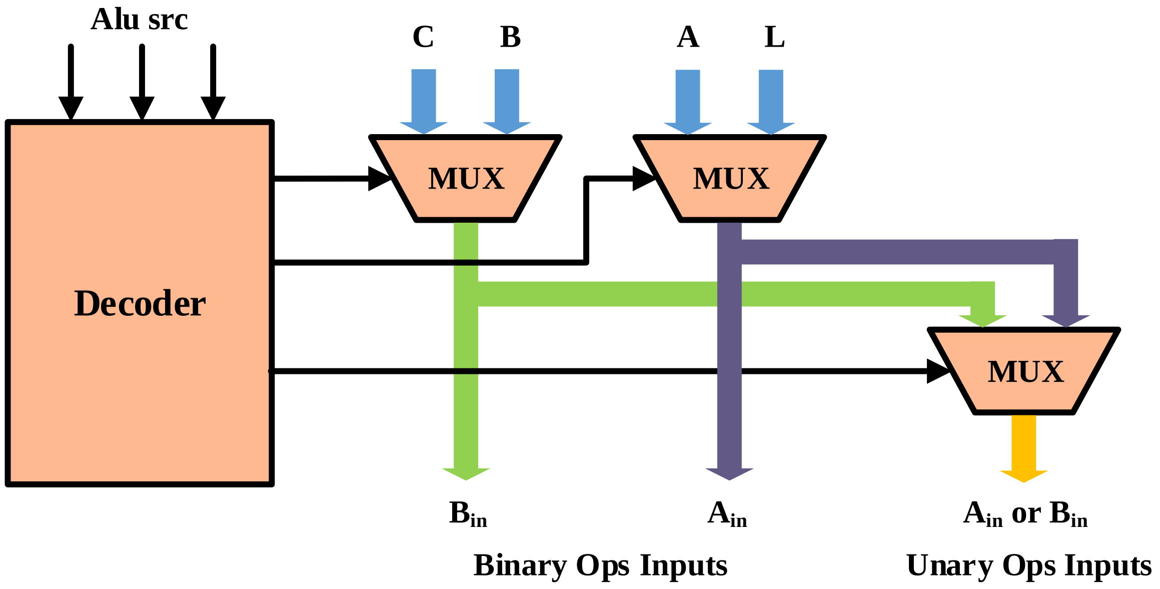

Fig. 2.6 Reg to ALU operations

The alu dest (bit 23-22) will enable the load inputs of the C, B, A or L registers and the Reg in (bit 12) selects the input to the register.

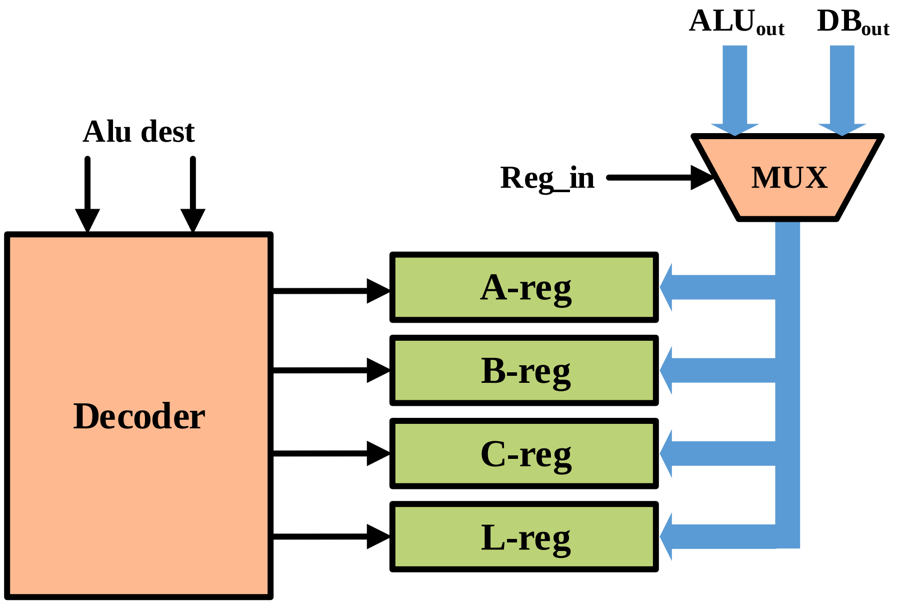

Fig. 2.7   Reg to ALU operations