Arithmetic	Logic	Shift	Branch	Memory
add	and	RL	b unc.	Load to reg from mem
sub	or	RR	b w Link	Load imm to reg
add ind.	xor	SL	b ind. w. Link	store from reg to mem
sub ind.	clear	ASR	b if zero	
		LSR	b if not zero	
			b if carry set	
			b if carry clear	

R0, R1, R2, R3, R4, R5, LR, PC

OP → 00 (Arith/Logic)

cmd	
000	add
001	sub
010	add ind.
011	sub ind.
100	and
101	or
110	xor
111	clear

OP → 01 (Shift)

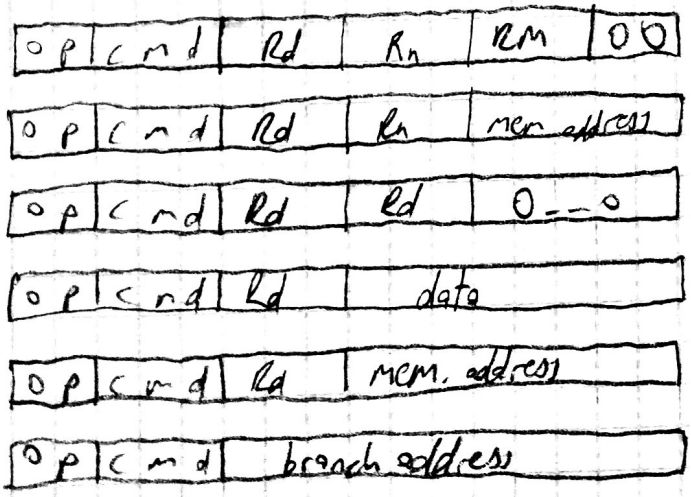
cmd	
000	RL
001	RR
010	SL
011	ASR
100	LSR
-	
-	
-	

OP → 10 (branch)

cmd	
000	b unc.
001	b w Link.
010	b ind w L.
011	b. if zero
100	b if not zero,
101	b if carry set
110	b if carry clear
111	not used

OP → 11 (mem)

cmd	
000	Load imm
001	Load
010	store



→ add, sub, and, xor, or, clear

→ add ind, sub ind.

→ Shifts

→ Load imm

→ Load/store

→ branch

Arithmetic & Logic inst. use ALU, Reg. file, Inst/Data Mem.

Shift inst. use Shift module, Reg. file, Inst/Data Mem.

Load/store inst. use Reg. file, Inst/Data Mem.

Instructions are 16 bits. Data is 8 bits.

Instruction mnemonics are easy to implement to datapath like MEM multicycle.