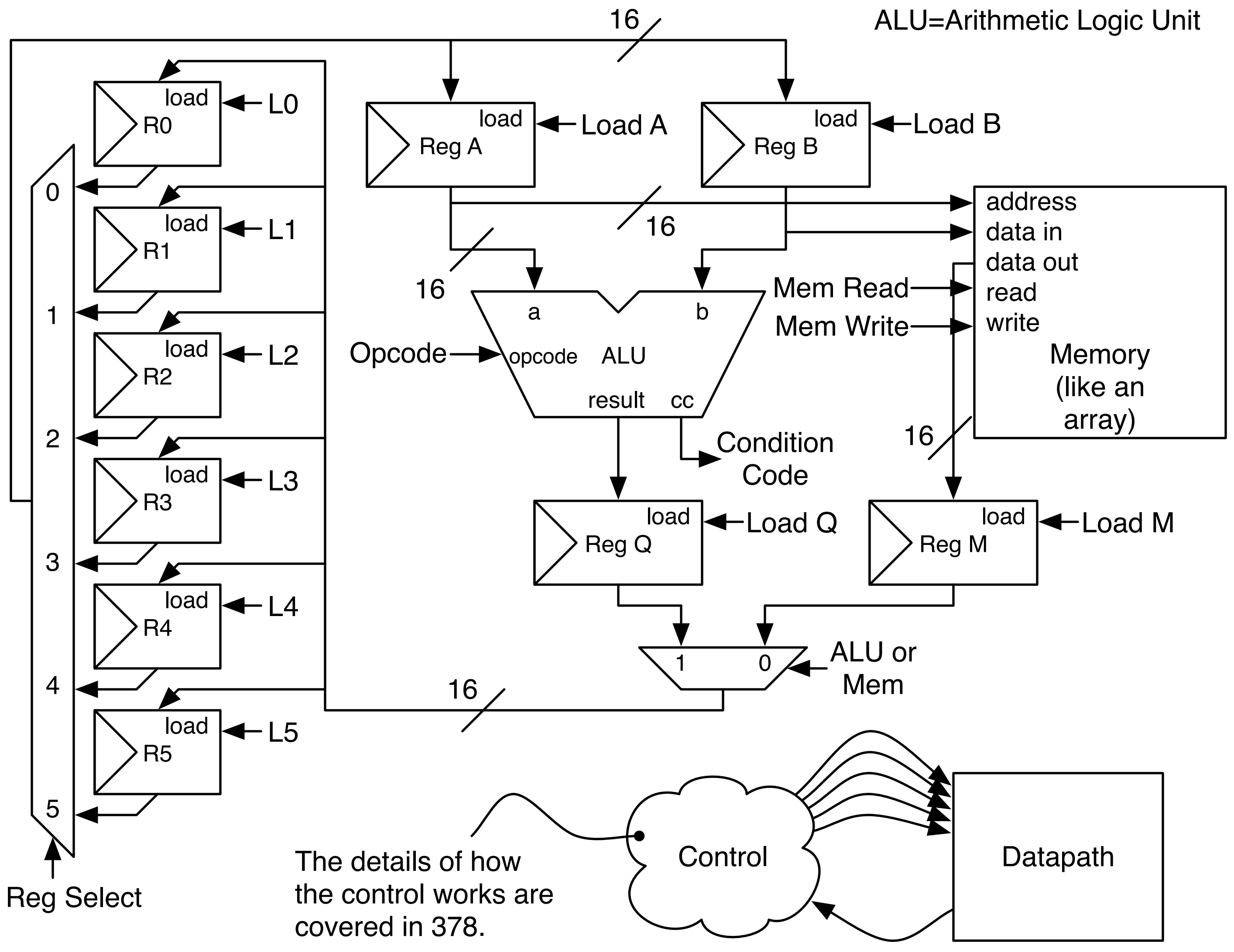
Computer Organization (a.k.a., The Mother of All Datapaths)



Instruction	Opcode	LoadA	LoadB	LoadQ	LoadM	ALUMem	MemRead	MemWrite	L{0-5}	RegSelect
Add	add	0	0	1	0	x	0	0	0	x
Subtract	sub	0	0	1	0	x	0	0	0	x
LoadAR{0-5}	x	1	0	0	0	x	0	0	0	{0-5}
LoadBR{0-5}	x	0	1	0	0	x	0	0	0	{0-5}
LoadR{0-5}ALU	x	0	0	0	0	1	0	0	{0-5}	x
LoadR{0-5}Mem	x	0	0	0	0	0	0	0	{0-5}	x
ReadMem	x	0	0	0	1	x	1	0	0	x
WriteMem	x	0	0	0	0	x	0	1	0	x

```
Control instructions:
Goto [inst#]
LessThan [inst#]
GreaterThan [inst#]
Done (a.k.a. Halt, Stop)
...
```

Write an assembly program to add 4 to the numbers stored at memory locations 100 through 200.

Initial assumptions: 4 is in R0
100 is in R1
201 is in R2
1 is in R3

1:	LoadA R1	R4 = Mem[R1]
2:	ReadMem	
3:	LoadR4 Mem	
4:	LoadA R4	R4 = R4 + 4
5:	LoadB R0	
6:	Add	
7:	LoadR4 ALU	
8:	LoadA R1	Mem[R1] = R4
9:	LoadB R4	
10:	WriteMem	
~~11:~~	~~LoadA R1~~	R1 = R1 + 1
12:	LoadB R3	
13:	Add	
14:	LoadR1 ALU	
15:	LoadA R1	if (R1 < R2) goto #1
16:	LoadB R2	
17:	LessThan #1	
18:	Done	

This instruction is not actually needed, because of instruction #8.

The Java/C/Whatever code snippet this corresponds to:

```
for (i = 100; i < 201; i++)
    A[i] = A[i] + 4;
```