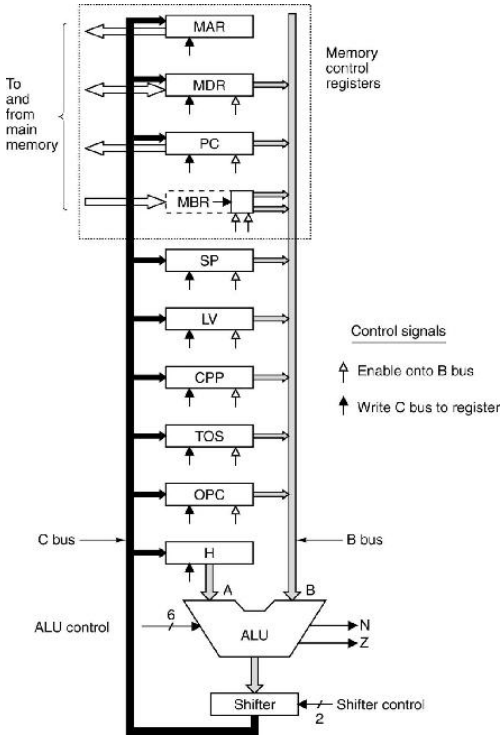
Mic-1 datapath

The Figure 4-1 datapath contains 3 types of components: registers, ALU, and shifter.



Registers

All registers are 1 word (4 bytes, 32 bits) wide.

Register	Description
SP	Stack Pointer
LV	Local Variable frame pointer
CPP	Constant pool pointer
TOS	holds the value at the Top Of Stack
H	Holding register for the ALU "A" input
MAR	Memory Address Register, holds addr of mem ref
MDR	Memory Data Register, holds data read to/from mem
PC	Program Counter, memory address of next instruction
MBR	Memory Register B, holds instruction at PC location

Most register values can be read onto the `B bus`. Most registers can be written from the `C bus`.

Memory registers (`MAR, MDR, PC, MBR`) may get/set values directly from memory.

`MBR` only uses 1 byte (of 4) to (usually) hold a 1 byte opcode.

ALU

This is our old friend from Chapter 3. 2's complement is used.

6 control bits:

- `F1, F0` - four ops are: `00`=A and B, `01`=A or B, `10`=not B, `11`=A + B
- `ENA` - enable A input, or 0 if false
- `ENB` - enable B input, or 0 if false
- `INVA` - invert A input, or just use A if false
- `INC` - increment ALU result if true by forced carry into lowest bit

Our text lists some useful ALU combinations in Figure 4-2.

F_0	F_1	ENA	ENB	INVA	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	$\bar{A}$
1	0	1	1	0	0	$\bar{B}$
1	1	1	1	0	0	A + B
1	1	1	1	0	1	A + B + 1
1	1	1	0	0	1	A + 1
1	1	0	1	0	1	B + 1
1	1	1	1	1	1	B – A
1	1	0	1	1	0	B – 1
1	1	1	0	1	1	–A
0	0	1	1	0	0	A AND B
0	1	1	1	0	0	A OR B
0	1	0	0	0	0	0
1	1	0	0	0	1	1
1	1	0	0	1	0	–1

Shifter

2 operations, each with a control bit

- `SLL8` - "Shift left logical 8 bits" - shift left 8 bits, filling least significant bits with 0's
- `SAR1` - "Shift right arithmetic 1 bit" - shift right 1 bit, leave most significant bit unchanged