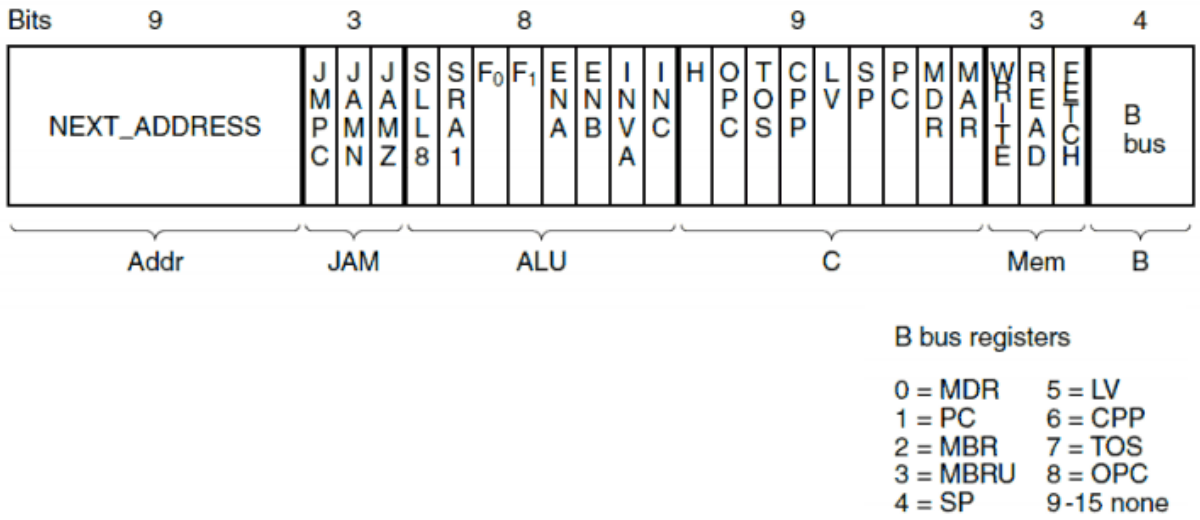
Mic-1 micro-instruction format

Figure 4-5: The micro-instruction format for the Mic-1



The micro-instruction word is 36 bits long. 24 bits control the datapath:

- `ALU` (8 bits) - ALU control bits that we know: `F0, F1, ENA, ENB, INVA, INC`. And 2 shifter bits: `SLL8` (shift left 8 bits) and `SRA1` (shift right 1 bit)
- `C bus` (9 bits) - selects registers (yes, it can be more than one) written from the C bus
- `Mem` (3 bits) - memory functions: read data, write data or fetch an instruction
- `B bus` (4 bits) - selects one register as source for the B bus

How memory refs work…

[Cycle N] assert mem read signal;
[Cycle N+1] mem value stored in MBR or MDR at **end** of cycle;
[Cycle N+2] mem value now available in MBR or MDR.

Mic-1 simplifies memory access by assuming mem read happens in 1 cycle. From page 251: "The assumption that memory takes one cycle... is never true, but the complexity introduced by a variable-length memory cycle is more than we want to deal with here." Amen!

12 bits control the selection of the next microinstruction:

- `Addr` (9 bits) - the address of the next micro-instruction
- `JAM` (3 bits) - these bits determine how the next micro-instruction is chosen

JAM bits are complicated.

- `JMPC` = use the PC address
- `JAMN` = if `N` is true (negative result in ALU), then add 1 to high-order bit of `Addr`
- `JAMZ` = if `Z` is true (zero result in ALU), then add 1 to high-order bit of `Addr`