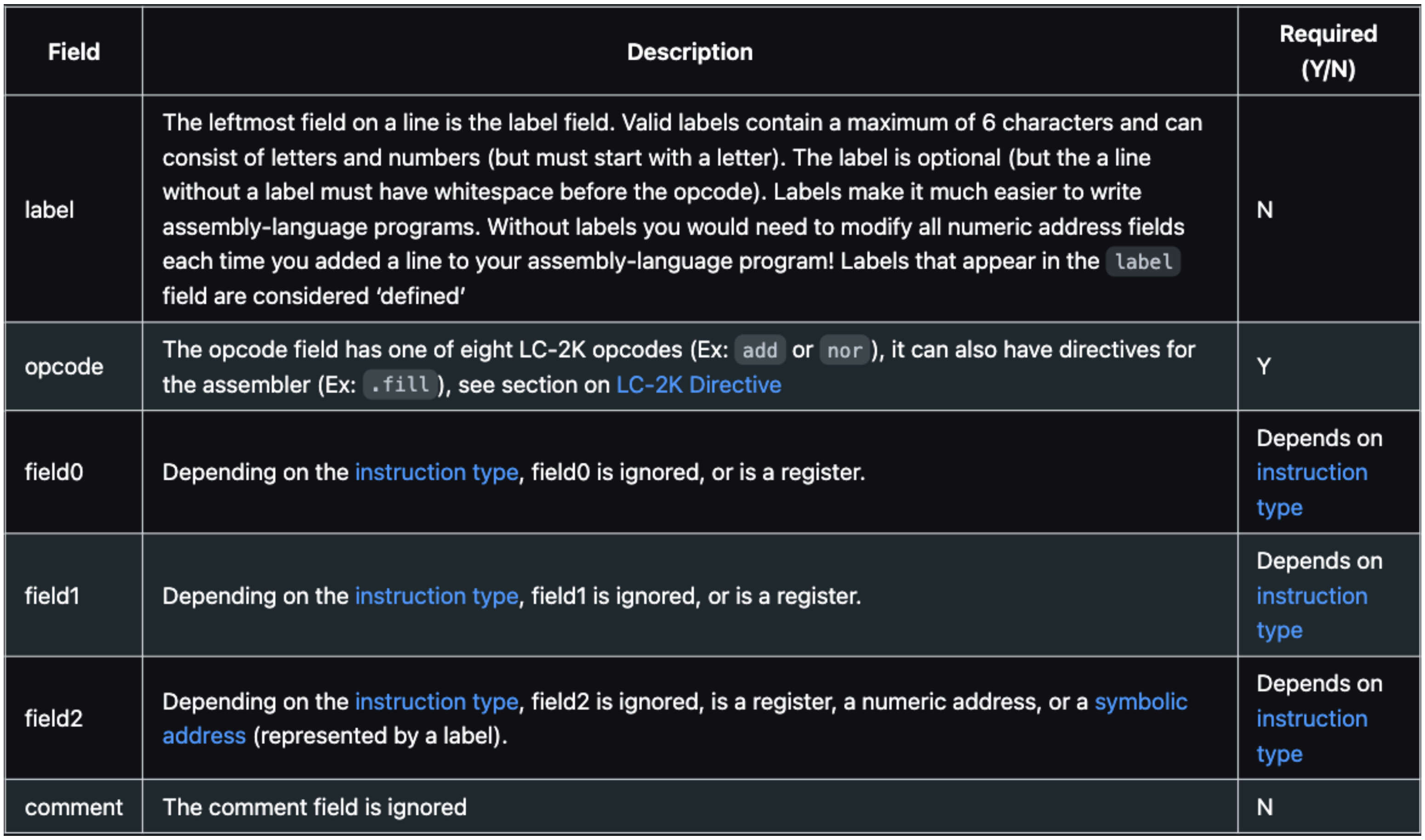
Assembly language name for instruction	Instruction Opcode in binary	Action
`add` (R-type instruction)	0b000	Add contents of `regA` with contents of `regB`, store results in `destReg`.
`nor` (R-type instruction)	0b001	Nor contents of `regA` with contents of `regB`, store results in `destReg`. This is a bitwise nor; each bit is treated independently.
`lw` (I-type instruction)	0b010	"Load Word"; Load `regB` from memory. Memory address is formed by adding `offsetField` with the contents of `regA`. Behavior is defined only for memory addresses in the range [0, 65535].
`sw` (I-type instruction)	0b011	"Store Word"; Store `regB` into memory. Memory address is formed by adding `offsetField` with the contents of `regA`. Behavior is defined only for memory addresses in the range [0, 65535].
`beq` (I-type instruction)	0b100	"Branch if equal" If the contents of `regA` and `regB` are the same, then branch to the address `PC+1+offsetField`, where `PC` is the address of this beq instruction.
`jalr` (J-type instruction)	0b101	"Jump and Link Register"; **First** store the value `PC+1` into `regB`, where `PC` is the address where this `jalr` instruction is defined. **Then** branch (set PC) to the address contained in `regA`. **Note** that this implies if `regA` and `regB` refer to the same register, the net effect will be jumping to `PC+1`.
`halt` (O-type instruction)	0b110	Increment the `PC` (as with all instructions), then halt the machine (let the simulator notice that the machine halted).
`noop` (O-type instruction)	0b111	"No Operation (pronounced no op)" Do nothing.

Instruction Type	Instructions in category	Description of required fields
R-Type Instructions	`add`, `nor`	`opcode`, `field0`, `field1`, and `field2` are required fields: `field0` is a register (regA) `field1` is a register (regB) `field2` is a register (destReg)
I-Type instructions	`lw`, `sw`, `beq`	`opcode` , `field0` , `field1` and `field2` are required fields: `field0` is a register (regA) `field1` is a register (regB) `field2` is either a numeric address, or a symbolic address (represented by a label)
J-Type instructions	`jalr`	`opcode`, `field0`, and `field1` are required fields: `field0` is a register (regA) `field1` is a register (regB)
O-Type instructions	`noop`, `halt`	Only the `opcode` field is required



Field	Description	Required (Y/N)
label	The leftmost field on a line is the label field. Valid labels contain a maximum of 6 characters and can consist of letters and numbers (but must start with a letter). The label is optional (but the a line without a label must have whitespace before the opcode). Labels make it much easier to write assembly-language programs. Without labels you would need to modify all numeric address fields each time you added a line to your assembly-language program! Labels that appear in the `label` field are considered ‘defined’	N
opcode	The opcode field has one of eight LC-2K opcodes (Ex: `add` or `nor`), it can also have directives for the assembler (Ex: `.fill`), see section on LC-2K Directive	Y
field0	Depending on the instruction type, field0 is ignored, or is a register.	Depends on instruction type
field1	Depending on the instruction type, field1 is ignored, or is a register.	Depends on instruction type
field2	Depending on the instruction type, field2 is ignored, is a register, a numeric address, or a symbolic address (represented by a label).	Depends on instruction type
comment	The comment field is ignored	N