Comparison Chart of SIC and SIC/XE machine

Specification	SIC	SIC/XE
Memory	• **Word size:** 3 bytes (24 bits) • **Total size:** 32,768 bytes (215). Thus any memory address will need at most 15 bits to be referenced ('almost' four hex characters).	• **Word size:** 3 bytes (24 bits) • **Total size:** 32,768 bytes (215). Thus any memory address will need at most 15 bits to be referenced ('almost' four hex characters).
Register	• **Total Registers:** 5 • **Accumulator (A):** Used for most of the operations (number 0) • **Index (X):** Used for indexed addressing (number 1) • **Linkage (L):** Stores return addresses for JSUB (number 2) • **Program Counter (PC):** Address for next instruction (number 8) • **Status Word (SW):** Information and condition codes (number 9).	• **Total Registers:** 9 , same 5 from SIC plus 4 additional ones. • **Base (B):** Used for base-relative addressing (number 3) • **General (S and T):** General use (numbers 4 and 5 resp.) • **Floating Point Accumulator (F):** Used for floating point arithmetic, 48 bits long (number 6)
Instruction Formats	• Only one instruction format of 24 bits (3 bytes / 1 word) • Opcode: first 8 bits, direct translation from the Operation Code Table • Flag (x): next bit indicates address mode (0 direct - 1 indexed) • Address: next 15 bits, indicate address of operand according to address mode.	• Four instruction formats • **Format 1** (1 byte): contains only operation code (straight from table) • **Format 2** (2 bytes): first eight bits for operation code, next four for register 1 and following four for register 2. • The numbers for the registers go according to the numbers indicated at the registers section (ie, register T is replaced by hex 5). • If the operation uses only one register the last hex digit becomes \0" (ie,TIXR T becomes B850) • **Format 3** (3 bytes): First 6 bits contain operation code, next 6 bits contain flags, last 12 bits contain displacement for the address of the operand. • Operation code uses only 6 bits, thus the second hex digit will be a affected by the values of the first two flags (n and i) • The flags, in order, are: n, i, x, b, p, and e. Its functionality is explained in the next section. • The last flag e indicates the instruction format (0 for 3 and 1 for 4) • **Format 4** (4 bytes): same as format 3 with an extra 2 hex digits (8 bits) for addresses that require more than 12 bits to be represented

Specification	SIC	SIC/XE
Addressing Modes	• Only two possible addressing modes • **Direct** (x = 0): operand address goes as it is Indexed (x = 1): value to be added to the value stored at the register x to obtain real address of the operand.	• five possible addressing modes plus combinations (see page 11 for examples) • **Direct** (x, b, and p all set to 0): operand address goes as it is. n and i are both set to the same value, either 0 or 1. While in general that value is 1, if set to 0 for format 3 we can assume that the rest of the flags (x, b, p, and e) are used as a part of the address of the operand, to make the format compatible to the SIC format • **Relative** (either b or p equal to 1 and the other one to 0): the address of the operand should be added to the current value stored at the B register (if b = 1) or to the value stored at the PC register (if p = 1) • **Immediate** (i = 1, n = 0): The operand value is already enclosed on the instruction (ie. lies on the last 12/20 bits of the instruction) • **Indirect** (i = 0, n = 1): The operand value points to an address that holds the address for the operand value • **Indexed** (x = 1): value to be added to the value stored at the register x to obtain real address of the operand. This can be combined with any of the previous modes except immediate.
Assembler Considerations	• Operation code gets translated directly from table (no need to check other bits) • x bit dependent on the addressing mode of the operand. If indexed the code will have to indicate it with \,X" after the operand name (ie. BUFFER,X) • The last 3 hex digits of the address will remain the same, the first hex digit (leftmost) will change if the address is indexed (first bit becomes one, thus the hex digit increases by 8). Ie, if the address of the operand is 124A and the addressing is indexed, the object code will indicate 924A.	• Operation code gets translated directly from table. While the first hex digit remains the same, the second one can change according to the values of the n and i flags. Thus, we can add 1, 2 or 3 to the operation code. • **Direct addressing** is mainly used in extended format (format 4) and is indicated with a \+" before the operand (an indication that the format is 4, which will also make the e flag to be 1). • **Relative:** for Base relative, the instruction BASE will precede the current instruction. • Any other format, except immediate, will be considered Program Counter relative. If the displacement with respect to the PC does not t into the 12 bits, the assembler should try to compute the displacement with respect to the Base register. If neither case works, the instruction should be extended to format 4, where the addressing mode becomes direct.

Specification	SIC	SIC/XE
		• **Immediate addressing** will be indicated by the use of \#" before the operand name/value (ie. #1) • **Indirect addressing** will be indicated by adding the prex \@" to the operand name (ie. @RETADR) • **Indexed addressing** will be indicated the same way as it was for the SIC machine, \,X" after the operand name (ie. BUFFER,X) • Hex digits for the address are not affected by the content of the flags, since the first two flags affect the second digit of the operation code, and the following four make up its own hex digit.