4/20/82

Review of Machine Programming Minicourse

The Minicourse by Carr Simioni on Machine programming is in general very good. The course begins with an introduction to Machine language programming principles. Part of this section was apparently extracted from the Zilog Z80 users manual. I would recommend that anyone ~~that~~ ~~no~~ seriously contemplating machine language programming purchase the Z80 users manual. The users manual gives the detail necessary to understand the Z80 instructions.

The section on Bally system operation gives a good description of the three custom chips which are responsible for much of the Bally's powerful graphics capability. Also the input/output ports are well defined. Since the custom chips and there associated I/O ports

are unique to the Bally system, this section is particularly valuable.

The section on Bally Basic Memory Usage gives a good description of how Basic mixes graphics and program text.

There is a good discussion on the reason hexadecimal (Base 16) numbering is used to represent binary numbers. The definition of Two's complement number is not entirely correct. The two's complement of a number is defined as the one's complement (ie bit reversal) plus one. In two's complement plus and minus numbers do not complement each other but excluding a carry do sum to zero.

The rest of the course deals with converting numbers as well as developing, and loading machine language code. The use of

The Stack is described. The starting location of the stack is at the top rather than at the bottom as stated in the text. The alternate register set and some of its uses is depicted. There is a good description of interrupt processing a most important aspect of machine language programming. The mechanics of producing machine code following this course are to convert Z80 opcodes and data into decimal numbers and poke these numbers into memory. I prefer poking the opcode and data in Hexadecimal (See Hex poker) since that is the standard for representing machine opcodes. In any case either method is tedious and time consuming. An assembler is what we really need!

Al Rathmell