\* Global variables:

\* A5 - target instruction pointer.

\* points to the current machine code in memory, for disassembly.

\* A6 - final target machine code in memory

\*

\* Subroutine conventions:

\* Parameters - pass using Func\_MyFunc(D1, D2, ...)

\* Return - return through D0

\* Preservation - all Dn and An registers that are modified must be saved

\* at entry on the stack, and restored from the stack before

\* RTS (D0 and A5 are exceptions).

\* Opcode Table

\* top level jump table determines first step function

\* based on 4 MSBs of opcode

\* Instructions are stored as longwords in memory

\* Address Table

\* jump table for determining addressing modes and registers

\* based on 6 bits, 3 for mode and 3 for register

\* Instructions are stored as longwords in memory

\* void Func\_MoveInstructionPointer(number of words)

\* D1

\* expects global variable A5 to contain current address location

\* void Func\_PrintSizeStandard()

\* expects global variable A5 to be pointing at current opcode.

\*

\* prints the size for instructions who fit the following configuration:

\* bits 7,6 (where 6 and 7 are of the first opcode word)

\* 0,0 = byte

\* 0,1 = word

\* 1,0 = long

\* 1,1 = ? (error)

\* void Func\_Print(MsgAddress)

\* D1

\* prints the contents of memory location specified by D1

\* int Func\_GetBits(value, lowerBit, numBits)

\* D0 D1 D2 D3

\* Returns the bits from value[lowerBit] to value[lowerBit + numBits]

$$ Jump table for top level opcode distinction

$$ Functions for determining opcodes that need a second level of distinction

$$ Functions for determining opcodes that need a third level of distinction

$$ Jump table for address mode/register distinction

$$ Functions for dealing with each individual opcode

$$ Functions for dealing with each individual addressing mode