The machine is viewed as consisting of three pieces: code space, data space and stack. Code space is for program code, data space is for static data (and the heap, which you have none of), and stack is for work and frames. There are two kinds of integers: regular [1 word = 2 bytes] and long [1 longword = 4 bytes], and there are instructions to do character [byte] arithmetic.

Schulte's Machine Instructions

CS means Code Space; DS means Data Space
integers–word long; long ints–quadword long

NAME	CODE	ARG	OLD_STK–NEW_STK	ACTION
HALT	0x00		–	
SYSCALL	0x01		call_dep_params\w_call_type–	do system function
RGOTO	0x18	w_PC_offset	–	PC = PC + offset
RGOZ	0x1A	w_PC_offset	w_flag–	if flag=0, goto PC + offset
PUSHW	0x1C	w_value	–w_value	push value
GETSW	0x30		w_d_addr–w_value	
PUTSW	0x34		w_d_addr\w_value–	
ADDI	0x50		int1\int2–int1+int2	
SUBI	0x54		int1\int2–int1-int2	
MULI	0x58		int1\int2–int1*int2	
DIVI	0x5C		int1\int2–int1/int2	
ANDW	0x60		word1\word2–word1_bit_and_word2	
ORW	0x64		word1\word2–word1_bit_or_word2	
XORW	0x68		word1\word2–word1_bit_xor_word2	
NOTW	0x6C		word–bit_not_of_word	
TSTEQI	0x78		int–w_flag	if =0, push -1 else push 0
TSTLTI	0x7C		int–w_flag	if <0, push -1 else push 0
DUPW	0x80		w_value–w_value\w_value	duplicate
SWAPW	0x84		word1\word2–word2\word1	swap

SYSCALL functions (stack pictures are just before the SYSCALL):

NAME	CODE	ARG	OLD_STK–NEW_STK	ACTION
READ_INT	0		0–int	prompts and pushes read int on stack
PRINT_INT	1		int\1–	prints the integer

Format of the Output File

The object file is an ASCII file containing the generated code with intermixed directives to the linker/loader.

The instructions can be either written as approved memonics (in all upper case) or as the equivalent numbers. The arguements are numbers or symbols. the forward references are handled automatically.

A number followed immediately by a colon (":") (no space between number and colon) is taken as a command to change the next storage location (PC) to the number plus the start address of the module [specify the load address]. The initial load address is taken to be location 0 in the code section.

Ther are a series of linker/loader directives:

`.CODE` indicates that the loading is to be done to the code section [the initial default].

`.DATA` indicates that the loading is to be done to the data section.

name: (no space between name and colon) denotes a global symbol whose value is the (relocatable) current location in the current section. The loader complains if you try to redefine the name, and uses the first occurence.