

Operations											
Opcode	Name	Arguments	Binary	Hexa	Description	Carry	Codage Op	Cycles	Label size		
1	live	T_DIR	-	00000001	alive	0	0	10	4		
2	ldi	T_DIR,T_DIR_IND	T_REG	00000010	load	1 vers 0	1	5	4		
3	add	T_REG	T_REG,T_DIR_IND	00000011	add	0	5	5	4		
4	add	T_REG	T_REG	00000100	add	1 vers 0	1	10	4		
5	sub	T_REG	T_REG	00000101	subtraction	1 vers 0	1	10	4		
6 and	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	T_REG	00000110	and	0	11, 12, 13	11 12 > 13	1	6	4
7	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	T_REG	00000111	or	1	11, 12, 13	11 12 > 13	1	6	4
8	xor	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	00001000	xor	1	11, 12, 13	11 12 > 13	1	6	4
9	zjmp	T_DIR	-	00001001	jump if carry == 1	0	0	20	2		
10	ldi	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	00001010	load index	1	0	1	25	2	
11	stl	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	00001011	store index	0	25	2			
12	fork	T_DIR	-	00001100	fork	0	0	800	2		
13	ldi	T_DIR,T_DIR_IND	T_REG	00001101	long load	1 vers 0	1	10	4		
14	ldi	T_REG,T_DIR,T_DIR_IND	T_REG,T_DIR,T_DIR_IND	00001110	long load index	1 vers 0	0	50	2		
15	fork	T_DIR	-	00001111	long fork	0	0	1000	2		
16	aff	T_REG	-	00001000	aff	0	1	2	4		

Arguments				
Name	Sign	Binary code	Encode: (octet)	Значение
T_REG	r	01	1	Перекрут <i>ix</i> (раз с = число, количество байтов) в направлении от 1 до REG_NUMBER
T_DIR	%	10	24	Маска 2-ух 4-х битов в зависимости от label
T_IND		11	2	Перекрут в том числе значения T_IND от PC и сдвиг на 4 бита
Label				abcdefhiklmnopqrstuvwxyz: 0123456789

.COR FILE STRUCTURE		
size in bytes	description	
4	magic	
PROG_NAME_LENGTH	bot name	if (PROG_NAME_LENGTH + 1) % 4 != 0) -> выравнивание = 4 - (PROG_NAME_LENGTH + 1) % 4 - перепрыг на выравнивание
4	NULL	
4	size of executable code	
COMMENT_LENGTH	bot comment	
4	NULL	
N	executable code	

Processus value for each champions		
Name	Qty	Descriptions
Canry	1	Флаг, который меняется некоторыми инструкциями и используется в <code>jmp</code>
PC	1	Позиция процесса (адреса)
Registers	REG_NUMBER	(своего рода буфер на) REG_NUMBER регистров (переменных), каждый из которых, записывает REG_SIZE байт и регистры (PC) могут содержать значение

[illegible]

Assembler errors		
Validation	Type	Error message
name	NET STOP	Syntax error at token [TOKEN0004 001] LABEL "2"
name	NET MASM	Syntax error at token [TOKEN0010 014] ENDLINE
name	NET записывающийся кавычки	Syntax error at token [TOKEN0009 001] END "null"
name	NET STOP	Lexical error at [2 10]
name	NET KAWYCHKI	Syntax error at token [TOKEN0001 007] INSTRUCTION "open"
comment	NET KAWYCHKI	Lexical error at [2 10]
comment	NET записывающийся кавычки	Syntax error at token [TOKEN0009 001] END "null"
comment	NET STOP	Lexical error at [2 10]
comment	NET STOP	Syntax error at token [TOKEN0004 001] LABEL "2"
comment	NET MASM	Syntax error at token [TOKEN0001 014] ENDLINE
comment	NET записывающийся кавычки	Syntax error at token [TOKEN0009 001] END "null"
commands	NET команды указывающей в аргументе T, END	No such label file while attempting to dereference token [TOKEN0004 014] DIRECT LABEL "file"
commands	NET команды указывающей в LABEL	Invalid instruction of token [TOKEN0005 003] INSTRUCTION "x"
commands	указано больше аргументов	Syntax error at token [TOKEN0007 016] DIRECT "1"
commands	указан не корректный аргумент	Invalid parameter 0 type required for instruction file
commands	в команде	Syntax error at token [TOKEN0004 003] INSTRUCTION "x"
label	указан LABEL без команды	Syntax error at token [TOKEN0010 005] END "null"
label	два раза label	Syntax error at [2 10]
segment	два раза segment	Syntax error at token [TOKEN0009 001] COMMENT "comment"

ZORK EXPLANATION					
operation	note	value	hexa	byte No	
sti			0b	0	Здесь OPCODE
	codeage	01 10 10 00 Dwe8	08	1	Кодировка: 1 - пер. 2-прав. 3 - прав.
	arg1	%i	01	2	01 - первый регистр r1 не 0.
	arg2	%live	00	3	выделено байт 3 и 4
			01	4	Здесь значение (00)
	codeage	1%0	00	5	выделено байт 5 и 6
			01	6	Здесь значение (01)
and			06	7	Здесь OPCODE and6
	codeage	01 10 01 00 Dwe4	04	8	Кодировка: 1 - пер. 2-прав. 3 - регистр
	arg1	%i	01	9	Здесь значение (01) (1 байт для per)
	arg2	%0	00	10	для direct и выделится 4 байта
			00	11	и в значении у нас ноль, так что
			00	12	00 00 00 00
			00	13	00000000 00000000 00000000 00000000
	arg3	r1	01	14	1-ый регистр перем., выделено 1 байт
			01	15	Здесь OPCODE and1
	arg1	%i	00	16	00 - выделено 1 байт
	arg2	%i	00	17	(во так надо, написано в таблице)
			01	18	значение 01
			00	19	
zmp			09	20	Здесь OPCODE and9
	arg1	%live	00	21	
			00	22	выделено 2 бай. 00