Befehl	Argumente*	Beschreibung		
	<u> </u>	Addiert die zwei obersten Elemente des Stacks und		
add	-	entfernt diese. Das Ergebnis wird auf den Stack		
		gelegt.		
		Verknüpft die zwei obersten Elemente mit der AND-		
and	-	Operation und entfernt diese. Das Ergebnis wird		
		auf den Stack gelegt.		
dec	-	Dekrementiert den obersten Wert auf dem Stack.		
inc	-	Inkrementiert den obersten Wert auf dem Stack.		
inv	-	Invertiert den obersten Wert auf dem Stack.		
or		Verknüpft die zwei obersten Elemente mit der OR-		
	-	Operation und entfernt diese. Das Ergebnis wird		
		auf den Stack gelegt.		
nop	-	Keine Operation.		
sub		Subtrahiert die zwei obersten Elemente des Stacks		
		und entfernt diese. Das Ergebnis wird auf den		
	_	Stack gelegt. Dabei wird das zweite Element		
		"minus" das oberste Element gerechnet.		
swap		Vertauscht die zwei obersten Elemente auf dem		
эмар		Stack.		
xor		Verknüpft die zwei obersten Elemente mit der XOR-		
	-	Operation und entfernt diese. Das Ergebnis wird		
		auf den Stack gelegt.		
call	<addr> [14]</addr>	(Funktionsaufruf) Springt an die Stelle von		
		<addr> und führt den dort stehenden Code aus, bis</addr>		
		ein return erreicht wird.		
		Wichtig: Damit "return" wieder an die korrekte		
		Stelle zurückspringt, muss der Stack am Ende der		
		"Funktion" im gleichen Zustand wie zu Beginn der		
		"Funktion" sein.		
return	-	Springt zurück an die Stelle des letzten Funktionsaufrufes.		
goto	/addr\ [14]	Springt bedingungslos an die Stelle <addr>.</addr>		
goto	goto <addr> [14] Springt bedingungslos an die Stelle <addr Springt an die Stelle <addr>, falls das d</addr></addr </addr>			
jmz	<addr> [14]</addr>	Element des Stacks gleich null ist.		
		Springt an die Stelle <addr>, falls das oberste</addr>		
jmnz	<addr> [14]</addr>	Element des Stacks ungleich null ist.		
		Springt an die Stelle <addr>, falls die obersten</addr>		
jmc	<addr> [14]</addr>	zwei Stackelemente gleich sind.		
рор	_	Entfernt das oberste Element vom Stack.		
push	<elem> [8]</elem>	Legt den Wert <elem> auf den Stack.</elem>		
•	· <u>L</u> -J	Führt eine Schiebeoperation um 8 Bits nach links		
shl8	_	auf dem obersten Stackelement aus.		
-1- 4		Führt eine Schiebeoperation um 1 Bit nach rechts		
shr1	-	auf dem obersten Stackelement aus.		
dup		Erstellt eine Kopie des obersten Stackelements		
	_	und legt diese auf den Stack.		
dload		Lädt einen 8-Bit Wert aus dem RAM von der		
		Adresse, welche an erster Stelle auf dem Stack		
	_	liegt und ersetzt das oberste Stackelement mit		
		dem Wert.		
dstore		Speichert das zweite Element auf dem Stack an der		
436016		Adresse, welche an erster Stelle auf dem Stack		

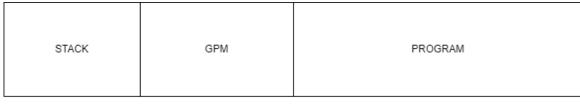
liegt. Nur die Adresse (1. Element) wird vom
Stack entfernt, nicht der Wert (2. Element).

^{*} Die Zahl in den eckigen Klammern entspricht der Größe in Bit.

Befehl	OP Code (dezimal)	OP Code (hexadezimal)
add	128	80
and	131	83
dec	134	86
inc	136	88
inv	166	a6
or	138	8a
nop	0	0
sub	141	8d
swap	144	90
xor	150	96
call	156	9c
return	161	a2
goto	30	1e
jmz	37	25
jmnz	44	2c
jmc	52	34
рор	60	3c
push	65	41
sh18	70	46
shr1	73	49
dub	75	4b
dloads	96	60
dstore	105	69

<u>Speicheraufteilung:</u>

Namo	Anz.	Einheiten	Adressierungsbereich		Gesamtspeicheranteil
Name	Zellen	ETHHETCEN	virtuell	physisch	desamespereneranceir
STACK	256	32 Bit	0-255	0-255	25% (1024 Byte)
GPM	256	32 Bit	256-511	256-511	25% (1024 Byte)
PROGRAM	2048	8 Bit	2048-4095	512-1023	50% (2048 Byte)



0 256 512