RT-Assembler: Befehlsliste

(`a b c p q` ... Variablen oder Zahlen, `i` ... Zahlen, `m` ... Marken, `s` ... Zeichenketten)

`mov`	`a b`	$a \leftarrow b$	(Transportbefehl)
`clr`	`a`	$a \leftarrow 0$	(Löschbefehl)
`inc`	`a`	$a \leftarrow a+1$	(Inkrement)
`dec`	`a`	$a \leftarrow a-1$	(Dekrement)
`add`	`a b`	$a \leftarrow a+b$	
`sub`	`a b`	$a \leftarrow a-b$	
`mul`	`a b`	$a \leftarrow a \cdot b$	
`div`	`a b`	$a \leftarrow a / b$	
`power`	`a b`	$a \leftarrow a^b$	
`root`	`a b`	$a \leftarrow b$-te Wurzel aus a. (Bei Quadratwurzel b=2 angeben.)	
`exp`	`a`	$a \leftarrow e^a$	
`exp10`	`a`	$a \leftarrow 10^a$	
`exp2`	`a`	$a \leftarrow 2^a$	
`expx`	`a b`	$a \leftarrow b^a$	
`log`	`a`	$a \leftarrow \log_e(a)$	
`log10`	`a`	$a \leftarrow \log_{10}(a)$	
`log2`	`a`	$a \leftarrow \log_2(a)$	
`logx`	`a b`	$a \leftarrow \log_b(a)$	
`sin`	`a`	$a \leftarrow \sin(a)$	
`cos`	`a`	$a \leftarrow \cos(a)$	
`tan`	`a`	$a \leftarrow \tan(a)$	
`cot`	`a`	$a \leftarrow \cot(a)$	
`sec`	`a`	$a \leftarrow \sec(a)$	$(= 1/\cos(a))$
`csc`	`a`	$a \leftarrow \mathrm{cosec}(a)$	$(= 1/\sin(a))$
`asin`	`a b`	$a \leftarrow \arcsin(a)$	
`acos`	`a b`	$a \leftarrow \arccos(a)$	
`atan`	`a b`	$a \leftarrow \arctan(a)$	
`acot`	`a b`	$a \leftarrow \mathrm{arccot}(a)$	
`asec`	`a b`	$a \leftarrow \mathrm{arcsec}(a)$	
`acsc`	`a b`	$a \leftarrow \mathrm{arccosec}(a)$	
`sinh`	`a`	$a \leftarrow$ sinus hyperbolicus (a)	
`cosh`	`a`	$a \leftarrow$ cosinus hyperbolicus (a)	
`tanh`	`a`	$a \leftarrow$ tangens hyperbolicus (a)	
`coth`	`a`	$a \leftarrow$ cotangens hyperbolicus (a)	
`sech`	`a`	$a \leftarrow$ secans hyperbolicus (a)	
`csch`	`a`	$a \leftarrow$ cosecans hyperbolicus (a)	
`asinh`	`a`	$a \leftarrow$ area sinus hyperbolicus (a)	
`acosh`	`a`	$a \leftarrow$ area cosinus hyperbolicus (a)	
`atanh`	`a`	$a \leftarrow$ area tangens hyperbolicus (a)	
`acoth`	`a`	$a \leftarrow$ area cotangens hyperbolicus (a)	
`asech`	`a`	$a \leftarrow$ area secans hyperbolicus (a)	
`acsch`	`a`	$a \leftarrow$ area cosecans hyperbolicus (a)	
`bin`	`a`	wenn $a \neq 0$ dann $a \leftarrow 1$, sonst $a \leftarrow 0$ (Logische Identität)	
`not`	`a`	wenn $a = 0$ dann $a \leftarrow 1$, sonst $a \leftarrow 0$ (Logische Negation)	
`or`	`a b`	wenn $a \neq 0$ oder $b \neq 0$ dann $a \leftarrow 1$, sonst $a \leftarrow 0$ (Logisch Oder)	
`and`	`a b`	wenn $a \neq 0$ und $b \neq 0$ dann $a \leftarrow 1$, sonst $a \leftarrow 0$ (Logisch Und)	

`abs`	`a`	Absolutbetrag von a bilden
`neg`	`a`	Vorzeichen von a umkehren (Negation)
`sgn`	`a`	Vorzeichen von a (+1, 0 oder −1)
`round`	`a`	a runden
`ceil`	`a`	a aufrunden
`floor`	`a`	a abrunden
`fix`	`a`	Vorkommastellen von a (a zur 0 hin runden)
`frac`	`a`	Nachkommastellen von a
`clip`	`a b c`	wenn $a<b$ dann $a \leftarrow b$, wenn $a>c$ dann $a \leftarrow c$, sonst $a \leftarrow a$
`cmod`	`a b c`	a mit Modulofunktion (Sägezahn) in b ... c clippen („clip and modulo“)
`random`	`a`	$a \leftarrow$ Zufallszahl zwischen 0 und 1
`cmpgt`	`a b m`	wenn $a>b$ Sprung nach m („compare greather than“)
`cmpge`	`a b m`	wenn $a \geq b$ Sprung nach m
`cmplt`	`a b m`	wenn $a<b$ Sprung nach m
`cmple`	`a b m`	wenn $a \leq b$ Sprung nach m
`cmpeq`	`a b m`	wenn $a=b$ Sprung nach m
`cmpne`	`a b m`	wenn $a \neq b$ Sprung nach m
`tstgt`	`a m`	wenn $a \geq 0$ Sprung nach m („test greather than“)
`tstge`	`a m`	wenn $a>0$ Sprung nach m
`tstlt`	`a m`	wenn $a<0$ Sprung nach m
`tstle`	`a m`	wenn $a \leq 0$ Sprung nach m
`tsteq`	`a m`	wenn $a=0$ Sprung nach m
`tstne`	`a m`	wenn $a \neq 0$ Sprung nach m
`jump`	`m`	Unbedingter Sprung nach m
`input`	`a s`	fordert Zahl a im Dialog mit Text s an (mit Programmunterbrechung)
`output`	`a s`	zeigt Zahl a im Dialog mit Text s an (mit Programmunterbrechung)
`pause`	`s`	gibt im Dialog einen Text s aus (mit Programmunterbrechung)
`proof`	`a s`	Informationsausgabe von a mit Text s (keine Programmunterbrechung)
`info`	`s`	Informationsausgabe Text s (keine Programmunterbrechung)
`cls`		Löschen des Ausgabetextes
`printn`	`a b c`	Ausgabe Zahl a in den Ausgabetext (mit b Vor- und c Nachkommastellen)
`prints`	`s`	Ausgabe Zeichenkette s in den Ausgabetext
`save`	`s`	Speichern des Ausgabetextes in Textdatei `s.txt`
`read`	`a b`	Zahl a (b=0) oder Feld a (b=Länge+1) aus Datei `a.dat` lesen
`write`	`a b`	Zahl a (b=0) oder Feld a (b=Länge+1) in Datei `a.dat` schreiben
`adrof`	`p a`	$p \leftarrow$ Adresse von a.
`get`	`a p q`	Das Symbol mit der Adresse (p+q) nach a lesen
`put`	`p q a`	a auf das Symbol mit der Adresse (p+q) schreiben
`init`		Programminitialisierung. Wird automatisch als 1. Befehl erzeugt.
`nop`		Nullbefehl („no operation“). Dieser Befehl macht nichts
`mode`	`a`	Setzen Programmmodus. a=0: ‚Ohne Halt‘, 1: ‚Fehlerhalt‘, 2: ‚Schrittweise‘
`halt`		Haltbefehl. Hält das Programm an
`err`	`a b`	$a \leftarrow$ Fehlercode (0: Kein Fehler, ≠0: Fehler). Bei Fehler Sprung nach m
`exit`		Programmbeendigung. Rückkehr zum Betriebssystem
`_name`	`s`	Dem Programm den Namen s geben
`_var`	`a`	Variable a deklarieren
`_dim`	`a i`	Feld a mit den Feldelementen $a(0)$ ... $a(i)$ definieren
`_lab`	`m`	Marke m definieren. – Andere Schreibweise: `m:`
`_config`	`i`	Virtuelle Maschine mit Parameter i konfigurieren
`_end`		Programmende. Letzte Zeile im Quellprogrammes