Instruction Groups.

Assembly Asylum Overview.

Binary.

SHL, SHR, XOR, OR, AND, NOT, ROR, ROL

General / Variable Operators.

LEN, STR, CP

Logic / Control.

JP, JZ, JNZ, JL, JLZ, JG, JGZ, LP, JSR, JSRZ, JSRNZ, RTS, RTI, CMP

Maths.

INC, DEC, ADD, SUB, MUL, DIV, NEG, ABS, TRUNC, RND, CEIL, ROUND, SQRT, MOD, POW, COS, ACOS, SIN, ASIN, TAN, ATAN

Registers.

LDA, STA, PUSH, POP, XCH, PUSHA, POPA, STC

System.

IN, OUT, SYS

Thread.

NOP, BRK, TICKS, IRQ, YLD, SLEEP, HLT

Parameter Types.

Constants, Memory Address, Variables, [pointer]s, Registers and Variable Offsets.

Compiler Directives.

.File, .DB, .DD, .DW, .MB .Includestart:irq:

Constants.

True, False, Pi…

Total Instructions.

64 parent instructions not including variations based on parameters.

Allowable Patterns

The list below shows the allowable patterns the compiler will accept with error. Although not a complete user manual it will help with getting started as we work on a full system manual.

Parameters:

- $address is the address of a variable loaded with the LDA $VARNAME syntax
- A constant is a numeric value

- A variable is the actual variable name specified after the compiler '.' directives
- A register is one of the allowable system registers "a, b, c, d, x, y, r0, r1'
- A pointer is a register containing the address of a variable with the register name enclosed in square brackets
- A label is a : colon appended name within your code used for logic operations. "_start:" and "_irq:" are examples of system specific labels.
- Any parameter specified after a '+' sign is an offset from the variable starting address.

```
nop
brk
inc
dec
in
out
lda $address
lda constant
lda register
lda variable
lda variable constant
lda variable register
lda variable variable
lda [pointer]
lda [pointer] + constant
lda [pointer] + register
lda [pointer] + variable
sta register
sta variable
sta variable + constant
sta variable + register
sta variable + variable
sta [pointer]
sta [pointer] + constant
sta [pointer] + register
sta [pointer] + variable
lp label
jp label
jz label
jnz label
jl label
jlz label
jg  label
jgz label
jsr label
jsrz label
jsrnz label
rts
rti
```

cmp $address
cmp constant
cmp register
cmp variable
cmp variable + constant
cmp variable + register
cmp variable variable
cmp [pointer]
cmp [pointer] + constant
cmp [pointer] + register
cmp [pointer] + variable
len variable
len [pointer]
push register
pop register
xch register, register
sub constant
sub register
sub variable
sub variable + constant
sub variable + register
sub variable + variable
sub [pointer]
sub [pointer] + constant
sub [pointer] + register
sub [pointer] + variable
add constant
add register
add variable
add variable + constant
add variable + register
add variable + variable
add [pointer]
add [pointer] + constant
add [pointer] + register
add [pointer] + variable
mul constant
mul register
mul variable
mul,variable + constant
mul,variable + register
mul,variable +variable
mul [pointer]
mul [pointer] + constant
mul [pointer] + register
mul [pointer] + variable
div constant
div register
div variable
div variable + constant
div variable + register

div variable + variable
div [pointer]
div [pointer] + constant
div [pointer] + register
div [pointer] + variable
shl constant
shr constant
neg
abs
trunc
rnd
ticks
irq
sys constant
yld
sleep constant
ceil
round
sqrt
mod
pow
cos
acos
sin
asin
tan
atan
xor
or
and
not constant
pusha
popa
str variable
str [pointer]
cp variable
cp [pointer]
ror constant
rol constant
hlt constant
stc