Opcodes' tables

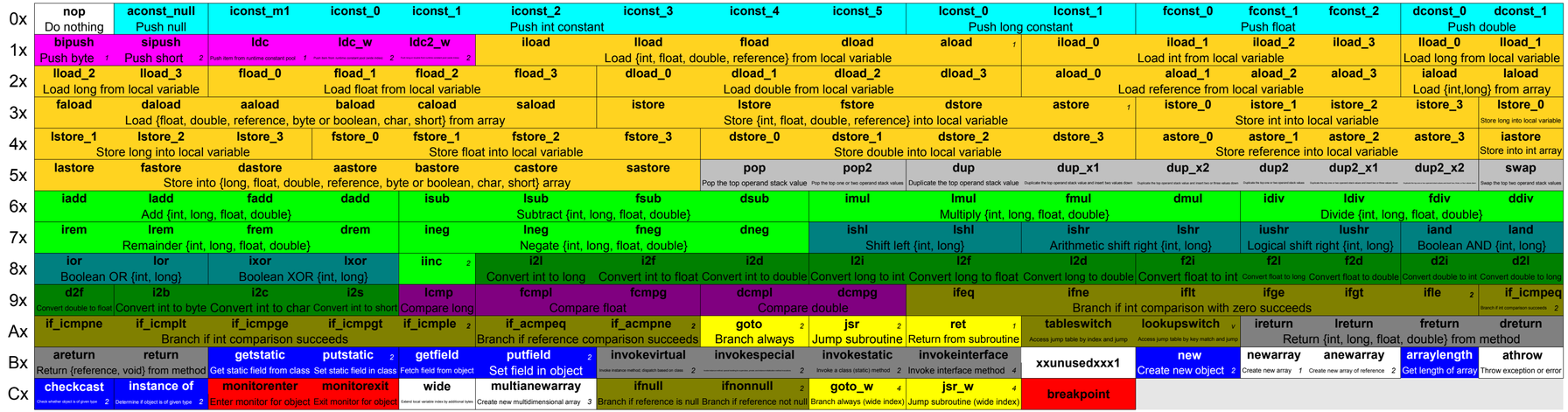
Complete

This file is free to modify and re-use, with no restrictions, even commercially.
it's an OpenOffice document.
1 grab its content via: *svn export https://corkami.googlecode.com/svn/trunk/oOo/opcodes_tables_complete*
2 rezip with subdirectories as opcodes_tables_complete.ods

opcodes tables are available as compact and complete form.

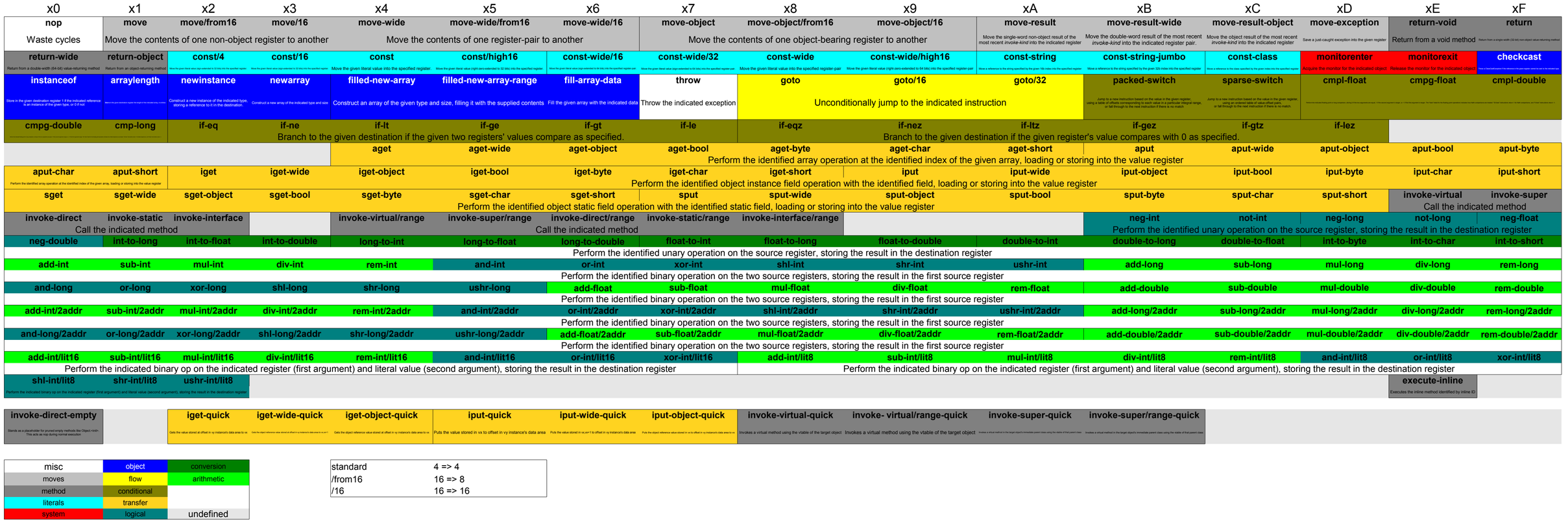
Feb 21, 2012
Ange Albertini 2012 http://corkami.com

inspired by the work of Daniel Plohmann


x0 x1 x2 x3 x4 x5 x6 x7 x8 x9 xA xB xC xD xE xF
0x
nop
Do nothing
aconst_null
Push null
iconst_m1
iconst_0
iconst_1
iconst_2
iconst_3
iconst_4
iconst_5
Push int constant
lconst_0
lconst_1
Push long constant
fconst_0
fconst_1
fconst_2
Push float
dconst_0
dconst_1
Push double
1x
bipush
Push byte 1
sipush
Push short 2
ldc
Push item from runtime constant pool 1
ldc_w
Push item from runtime constant pool (wide index) 2
ldc2_w
Push long or double from runtime constant pool (wide index) 2
iload
lload
fload
dload
aload 1
Load {int, float, double, reference} from local variable
iload_0
iload_1
iload_2
iload_3
Load int from local variable
lload_0
lload_1
Load long from local variable
2x
lload_2
lload_3
Load long from local variable
fload_0
fload_1
fload_2
fload_3
Load float from local variable
dload_0
dload_1
dload_2
dload_3
Load double from local variable
aload_0
aload_1
aload_2
aload_3
Load reference from local variable
iaload
laload
Load {int,long} from array
3x
faload
daload
aaload
baload
caload
saload
Load {float, double, reference, byte or boolean, char, short} from array
istore
lstore
fstore
dstore
astore 1
Store {int, float, double, reference} into local variable
istore_0
istore_1
istore_2
istore_3
Store int into local variable
lstore_0
Store long into local variable
4x
lstore_1
lstore_2
lstore_3
Store long into local variable
fstore_0
fstore_1
fstore_2
fstore_3
Store float into local variable
dstore_0
dstore_1
dstore_2
dstore_3
Store double into local variable
astore_0
astore_1
astore_2
astore_3
Store reference into local variable
iastore
Store into int array
5x
lastore
fastore
dastore
aastore
bastore
castore
sastore
Store into {long, float, double, reference, byte or boolean, char, short} array
pop
Pop the top operand stack value
pop2
Pop the top one or two operand stack values
dup
Duplicate the top operand stack value
dup_x1
Duplicate the top operand stack value and insert two values down
dup_x2
Duplicate the top operand stack value and insert two or three values down
dup2
Duplicate the top one or two operand stack values
dup2_x1
Duplicate the top one or two operand stack values and insert two or three values down
dup2_x2
Duplicate the top one or two operand stack values and insert two, three or four values down
swap
Swap the top two operand stack values
6x
iadd
ladd
fadd
dadd
isub
lsub
fsub
dsub
Add {int, long, float, double}
imul
lmul
fmul
dmul
Subtract {int, long, float, double}
idiv
ldiv
fdiv
ddiv
Multiply {int, long, float, double}
Divide {int, long, float, double}
7x
irem
lrem
frem
drem
Remainder {int, long, float, double}
ineg
lneg
fneg
dneg
Negate {int, long, float, double}
ishl
lshl
Shift left {int, long}
ishr
lshr
Arithmetic shift right {int, long}
iushr
lushr
Logical shift right {int, long}
iand
land
Boolean AND {int, long}
8x
ior
lor
Boolean OR {int, long}
ixor
lxor
Boolean XOR {int, long}
iinc 2
i2l
i2f
i2d
Convert int to long
Convert int to float
Convert int to double
l2i
Convert long to int
l2f
Convert long to float
l2d
Convert long to double
f2i
Convert float to int
f2l
Convert float to long
f2d
Convert float to double
d2i
Convert double to int
d2l
Convert double to long
9x
d2f
Convert double to float
i2b
Convert int to byte
i2c
Convert int to char
i2s
Convert int to short
lcmp
Compare long
fcmpl
fcmpg
Compare float
dcmpl
dcmpg
Compare double
ifeq
ifne
iflt
ifge
ifgt
ifle 2
Branch if int comparison with zero succeeds
if_icmpeq
Branch if int comparison succeeds 2
Ax
if_icmpne
if_icmplt
if_icmpge
if_icmpgt
if_icmple 2
Branch if int comparison succeeds
if_acmpeq
if_acmpne 2
Branch if reference comparison succeeds
goto 2
Branch always
jsr 2
Jump subroutine
ret 1
Return from subroutine
tableswitch v
Access jump table by index and jump
lookupswitch v
Access jump table by key match and jump
ireturn
lreturn
freturn
dreturn
Return {int, long, float, double} from method
Bx
areturn
return
Return {reference, void} from method
getstatic
Get static field from class
putstatic 2
Set static field in class
getfield
Fetch field from object
putfield 2
Set field in object
invokevirtual
Invoke instance method; dispatch based on class 2
invokespecial
invokestatic
Invoke a class (static) method 2
invokeinterface
Invoke interface method 4
xxunusedxxx1
new
Create new object 2
newarray
Create new array 1
anewarray
Create new array of reference 2
arraylength
Get length of array
athrow
Throw exception or error
Cx
checkcast
Check whether object is of given type 2
instance of
Determine if object is of given type 2
monitorenter
Enter monitor for object
monitorexit
Exit monitor for object
wide
Extend local variable index by additional bytes
multianewarray
Create new multidimensional array 3
ifnull
Branch if reference is null
ifnonnull 2
Branch if reference not null
goto_w 4
Branch always (wide index)
jsr_w 4
Jump subroutine (wide index)
breakpoint
Fx
impdep1
impdep2
misc
constants
immediates
locals
stack
arithmetic
logical
conversion
comparison
conditional
flow
method
object
system
undefined

JVM (Java)

Common Intermediate Language (.Net)


x0 x1 x2 x3 x4 x5 x6 x7 x8 x9 xA xB xC xD xE xF
0x
nop
Waste cycles
move
move/from16
move/16
Move the contents of one non-object register to another
move-wide
move-wide/from16
move-wide/16
Move the contents of one register-pair to another
move-object
move-object/from16
move-object/16
Move the contents of one object-bearing register to another
move-result
Move the single-word non-object result of the most recent invoke-kind into the indicated register
move-result-wide
Move the double-word result of the most recent invoke-kind into the indicated register pair.
move-result-object
Move the object result of the most recent invoke-kind into the indicated register
move-exception
Save a just-caught exception into the given register
return-void
Return from a void method
return
1x
return-wide
Return from a double-width (64-bit) value-returning method
return-object
Return from an object-returning method
const/4
const/16
const
Move the given literal value into the specified register.
const/high16
const-wide/16
const-wide/32
const-wide
Move the given literal value into the specified register-pair
const-wide/high16
const-string
const-string-jumbo
const-class
monitorenter
Acquire the monitor for the indicated object
monitorexit
Release the monitor for the indicated object
checkcast
2x
instanceof
Store in the given destination register 1 if the indicated reference is an instance of the given type, or 0 if not
arraylength
newinstance
Construct a new instance of the indicated type, storing a reference to it in the destination
newarray
Construct a new array of the indicated type and size
filled-new-array
Construct an array of the given type and size, filling it with the supplied contents
filled-new-array-range
fill-array-data
Fill the given array with the indicated data
throw
Throw the indicated exception
goto
goto/16
goto/32
Unconditionally jump to the indicated instruction
packed-switch
sparse-switch
cmpl-float
cmpg-float
cmpl-double
3x
cmpg-double
cmp-long
if-eq
if-ne
if-lt
if-ge
if-gt
if-le
if-eqz
if-nez
if-ltz
if-gez
if-gtz
if-lez
Branch to the given destination if the given two registers' values compare as specified.
Branch to the given destination if the given register's value compares with 0 as specified.
4x
aget
aget-wide
aget-object
aget-bool
aget-byte
aget-char
aget-short
aput
aput-wide
aput-object
aput-bool
aput-byte
Perform the identified array operation at the identified index of the given array, loading or storing into the value register
5x
aput-char
aput-short
iget
iget-wide
iget-object
iget-bool
iget-byte
iget-char
iget-short
iput
iput-wide
iput-object
iput-bool
iput-byte
iput-char
iput-short
Perform the identified object instance field operation with the identified field, loading or storing into the value register
6x
sget
sget-wide
sget-object
sget-bool
sget-byte
sget-char
sget-short
sput
sput-wide
sput-object
sput-bool
sput-byte
sput-char
sput-short
invoke-virtual
invoke-super
Perform the identified object static field operation with the identified static field, loading or storing into the value register
Call the indicated method
7x
invoke-direct
invoke-static
invoke-interface
Call the indicated method
invoke-virtual/range
invoke-super/range
invoke-direct/range
invoke-static/range
invoke-interface/range
Call the indicated method
neg-int
not-int
neg-long
not-long
neg-float
Perform the identified unary operation on the source register, storing the result in the destination register
8x
neg-double
int-to-long
int-to-float
int-to-double
long-to-int
long-to-float
long-to-double
float-to-int
float-to-long
float-to-double
double-to-int
double-to-long
double-to-float
int-to-byte
int-to-char
int-to-short
Perform the identified unary operation on the source register, storing the result in the destination register
9x
add-int
sub-int
mul-int
div-int
rem-int
and-int
or-int
xor-int
shl-int
shr-int
ushr-int
add-long
sub-long
mul-long
div-long
rem-long
Perform the identified binary operation on the two source registers, storing the result in the first source register
Ax
and-long
or-long
xor-long
shl-long
shr-long
ushr-long
add-float
sub-float
mul-float
div-float
rem-float
add-double
sub-double
mul-double
div-double
rem-double
Perform the identified binary operation on the two source registers, storing the result in the first source register
Bx
add-int/2addr
sub-int/2addr
mul-int/2addr
div-int/2addr
rem-int/2addr
and-int/2addr
or-int/2addr
xor-int/2addr
shl-int/2addr
shr-int/2addr
ushr-int/2addr
add-long/2addr
sub-long/2addr
mul-long/2addr
div-long/2addr
rem-long/2addr
Perform the identified binary operation on the two source registers, storing the result in the first source register
Cx
and-long/2addr
or-long/2addr
xor-long/2addr
shl-long/2addr
shr-long/2addr
ushr-long/2addr
add-float/2addr
sub-float/2addr
mul-float/2addr
div-float/2addr
rem-float/2addr
add-double/2addr
sub-double/2addr
mul-double/2addr
div-double/2addr
rem-double/2addr
Perform the identified binary operation on the two source registers, storing the result in the first source register
Dx
add-int/lit16
sub-int/lit16
mul-int/lit16
div-int/lit16
rem-int/lit16
and-int/lit16
or-int/lit16
xor-int/lit16
add-int/lit8
sub-int/lit8
mul-int/lit8
div-int/lit8
rem-int/lit8
and-int/lit8
or-int/lit8
xor-int/lit8
Perform the indicated binary op on the indicated register (first argument) and literal value (second argument), storing the result in the destination register
Perform the indicated binary op on the indicated register (first argument) and literal value (second argument), storing the result in the destination register
Ex
shl-int/lit8
shr-int/lit8
ushr-int/lit8
Perform the indicated binary op on the indicated register (first argument) and literal value (second argument), storing the result in the destination register
execute-inline
Executes the inline method identified by inline ID
Fx
invoke-direct-empty
Stands as a placeholder for pruned empty methods like Object.<init>. This acts as nop during normal execution.
iget-quick
Gets the value stored at offset in vy instance's data area to vx
iget-wide-quick
Gets the object reference value stored at offset in vy instance's data area to vx and vx+1
iget-object-quick
Gets the object reference value stored at offset in vy instance's data area to vx
iput-quick
Puts the value stored in vx to offset in vy instance's data area
iput-wide-quick
Puts the value stored in vx,vx+1 to offset in vy instance's data area
iput-object-quick
Puts the object reference value stored in vx to offset in vy instance's data area to vx
invoke-virtual-quick
Invokes a virtual method using the vtable of the target object
invoke- virtual/range-quick
Invokes a virtual method using the vtable of the target object
invoke-super-quick
Invokes a virtual method in the target object's immediate parent class using the vtable of that parent class
invoke-super/range-quick
Invokes a virtual method in the target object's immediate parent class using the vtable of that parent class.
misc
moves
method
literals
system
object
flow
conditional
transfer
logical
conversion
arithmetic
undefined
standard 4 => 4
/from16 16 => 8
/16 16 => 16
Dalvik Virtual Machine (android)

Android	1	http://source.android.com/tech/dalvik/dalvik-bytecode.html
	2	http://pallergabor.uw.hu/androidblog/dalvik_opcodes.html
.Net	1	http://www.ecma-international.org/publications/standards/Ecma-335.htm p355-469
	2	http://www.asukaze.net/etc/cil/opcode.html
Java		http://java.sun.com/docs/books/jvms/second_edition/html/Instructions.doc.html