HIGH-LEVEL FUNCTIONS:

```
        XOP     @NULL,n
```

NULL is a dummy address, not even used. n is the xop number assigned by Myarc.

All arguments are passed and returned via a bi-directional workspace. This will allow re-entrant coding within a multi-tasking system. Furthermore, it allows easy interface into other languages, and aids in the implementation of the DSR interface.

INDEX	FUNCTION	DESCRIPTION
00	OpenDialog	Opens one dialog
01	CloseDialog	Closes one dialog
02	SetPointerD	Changes the pointer sprite when within Dialog
03	ClearPointer	Resets the pointer sprite definition to normal
04	SetDialogTitle	Changes the current title of the dialog
05	MoveDialog	Changes the location of a Dialog on screen
06	AddGadget	Add gadget data to lists (does not display)
07	RemoveGadget	Removes gadget data from lists (not erase)
08	OnGadget	Disables a gadget (puts in ghost mode)
09	OffGadget	Re-enables a gadget (normal state)
0A	RefreshGadgets	Re-displays gadget image (hardly used)
0B	ModifyProp	Modifies the current parms of a prop gadget
0C	DrawImage	Draws one, or several linked images on...
0D	DrawBorder	Draws one, or several linked border on...
0E	PrintIText	Draws one, or several linked texts on...
0F	InTextLength	Calculates width of text in pixels
10	RemakeDisplay	Re-draws everything on screen (what for?)
11	SetMenuStrip	Attaches a menu strip to a window
12	ClearMenuStrip	Removes a menu strip from a window
13	OnMenu	Turns on a particular menu
14	OffMenu	Turns off a particular menu
15	ItemAddress	Derives the Address of a menu Item
16	OpenWindow	Opens one, or several linked windows
17	CloseWindow	Closes one, or several linked windows
18	IconizeWindow	Places window in next available icon slot

LOW-LEVEL (machine dependant) FUNCTIONS:

```
        XOP     @NULL,n+1
```

INDEX	FUNCTION	DESCRIPTION
00	SetVideoMode	Sets the current video mode
01	GetVideoMode	Returns the current video mode
02	SetCurPos	Sets the cursor position used in text modes
03	ReadCurPos	Returns the cursor position in any page
04	SetDisPage	Sets the current display page
05	GetDisPage	Returns the current display page
06	ScrollWinUp	Scrolls a text window up by line amounts
07	ScrollWinDown	Scrolls a text window down by line amounts
08	ScrollWinLeft	Scrolls a text window left by line amounts
09	ScrollWinRight	Scrolls a text window right by line amounts
0A	WriteCharColor	Writes a char and color (graphics) to screen
0B	ReadCharColor	Reads a char and color (graphics) from screen
0C	SetBorderColor	Sets the border color of the screen
0D	SetColPalette	
0E	SetPixelColor	Sets a pixel on (graphics 1-2), changes color
0F	GetPixelColor	Returns status (graphics 1-2), & color
10	SetVectorColor	Draws a line of color, pixel on (graphics 1-2)
11	ChipLINE	Uses chip command to draw line
12	ChipSRCH	Uses chip command to search for color
13	ChipPSET	Uses chip command to set pixel
14	ChipPOINT	Uses chip command to draw a point in color

Another task will be loaded to perform the "routine" housekeeping functions. i.e. mouse polling, gadget scanning, and system gadget execution. This task will broadcast messages to receiving ports denoted by the application.

```
        R2x = INTEXT pointer
        R3x = LeftOffset
        R4x = TopOffset
```

LeftOffset, TopOffset are added to LeftEdge, TopEdge of each Image structure linked to this call.

```
InTextLength    XOP     @NULL,n

        R0x = 000F
        R1x = INTEXT pointer
-------------------------------
        R0x = The returned width in pixels
```

This section shows all of the gadget routines.

Gadget Structure

```
GADGET  DATA    NextGadget - pointer
        DATA    LeftEdge, TopEdge
        DATA    Width, Height
        DATA    Flags:
        DATA    Activation:
        DATA    GadgetType
        DATA    GadgetRender - pointer
        DATA    SelectRender - pointer
        DATA    GadgetText - pointer
        DATA    SpecialInfo - pointer
        DATA    GadgetID
        DATA    UserData - pointer
```

Flags:
GadgHComp - If set, the bits within the gadget's select box will be complemented.
GadgHBox - A box will be drawn around the gadget's select box.1
GadgHImage - An alternate Image or Border will be displayed.
GadgImage - If this bit is set, the GadgetRender and SelectRender are Images, if reset, they are Borders.
GRelBottom - Set this flag if the gadget's TopEdge describes an offset relative to the bottom of the display element containing it. Reset this flag if TopEdge is relative to the top.
GRelRight - Same thing with the LeftEdge.
GRelWidth - Set this flag if the gadget's Width variable describes an increment to the width of the display element containing the gadget. Clear this flag if Width is an absolute value.
GRelHeight - Same thing with the Height.
Selected - Use this flag to preselect the on/off selected state for a toggle-selected gadget. If the flag is set, the gadget starts off being on and is highlighted. If the flag is clear, the gadget starts off in the unselected state.
GadgDisabled - If this flag is set, this gadget is disabled. The Gadget will be ghosted. Use OnGadget and OffGadget to change the current state.

Activation:
ToggleSelect - When this bit is set, the on/off state of the gadget (and its imagery) toggles each time it is hit.
GadgImmediate - Set this bit if you want the program to know immediately when the user selects this gadget.
RelVerify - "Release Verify", Set this bit if you want this gadget selection broadcast to your program onlyu if the user still has the pointer positioned over this gadget when releasing the select button.
FollowMouse - When the user selects a gadget that has this flag set, the

StringCenter - If this flag is set, the text in a string gadget is center-justified when rendered.
StringRight - If this flag is set, the text in a string gadget is right-justified when rendered.
LongInt - If this flag is set, the user can construct a 32-bit signed integer value in a normal string gadget. You must also preset the string gadget input buffer by putting an initial integer string in it.

This stuff should be referenced by the SpecialInfo pointer if proportional.

```
PROPINF DATA     Flags:
        DATA     HorizPot, VertPot
        DATA     HorizBody, VertBody
        DATA     CWidth, CHeight
        DATA     HPotRes, VPotRes
        DATA     LeftBorder, TopBorder
```

Flags:
AutoKnob - Set this if you want to use the auto-knob.
FreeHoriz - If this is set, the knob can move horizontally.
FreeVert - If this is set, the knob can move vertically.
KnobHit - This flag is set when this knob is hit by the user.
PropBorderLess - Set this if you want your proportional gadget to appear without a border drawn around its container.

This stuff should be referenced by the SpecialInfo pointer if string or int.

```
STRING  DATA     Buffer - pointer
        DATA     UndoBuffer - pointer
        DATA     BufferPos
        DATA     MaxChars
        DATA     DispPos
        *
        DATA     UndoPos
        DATA     NumChars
        DATA     DispCount
        DATA     Cleft, Ctop
        DATA     LayerPrt ??? - pointer
        DATA     Integer
```

```
AddGadget          XOP      @NULL,n

        R0x = 0006
        R1x = Window structure pointer
        R2x = GADGET pointer
        R3x = Position for the new gadget to go in the list

RemoveGadget       XOP      @NULL,n

        R0x = 0007
        R1x = Window structure pointer
        R2x = GADGET pointer

OnGadget           XOP      @NULL,n

        R0x = 0008
        R1x = Pointer to GADGET that you want enabled
        R2x = Window structure pointer

OffGadget          XOP      @NULL,n

        R0x = 0009
        R1x = Pointer to GADGET that you want disabled
        R2x = Window structure pointer
```

```
          R0x = 000A
          R1x = Pointer to GADGET where the redrawing should start
          R2x = Window structure pointer

ModifyProp        XOP      @NULL,n

          R0x = 000B
          R1x = GADGET pointer
          R2x = Window structure pointer
          R3x = Flags
          R4x = HorizPot
          R5x = VertPot
          R6x = HorizBody
          R7x = VertBody
```

This section shows all of the menu routines.

Menu Structure

```
MENU      DATA     NextMenu - pointer
          DATA     LeftEdge,.TopEdge
          DATA     Width, Height
          DATA     Flags:
          DATA     MenuName - pointer
          DATA     FirstItem - pointer
```

Flags:
MenuEnabled - If set, the menu may be pulled-down. This state may be changed by OnMenu or OffMenu.
MiDrawn - Indicates whether or not this menu's items are displayed to the user.

Menu Item Structure

```
MNITEM    DATA     NextItem - pointer
          DATA     LeftEdge, TopEdge
          DATA     Width, Height
          DATA     Flags:
          DATA     MutualExclude-hw
          DATA     MutualExclude-lw
          DATA     ItemFill - pointer
          DATA     SelectFill - pointer
          DATA     Command
          DATA     NextSelect
```

Flags:
CheckIt - Check will be displayed if Checked is set.
Checked - Set or reset for defaulted selection. Mutual Exclusion controls this.
ItemText - Set for text, reset for image. Sorry no borders!
CommSeq - If set, this item has an equivalent command key sequence held in Command.
ItemEnabled - If set, this item is enabled. If reset, item is ghosted.
HighFlags:
HighComp - Complements all of the bits (colors).
HighBox - Draws a box around select box.
HighImage - Alternate Imagery (SelectFill).
? HighNone - No highlighting.
IsDrawn - Hmm, should I do SubMenus.
HighItem - This flag is set when this item is highlighted.

```
        R0x = 0011
        R1x = Pointer to a window structure
        R2x = MENU pointer

ClearMenuStrip  XOP     @NULL,n

        R0x = 0012
        R1x = Pointer to the window structure.

OnMenu          XOP     @NULL,n

        R0x = 0013
        R1x = Pointer to window structure.
        R2x = MenuNumber value.

OffMenu         XOP     @NULL,n

        R0x = 0014
        R1x = Pointer to the window structure.
        R2x = MenuNumber value.

ItemAddress     XOP     @NULL,n

        R0x = 0015
        R1x = MENU pointer
        R2x = MenuNumber value.
-------------------------------------
        R0x = Returned address of a menu item
```

NewDialog Structure

```
NDIALOG DATA    LeftEdge, TopEdge
        DATA    Width, Height
        DATA    DetailColor
        DATA    BlockColor
        DATA    Flags:
        DATA    FirstGadget - pointer
        DATA    CheckMark - pointer
        DATA    Text - pointer
        DATA    BitMap - pointer
```

Flags:
DialogTitle - Will generate a title bar, and optionally allow you a system menu with move, and close capabilities.
DialogClose - Allow the user to initiate a close dialog message.
DialogMove - Allow the user to arbitrarily move the dialog on the screen.

GimmeZeroZero - If set, position (0,0) refers to the area just under the title bar. Also, clipping will be done to assure that gadgets will not be drawn on. Checking to make sure the dialog is not over-run.
Simple_Refresh - None of the dialog is preserved.
Smart_Refresh - Display is only preserved when another object overlaps it. Data clipped that is out-of-bounds is discarded.
Super_BitMap - Your own bitmap is used to update the display.
BackDrop - If set, the dialog will be in the background.
ReportMouse - Sets the dialog to receive pointer movements as x,y coordinates.
BorderLess - If set, no border lines are drawn around the dialog.
Activate - If set, the dialog becomes active when it is opened.
NoCareRefresh - If set, no refresh messages are sent to the program.

```
DIALOG  DATA    LeftEdge, TopEdge
        DATA    Width, Height
        DATA    MouseX, MouseY
        DATA    GZZMouseX, GZZMouseY
        DATA    Dwindow - pointer
        DATA    RPort - pointer
        DATA    BorderTop, BorderBottom
        DATA    BorderLeft, BorderRight
        DATA    BorderRPort - pointer
        DATA    UserData - pointer


OpenDialog          XOP       @NULL,n

        R0x = 0000
        R1x = NDIALOG pointer
----------------------------------------
        R0x = DIALOG pointer is returned.

CloseDialog         XOP       @NULL,n

        R0x = 0001
        R1x = DIALOG pointer

SetPointerD         XOP       @NULL,n

        R0x = 0002
        R1x = DIALOG pointer
        R2x = Pointer to sprite definition of pointer.
        R3x = Width - ignored on this version.
        R4x = Height - ignored on this version.
        R5x = Xoffset - placement of sprite relative to upper-left.
        R6x = Yoffset - placement of sprite relative to upper-left.

ClearPointer        XOP       @NULL,n

        R0x = 0003

SetDialogTitle      XOP       @NULL,n

        R0x = 0004
        R1x = DIALOG pointer
        R2x = Pointer to an ASCII-Z string to represent new title.

MoveDialog          XOP       @NULL,n

        R0x = 0005
        R1x = DIALOG pointer
        R2x = DeltaX
        R3x = DeltaY
```

This section contains all of the low-level (machine dependent) functions. Most of the functions are compatible with the 99/4A, but a few of them are chip commands for the MSX-VIDEO DATA PROCESSOR.

```
SetVideoMode        XOP       @NULL,n+1

        R0x = 0000
        R1x = Video mode
        0000    Text 1 mode
        0001    Text 2 mode
        0002    MultiColor mode
        0003    Graphic 1 mode
        0004    Graphic 2 mode
        0005    Graphic 3 mode
        0006    Graphic 4 mode
        0007    Graphic 5 mode
        0008    Graphic 6 mode
        0009    Graphic 7 mode

GetVideoMode    XOP     @NULL,n+1

        R0x = 0001
---------------------------
        R0x = Returned Video mode

SetCurPos       XOP     @NULL,n+1

        R0x = 0002
        R1x = Row number
        R2x = Column number
        R3x = Page number

ReadCurPos      XOP     @NULL,n+1

        R0x = 0003
        R1x - Page number
------------------------------
        R0x = Returned row number
        R1x = Returned column number

SetDisPage      XOP     @NULL,n+1

        R0x = 0004
        R1x = Page number

GetDisPage      XOP     @NULL,n+1

        R0x = 0005
-------------------------
        R0x = Returned page number

ScrollWinUp     XOP     @NULL,n+1

        R0x = 0006
        R1x = Row number of upper left corner
        R2x = Column number of upper left corner
        R3x = Row number of lower right corner
        R4x = Column number of lower right corner
        R5x = Character value for blank lines
        R6x = Number of lines to scroll

ScrollWinDown   XOP     @NULL,n+1

        R0x = 0007
        R1x = Row number of upper left corner
        R2x = Column number of upper left corner
        R3x = Row number of lower right corner
        R4x = Column number of lower right corner
        R5x = Character value for blank lines
        R6x = Number of lines to scroll

ScrollWinLeft   XOP     @NULL,n+1
```

```
          R1x = Row number of upper left corner
          R2x = Column number of upper left corner
          R3x = Row number of lower right corner
          R4x = Column number of lower right corner
          R5x = Character value for blank lines
          R6x = Number of lines to scroll

ScrollWinRight  XOP     @NULL,n+1

          R0x = 0009
          R1x = Row number of upper left corner
          R2x = Column number of upper left corner
          R3x = Row number of lower right corner
          R4x = Column number of lower right corner
          R5x = Character value for blank lines
          R6x = Number of lines to scroll

WriteCharColor  XOP     @NULL,n+1

          R0x = 000A
          R1l = ASCII character to write to screen
          R1h = Foreground/Background color for character in graphics modes
          R2x = Display page
          R3x = Number of times to write character and color

* Uses current cursor location

ReadCharColor   XOP     @NULL,n+1

          R0x = 000B
          R1x = Display page
-----------------------------------
          R0l = ASCII character read from screen
          R0h = Foreground/Background color of character

* Uses current cursor location

SetBorderColor  XOP     @NULL,n+1

          R0x = 000C
          R1x = Color of Border

SetColPalette   XOP     @NULL,n+1

          R0x = 000D
          R1x = Palette color ID
          R2x = Color or palette value to be used with color ID

SetPixelColor   XOP     @NULL,n+1

          R0x = 000E
          R1x = X coordinate of pixel
          R2x = Y coordinate of pixel
          R3l = Color to set - Foreground/Background for Graphics 1-2

GetPixelColor   XOP     @NULL,n+1

          R0x = 000F
          R1x = X coordinate of pixel
          R2x = Y coordinate of pixel
-----------------------------------------------------
          R0l = Returned color - Foreground/Background for Graphics 1-2

SetVectorColor  XOP     @NULL,n+1
```

```
         R1x = X coordinate of first pixel
         R2x = Y coordinate of first pixel
         R3x = X coordinate of second pixel
         R4x = Y coordinate of second pixel
         R51 = Color to set - Foreground/Background for Graphics 1-2

ChipLINE          XOP      @NULL,n+1

         R0x = 0011

ChipSRCH          XOP      @NULL,n+1

         R0x = 0012

ChipPSET          XOP      @NULL,n+1

         R0x = 0013

ChipPOINT         XOP      @NULL,n+1

         R0x = 0014
```