**Machine Intelligence Technologies**
Leading the way in android development. You don't program our machines, you teach them!

# KISS OS Documentation

Alpha Version

This document describes the KISS OS operating system

The hardware requirements for the operating system are as follows:

A modified propeller Proto board as described in a series of video clips on the machine intelligence technologies Web site.

A minimum of 2 MB of extended RAM which can be purchased from machine intelligence technologies.

A PS2 standard keyboard.

A NTSC video monitor.

The parallax Spin compiler.

A USB umbilical cord.

A parallax data logger module. $35

A PC compatible system.

A USB Thumb drive containing less than 4 GB of storage.
Using the FAT 16 format i.e. 16 bit file allocation table (FAT).
If the drive is not in FAT16 format it can be reformated on the PC to conform to the FAT16 format.

## TO THE HOBBYIST RESEARCHERS

We are providing this ALPHA PRELEASE version to you so that you may help us uncover the bugs and thereby provide you with the best possible product. Those who participate in this testing will be provided with the final

release version of the KISS-OS free of charge as well for the life of this project.

This operating system is not a general-purpose operating system but rather it is designed specifically to support the machine intelligence algorithms in an android robot. It contains all of the functionality necessary to manage the memory resources, the sensory resources, and the servo resources contained in the robot. The OS uses "WINDOWS" compatible files. It can create text files and display text files but it contains no text editor. All text editing must be performed on a PC compatible system. It can then read and display the text files. The thumb drive is used for a permanent storage device to save the neural information which the robot has accumulated. This provides a means to restore the ERAM during the boot up procedure.

This is an alpha version of the operating system. It is not fully functional and has some known bugs which will be listed in this document. This version is intended to gather as much technical feedback as possible before we finalize the design. The user community is therefore intimately involved in the design and all are considered fellow researchers. We therefore will value and respect your technical input.

# KISS OS

The KISS OS represents a new architecture for the propeller software. The operating system manages all of the resources in the propeller system such as: the expanded RAM (2MB ERAM board), the 32K main Propeller Hub RAM, the 2K COG RAMs (x8 COGs), the thumb drive memory (up to 4GB), the display, the keyboard, and any other I/O devices in the system.

The operating system represents a major departure from the Parallax PC umbilical approach. It is a stand-alone system which does not require a direct connection to the PC. The operating system kernel resides in the on-board 32K EEPROM module on the Proto board. Any other necessary driver programs reside on the thumb drive.

At boot up the kernel is loaded from the EEPROM and then the kernel loads in the other necessary drivers from the thumb drive into either the specific COG 2K RAM and/or loads other drivers and segments into the ERAM for higher speed access.

COG 0 is loaded with the SPIN interpreter and then COG 1 is loaded with the display driver. COG 2 is then loaded with the keyboard driver and finally COG 3 is loaded with the data logger driver. All other COGS are then loaded with the COG RES routine which remains resident.

The COG RES routine occupies the first 64 long (32-bit) locations and provides functions to access ERAM and main Hub RAM to issue requests to the operating system kernel. Each of these functions will be described later. This means that all COGs are either running the COG RES routine or specific I/O drivers.

Because most functions are performed through calls to the operating system, it is relatively easy for the user to write their own COG routines. The operating system performs most of the more complex tasks.

An example program is provided to show the user how to interface with the operating system. While the operating system has been designed specifically to support machine intelligence functions, it is generalized enough to be used in a wide range of applications. Some debug commands are also included in the operating system to help the user debug their programs.

## Installing the KISS-OS:

The KISS-OS kernel is distributed as a precompiled binary file. To install this OS kernel follow these steps:

1. Connect your prop proto board to a USB port on your PC. Connect at least one ERAM board to the prop proto board. Connect the data logger module to the prop proto board and be sure that a flash/thumb drive with

4GB of storage or less is installed.

2. Turn on your proto board power slide switch to the second position.

3. Open the Parallax SPIN compiler in a window on your PC.

4. Type this dummy program into the blank area.

```
PUB Dummy
  return
```

5. Press the "F8" key on your PC keyboard. A window similar to the one shown on page 57 of your propeller manual will appear.

6. click the "open file" button at the bottom of the window. Now load the "KISS_OS.eeprom" file into this window. This is the KISS-OS kernel file that you will load into the EEPROM on your prop proto board.

6. Click the "Load EEPROM" button on the bottom of the window and wait for the load to complete. The system will then begin it's automatic boot up procedure and test the ERAM and flash drive for proper operation. Then it will display a sign on message similar to this:
"CogID = 4 KISS_OS0.2L6 Cmd = "

The current parallax SPIN compiler design does not consider the installation of this type of "STAND ALONE" software system so we will have to use a work around procedure to get the operating system kernel installed.

After the OS kernel is installed the USB umbilical cable can be disconnected. All other software modules will be loaded via the thumb drive connected to the data logger.

The Software module will be compiled on the PC and then saved to a USB thumb drive. Then this USB thumb drive will be disconnected from the PC and connected to the data logger. The Software modules can then be loaded using the RUN command.

The only time your DROID brain will need to be connected to the PC is when you wish to load a new operating system kernel into the EEPROM module on your PROP Proto board. Once the operating system kernel is installed and running it will test the vital resources such as the data logger module and thumb drive, and the ERAM. The checks will determine if these are installed and operating properly.

KISS OS (operating system) Kernel Functions

The commands can be ended with the "end" key and block commands use the "enter" key to proceed with the next block. An example would be the "DB" command which displays main hub RAM and 8 byte blocks. The "enter" key is used to display the next block and the "end" key is used to terminate the commands.

GA: Show MRAM label addr

GCA: Show CRAM label addrs.
These are the entry points for
COG res functions.

MTE: Xfer MRAM To ERAM

CRC: Type a Cmd String to show
compression code.

SDL: Start Datalogger

DIR: List Current Directory

DB: Show MRAM Block

M: Disp/Chg MRAM Location

E: Set/Clr DIRA

H: List cmds (NOT IMPLIMENTED)

FTC: File (bin) to CRAM

CTF: CRAM (bin) to File

FTE: File (bin) to ERAM

ETF: ERAM (bin) to File

FTM: File (bin) to MRAM

MTF: MRAM (bin) to File

DF: Show txt file

DELF: delete file

RUN: Load a binary file into a COG
execute it.
All COG programs must begin at location " 64 "
because the COG resident routine uses the first
64 locations from 0 to 63. The first statement in
all COG programs must be "ORG 64" and the
last statement should be " FIT ". The operating
system will supports program segmentation for
programs larger than COG RAM via COG
resident calls. Use your KISS Debugger to test
your code for now. This will be corrected ASAP.

TTF: keybd txt to File

# Machine Intelligence Technologies

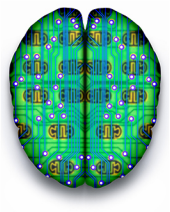
Leading the way in android development. You don't program our machines, you teach them!

Known bugs

1. A zero length file on the flash drive can not be deleted and causes the data logger to fail. A work around is to to plug the flash drive into a PC and delete the file.

2. the "RUN" command does not work properly. A new OS kernel will be sent as soon as this is fixed.

3. Virtual memory COG res functions do not work correctly. Most programs will not need these functions as long as the program fits into COG RAM. Programs that need to be longer can be segmented by the programmer.



Dr. James O. Gouge
Machine Intelligence Technologies
588 CR 315
Stockdale, TX 78160
www.machineinteltech.com
support@machineinteltech.com