FLY8
Flight Simulator

Eyal Lebedinsky

eyal@eyal.emu.id.au

FLY8
Flight Simulator

Eyal Lebedinsky

eyal@eyal.emu.id.au

1. Introduction

This program is a flight simulator. It puts more emphasis on the dynamics than on the cosmetics: just wire-frame. It can run on many machines as it was written for portability. The distribution includes support for msdos, mswin and unix (used to work on Amiga but I have no access to one anymore). Other ports were done but I did received any sources back yet. On Intel based systems a 386DX is the minimum for any decent performance. A fast video controller is a boon as the program, when running on a 386DX/40Mhz, spends 70-80% of its time pushing pixels. On non-intel machines you can try and see if it is fast enough...

As of this release the msdos version requires a 386 or better. If I get enough requests then I will compile it for 286 but it is too slow on such machines most of the time anyway.

The program was written for fun. I borrowed ideas from everywhere and hope to hear some more. The design is based on a program I wrote more than 20 years ago at uni (the Technion). I had an excellent coach (Danny Cohen) and I still have fond memories of those times. But now my computer has more than 24Kbytes of memory! so Fly8 is written in C (Fly8 was the name of the last version of the original program dated 12-JAN-1974, it was written in PDP15 assembly - macro15 - for a VT15 graphics processor).

The actual purpose of this program is to give me an opportunity to experiment with various aspects of flight simulation, but mostly with (1) the HUD symbology (or, more generaly, with man-machine interaction) and (2) the studying of basic aerodynamics (as well as general real time simulation techniques). This explains why there are so many HUD options and such a proliferation of flight dynamics models, as well as why the simulation parameters are user definable as input files at run time. Of course, the experimental nature of the program means that it must be distributed in source form.

On the PC the basic screen drawing uses the standard Microsoft graphics library. It is OK but not very fast; the main advantage is that it will support most video adapters. The fast graphics driver was built from the routines from DJGPP with much personal additions. The flight dynamics was influenced by an SGI program I saw and ACM. The timer routines come from a microsoft journal article, the user-input routine (notice how you can use arrow keys etc? use up-arrow to retrieve history.. I will document it one day) comes from DDJ (or was it CUJ? author name is Bob Bybee). Well, I avoid re-inventing wheels unless it is fun. The program compiles with Microsoft C, Borland C, gcc on a friend's Amiga, Sun and Linux and I hope on other platforms; it is written to be portable. It runs under MSDOS, MSWindows, Amiga and unix.

What? what? WHAT? you want to see some action? OK. just skip to the next chapter then come back.

The full set of commands is detailed in the 'commands' chapter. Here we will look at the program areas in general.

There are two rather distinct kinds of commands that one uses: commands that drive (fly) the game and commands that configure, set options and so on (which are used with less urgency). It was attempted to get the important commands into the keyboard (a one keystroke command) while the others go into the main menu system (accessed with the Esc key). Some of the urgent commands may bring up a menu which you

may ignore if you know the keystokes.

The urgent commands will control the vehicle flight and the other subsystems (radar, HUD, HDD, weapons etc.). You will notice early that the program lacks the traditional instrument panel: it is intended to be driven from the HUD and other digital displays.

The vehicle is also driven by a pointing device (a mouse or, preferably, a joystick). It will run off the keypad when you have no such device. The pointer is used only for steering control although the buttons can be mapped to auxiliary functions.

The display area is typicaly divided into the main view and a number of secondary Head Down Displays (HDD). The design has a dozen or so on-board instruments that generate visual data; you select which ones should be displayed on which HDD. The main view is what you see through the cockpit. The HUD can be overlaid onto this view (as is the case with a real plane). Other data may also be shown here for convenience.

One instrument is designated as an alternate main view (use the 'v' command to see it). The 'windows' menu handles the screen format and configuration.

The program generates various messages as it goes along, these will appear at the bottom of the main view and stack up. Each message has a time-out for deletion but you can use 'c' to clear the lot. When the program needs user input it will open a prompt line at the very bottom of the main view (in magenta color) where your data will show. You can use the normal editing keys while entering data here - previous entries are accessible with the up/down arrows. See 'input line editing' later.

You may find some of the commands/options strange (if not outright insane); this will be related to my taste or (mostly) to much history and quick fixing that did not completely settle yet. I have looked at other programs (like F3, JF2, ACM and SGI f.s.) but this was after the first version of this program was finished, so some good ideas missed the bus this time. In the future I hope to polish the user interface (especialy after other people get to use it and express an opinion).

Being as the program is still evolving you will find some areas less complete than others. I hope that there is enough of it to make it useful. I expect to see contributions (of ideas and code) from other people; I will continue to develop the program (at least for a while) and would like to see it take it's own path in life [heavy stuff :-)].

2. Installation

On most environments all you need to do is unpack the distribution archive into your prefered directory and you are set. Make sure you un-archive the parts such that the directory structure is maintained. Refer to the README file for specific instructions, you will probably want to enable support for your graphics card (on MSDOS).

Another thing you will want to do is select your prefered pointing device (the joystick is best). Examine the examples in fly.ini and read the later chapter about the options.

3. Quick start

In this chapter the symbol '@' is used to denote the Enter key. It will give you a feel for what the program is like. With the program installed, type

```
fly8 z5
```

This starts the program in a demo setting and is useful to see if all is OK. It is also great as a screen blanker :-)

If the fly.ini options are correct then your plane will take off and start looking for action. Some messages

are displayed during startup - these will disappear after a short while. The screen will show a simple view of the runway, a ground grid (in gray) and an overlaid HUD.

If there is no picture, try hitting 'Enter'. Then try 'Esc' 'x' 'y'. If no luck then kill it (or reboot on msdos, I guess). Now check the fly.log file which may have an error message in it.

On msdos it was found that with some accelerator cards the program hangs (don't know why, I use the MicroSoft C graphics library and most advanced cards should emulate VGA). Try installing the correct video VESA bios.

Do not run this program in a dos window of MSWindows or OS/2, use the MSWindows version for MSWindows (although it probably will work fine, there is a chance of locking your machine up by not doing so). In a full window OS/2 session it was reported to work OK.

The scene will include you and 5 other planes (drones). Your auto-pilot will track and shoot the drones. As they are shot down, new ones take off. To take control back from the autopilot hit Shift-C. Now use the joystick (or whatever input device you choose) to fly the plane. One mouse or joystick button shoots (same as F1). When the target is in the correct position the autopilot will shoot (unless you tell it not to with 'k'). The idea is to fly your plane so that the target is inside the aiming reticle (the small circle) and then shoot. A SHOOT cue will flash when your aim is correct.

You may want to know the current settings (like throttle, flaps etc.) Turn on the 'panel' display (Esc hud/hdd/panel). The 'Esc' key activates the menu system. Most selections can be used as a short sequence of the associated letters; the last sequence can be entered as Esc-u-d-p-Esc. The hud menu is also directly available as the 'u' command so you can key u-d-p-Esc instead.

This is how you fly the plane: moving the joystick sideways will start the plane rolling. The further you move the stick the faster the roll. Once you center the stick the roll will stop. In order to fly level you need to roll the other way until you are level and then center the stick.

Moving the stick away from you will push the nose of the plane DOWN, pulling the stick back will pull the nose up. When the stick is centered the plane will maintain it's climb angle (pitch).

So far we rolled and pitched but we did not yet turn. In order to turn one needs to use both controls. To turn right, first roll to the right, then pull the plane 'up'; at this point 'up' is actualy 'right'. Remember that the joystick controls the plane relative to itself (the pilot if you wish) and NOT relative to the ground. Once you turn in the desired dirction you can roll left to resume level flight in the new heading.

Because the plane has weight, if you roll and start turning the plane will also start falling down (the wings no longer support the full weight of the plane) so a realistic turn will call for a moderate roll and not a full 90 degrees. The harder you will pull the stick, the faster you will turn and the larger roll you should execute to maintain level during the turn.

You probably do not want more instructions at this 'quick start' section, not to mention that I never flown a plane and am not qualified for much in the way of flight instruction. Any volunteers for writing a flight manual chapter?

Knowing how to fly the plane is not enough, you also need to know how to participate in the game (fight). Actualy, in order to start winning you will need to be able to fly without thinking, you will need your logic powers to control your situation and plan your moves.

Your strategy is to avoid being hit and try and kill all enemy planes. There it is, as simple as can be. I wonder why people fill books with chat about Basic Flight and Air Combat Maneuvering :-)

Now a quick look at controlling other equipment. For takeoff, release the wheel brakes ('b') and set full throttle ('1') or even light the afterburner (a few hits on the '.' key). You may want to set the flaps (a few ']', then reset with '[') but it is not necessary. At a speed of 150-200 pull the nose up gently (not more than 10-15 degrees) and wait for takeoff. After you gain some high (but rather soon) retract the landing gear ('g'). There you are in the air. Do not try a sharp turn too soon as you may loose altitude and hit the ground, unless you are experienced with this sort of thing.

To land, reduce speed and approach the runway at a steady descent. Just before touchdown reduce the descent to the bare minimum (don't forget to lower the landing gear ('g') in time but not too early). Once on the ground engage the speed-brakes ('+') and reverse thrust (just hit 3 until you have -100 power showing; this is not always available). When your speed is low enough engage the wheel brakes ('b') and towards the end idle your engines ('0') and release the speed brakes ('+'). Once you are stationary on the ground with the engine idle your fuel will start to be replaced and finally your wepons will be replaced and your damage will be reset.

When flying, use 'r' to switch the radar on and off, use 'w' to select your weapon and use 'v' to switch to a map view of the world (with you at the center) and back to normal view.

This should do you for starters.

While we are here, do 'Esc' 'i' 'Enter'. Some numbers will show at the top of the screen. The first is the total time (in milliseconds) for one frame, the second is the video-drawing time. If the total time goes over 100 often then you should buy a faster machine (actually, if it is the second numbers that dominates, a faster video card may be a better investment). If it stays under 60-70 then all is fine.

On MSDOS, if you have a TsengLab ET4000 based card then try running

```
fly8 dvgrfast:et4k Vgret4k m800x600x256 z5
```

and if it works you will notice the speedup. Other cards are supported, check the readme for details. The standard Microsoft library does not do double buffering above 640x350x16 (even in C8); don't know why - the memory is there. You may wish to edit the file 'fly.ini' with your preferred setup so that you will not need to specify it in the future.

In the more general case, if you have a VESA VBE compliant BIOS, then try:

```
fly8 dvgrfast:vesa Vgrvesa1p m800x600x256 z5
```

it will perform well if it works. If you can load your VESA BIOS into ram then do so, it often is much faster. If you do not have a VESA BIOS supplied with your card then look around for one. There is at least one package (the universal vesa driver UNIVBE) which should work on most cards.

To exit hit 'Esc' 'x' and 'y'.

4. Commands Reference

Fly8 commands are one keystroke each but some expect some data or options to follow, which may bring up a prompt or a menu.

Some commands are used only when the 'keypad' is selected as the pointing device for flying. Otherwise the commands are grouped here by their physical location and organized alphabetically.

The program usually runs with the NumLock engaged which means that the keypad keys duplicate the digits 0 through 9 and the period '.'.

There is no current facility to redefine the usage of the keys but the keyboard macros can be of use here. See under F7/F8.

4.1 Alphabetic Keys

Most commands toggle their function on/off, some cycle through modes.

A select aiming reticle mode. (cycles) [obsolete] This is used for experimenting with various LCOS formulae and will be gone once it settles down. At this point the calculations are based on linear motions, it should be modified to follow an arc instead. The setting is shown as 'Mn' in the 'modes' screen in the 'radar' part.

0 no acceleration correction

1 0.5 second correction

2 1.0 second correction

3 t/2 seconds correction (t=time to impact) (default)

4 t*t/2 seconds correction

b Wheel brakes (toggle). Can be applied at any time but only effective when on the ground. In reality these should not be operated at high speed (use speed brakes and reverse thrust for initial slow down).

c Clear text area. By now there is no text area anymore so this is used for the auxiliary function of removing all outstanding messages. If the windows boundary got dirty then use the Screen.Clear menu option to completely redraw the screen.

C Chase the locked target (toggle) This is the auto-pilot mode. If there is no target (or the radar is turned off) then the plane will wander around the airport perimeter. If there are ground targets then these will be chased and the plane will crash! This mode is activated by the command line option 'z'. If the kill-mode is enabled ('k') then the auto-pilot will fire at the target.

D Descend the parachute. After you eject it may take a while to get to the ground. You can pass the time by looking around (use the arrow and F5/F6 zoom keys), or you can jump to the landing phase with this command. If you land before your plane crashes then you will have to wait (a WAIT notice will be shown).

d Declutter the HUD. This will remove some HUD items that get in the way when you are in a dogfight. You can use the Hud.Parts menu to selectively remove HUD items.

E Eject. If your plane is not dead, your ejecting will send it crashing. You are then on a parachute descending slowly. Use 'D' for a quick descent.

f Select radar target-acquisition mode (cycle). Controls the manner by which the radar selects a target as explained later in the relevant section.

g Landing gear up/down (toggle). You can't raise the gear while on the ground.

h Help (also '?')

i Intelligence: identify all visible targets (toggle) This is what makes this program better than the real thing... in this mode all visible targets are identified even when outside of the HUD area. You need to have HUD data mode enabled to get info about the targets. The aiming reticule will not show when in this mode.

j Radar sees only real pilots (ignores drones) (toggle). The program can generate drones for target practice. If you want the radar to ignore these and only show (and select) real planes then use this command. There are other pilots only when you are networked.

k Kill- auto shoot when ready (toggle). When a target is in range and in correct position the auto-pilot will flash a SHOOT cue. In this mode the selected weapon will be fired at this point. The radar must be active with a locked target (you should see a piper).

l Lock target (toggle) The radar can operate in two modes, either it continuously selects a target according to the designated target acquisition mode or it stays locked on the current one. In the locked mode, the first selected target will be locked on and no more searching will be done. When not locked the target designator will be shown as a broken box. You can use the un-lock command (usually attached to one of the buttons and to the space-bar) to release the current target and acquire a new one.

m Show general program status. (toggle) This replaces the numerous mode indicators that planes have.

M Set the virtual buttons shift mode. This command is expected to be entered from a macro so NO prompting is done to the user, you will be now typing blind. You will enter a list of characters which will change the buttons mode. The first unrecognized character will terminate this command (it is good practice to use the Esc key for that).

- 0 The following commands will enable the designated mode.
- 1 The following commands will disable the designated mode.
- 2 The following commands will toggle the designated mode. This is the default at the start of this command.
- a Set the Btn-Alt mode.
- c Set the Btn-Ctrl mode.
- p Set the Btn-Special mode.
- s Set the Btn-Shift mode.
- x Disable all modes and enter ’enable’ mode (same as ’1’ above).
- * Quit without changing the modes.

o Observer select. See the world from another object’s point of view. Also useful for just a list of the objects. The list of current objects is presented. Hit Enter to abort or select an object. The selection ’c’ will return you to your controlled plane. ’l’ will select your target as the view object (if there is a target). In the list, piloted planes have a ’c’ and your target has an ’l’. Note that you cannot use this command if networking is active. Also as objects come and go, by the time you select an object (by a sequence number in the list) it may have moved up the list - you end up with the wrong object. The command is not considered important enough to make it any more robust.

O As ’o’ but also shows minor objects.

p Pause. Will not work when net is active (toggle). The "Pause" message tells you that you are. Use ’p’ again to resume (which should clear the "Paused" message).

P Report a button press. The next character must be a button name.

q Quiet (sound) mode. (cycle) Sets the sounds level to one of the following. Note that the independent ’aural alarm’ option can be used to turn the nagging alarms on/off.

- 0 no sound.
- 1 only shoot/hit/alarms sounds (default).
- 2 all sounds and effects. For now the only effect is the engine noise.

r Activate radar (toggle)

R Report a button release. The next character must be a button name.

s Will show some stats. Following that list is given a summary of objects currently active (does not include the landscape objects).

S Resupply plane: full stores and fuel, reset damage. This is a cheat!

u Hud configuration. see ’hud commands’ later. Identical to the main menu ´hud’ function.

v Select normal/alternate view (toggle). Will bring the designated alternate instrument into the main view. The alternate view is defined through the window configuration menu. Note the view name at the top right corner of the screen.

w Select weapon (cycle).

W Remove all weapon stores. The plane manoeuvres better this way. You can still use the weapons, the counters will just go negative.

x

Calibrate pointer. Mainly for joystick. When Fly8 starts it sets the joystick to an 'uncalibrated' mode. You now need to play the stick to all of it's positions, which means move the x and y to the edges, move the throttle to both ends and move the FCS hat to all positions. Finally leave the stick centered, the throttle at the position which you want to be the point where AB starts and leave the hat centered. Now hit 'x' and the program will re-calibrate itself.

After playing for a while you can again hit 'x' but you will have to take your hards off the stick and reposition the throttle before doing so (and you MUST have played for a while so that the full extent of the controls was used).

4.2 Symbol Keys

These are the rest of the keys on the main keyboard. For clarity each key's name is spelled out. If it is allocated then the function follows.

[back del]	
[escape]	invoke the menu system
[space]	release radar lock
‘ [grave accent]	
- [hyphen]	see keypad '-'
= [equals sign]	
\ [back slash]	used to reset the trim to nill.
[[l-bracket]	less flaps
] [r-bracket]	more flaps
; [semicolon]	
’ [quote]	
, [comma]	
. [period]	see keypad '.'
/ [slash]	see keypad '/'
˜ [tilde]	
! [bang]	shell to system. May not restore some environment parameters and pallette. Use 'exit' to resume. Not implemented on windowed enviroments - just open another window if you need so...
@ [at symbol]	
# [hash]	
$ [dollar]	
% [percent]	
ˆ [caret]	
& [ampersand]	

* [asterisk]	see keypad ’*’
([l-paren]	less wheel brakes.
) [r-paren]	more wheel brakes.
_ [underdash]	
+ [plus]	see keypad ’+’
| [pipe]	
{ [l-brace]	less spoilers
} [r-brace]	more spoilers
: [colon]	
" [double quote]	
< [less than]	less speed brakes.
> [greater than]	more speed brakes.
? [question mark]	help

4.3 Keypad

The keypad is a collection of keys that replicate the main keyboard. These are described as three groups by function.

The following keys surround the numerical keypad and are not affected by the Shift key.

-	view right (+45 degrees)
*	view ahead
/	view left (-45 degrees)
+	speed brakes (toggle). Note that the Speed brakes take time to deploy, it’s status is shown on the control panel in percent of full extension.
(Enter)	unallocated

The rest of these keys must have NumLock on.

These first four keys respond only if the keypad is your pointing device.

8 (up)	pitch (pull nose) up
2 (down)	pitch (push nose) down
6 (right)	roll right
4 (left)	roll left

The following commands extend all pointing devices capabilities:

5 (center)	center ailerons and elevetors, like centering the joystick. Useful when using a mouse (or trackball): will move the reference point to where the mouse is at this moment.
7 (Home)	stop rolling. Levels the plane. For quiche eaters.
9 (PdUp)	more (+5%) power
3 (PdDn)	less (-5%) power

1 (End)	mil (dry) (100%) power
0 (Ins)	zero (0%) power
. (Del)	after burner up (+1 unit which is 20% of full scale)

4.4 Special keys

These keys are a group of six on most keyboard but can also be duplicated using the Shift key and a numerical keypad key.

PageUp	level (heading 0, pitch 0, roll 0)
PageDn	reset coordinates to zero (back to base)
Home	unallocated
End	unallocated
Insert	unallocated
Delete	unallocated

4.5 Function keys

The function keys are normally used in plain mode (no Shift, Alt or Ctrl). When the menu is on the upfront, the left column ten selections are accessible with F1-F10 while the right column uses AltF1-AltF10.

F1 shoot. Usually also mapped the mouse left button and the joystick trigger button.

F2 rudder left

F3 rudder center

F4 rudder right

F5 zoom in (more detail, narrower view, eye further from window).

F6 zoom out (less detail, wider view, eye closer to window).

F7 Macro/HotKey definition. Any key can be used for a macro name (except F7/F8). If you define a macro for a HotKey (Ctrl- and Alt- 'a' thru 'z') then it can be played back with one keystroke. Other keys are played using the F8 key. If you use a Macro during recording then the Macro will be recorded. If you later re-define this Macro then it will affect any other Macros that uses it. During macro expansion there is a limit of 16 levels of nesting. There is no capability for Macros definition editing.

define	F7<macro-key><keystrokes>F7 If the key is already defined then you are warned of the re-definition. You may abort at any stage (F8F8) and the original definition will remain. If you hit F8 during recording then you are prompted by Abort/Cont/Quote? to which you may respond by F8 (abort the recording), F7 (ignore the F8 and continue recording) or any other key (the key will be recorded with the F8 expecting it to be another macro).
delete	F7<macro-key>F7 It is not possible to record a null macro.

F8 F8<macro-key>

HotKey play	<HotKey> Hot keys are Ctrl-a through Ctrl-z and Alt-a through Alt-z.

F9 zoom in (more detail, narrower view, eye further from window) in the external view window (default is the radar map).

F10 zoom out (less detail, wider view, eye closer to window) in the external view window (default is the radar map).

F11 unallocated

F12 unallocated

4.6 Alt keys

Alt-Arrows: see below.

Alt-a thru Alt-z are reserved for user defined HotKeys.

The other keys are unallocated.

4.7 Ctrl keys

Ctrl-Arrows: see below.

Ctrl-a thru Ctrl-z are reserved for user defined HotKeys.

The other keys are unallocated.

4.8 Arrow keys

up	turn gaze (head) down All turns by 5 degrees
down	turn gaze (head) up
left	turn gaze (head) left
right	turn gaze (head) right use '*' to restore normal front view.
CTLup	trim nose down.
CTLdown	trim nose up.
CTLright	trim rudder right.
CTLleft	trim rudder left.
ALTup	debug (varies).
ALTdown	debug (varies).
ALTright	debug (varies).
ALTleft	debug (varies).

5. Menus

A menu has a list of options, each associated with a key and a function. To select a function use the Up/Dn arrows to highlight it and then Enter, or directly press the corresponding key. When on the up-front, the associated letters are NOT shown but are recognized; use the CtrlF-keys to select left column functions and AltF-keys for the right column. See under HDD later.

Menus can be nested, in which case the previous selections are listed first (in a staggered fasion and highlighted) followed by the current menu. The selected option is highlighted (white) while the others are

displayed in gray.

The Esc key brings up the top menu but later is used to abort a menu. During menu navigation use these keys:

Esc
: aborts the menu

Enter
: accept the current selection option

UpArrow
: select previous option

DnArrow
: select next option

other
: select the corresponding option

If a command is invalid (top/bottom of list or undefined option) then a beep is emited and the keystroke is ignored.

The menu system is changing rapidly so the following may be incomplete.

Some other commands may pop up a menu, in which case it behaves in a similar way.

If you have the up-front instrument active then the menus will appear on it rather than on the main menu. This is a stupid attempt to make your interaction look similar to what happens in a modern fighter. It is often the case that the pilot has a panel which can display about 10 alpha-numeric words in bright white (daylight readable). These will usually use special LEDs and a more elaborate font than the 7-seg digits (9 segment?). There are pushbuttons beside these words. The words are arranged in two columns, each with 10 words, with 10 buttons on the right and 10 on the left. As you press a button new information is displayed. Other planes use a real CRT but still have a 5-buttons arrangement (mostly on all four sides, totaling 20).

In Fly8 the up-front device has two columns of words, each column can show 10 words, each word can be 10 characters long. There are 10 buttons on the left (Ctrl-F1 through Ctrl-F10) and 10 on the right (Alt-F1 through Alt-F10). Use these keys to make selections. You can still use the normal command letters (if you know them, as they are not displayed here) as well as move about with the Up/Down keys: a small dash between the key number and the text identifies the currently selected option.

5.1 Top Menu

Exit	quit the program confirm with 'y' on subsequent menu
Help	toggles the help screen on/off
Pointer	select pointing device
Screen	screen options.
Windows	set windows configuration
Info	select stats info level
Emit	create some random objects or remove them
Hud	configure HUD
Net	networking commands

Options	set global program options
Auto	selects some autopilot options
Debug	set debug options
Buttons	Allows to control the functionality of the pointer buttons. Usuaslly you set these options at startup time. The 'B' command will too bring this menu up. To actually report a button press/release you use the 'P', 'R' and 'M' commands.
Command	Allows to control the behaviour os the simple keystroke commands which by default have a 'Toggle' action (like 'w', 'g' etc.).

5.2 Pointer Menu

A list of all available pointing devices is offered. Select one. All systems have a keypad device, most have a mouse and the PC has a joystick. You will be asked to specify pointer options - read about it later in the command line options section.

5.3 Screen Menu

Screen.Palette
: program the palette

Screen.Colors
: assign colors to visual elements

Screen.Stereo
: select a stereo mode

Screen.Dbl Buff
: set buffering mode to double/single. A message is posted on the new mode. Double buffering is not supported in all environments and in all modes, and in some cases it may be significantly slower than single buffering.

Screen.Blanker
: toggle screen blanker mode. Borders and some fixed data are not shown in blanker mode.

Screen.HUD pos
: select it the HUD is focused at infinity (default) or on the HUD face. Only used for experimenting, rather useless to change the default.

Screen.Solid Sky
: this option causes a background to be drawn with sky/earth colors. By default the background is left in the window's background color and the sky is shown as a sequence of blue lines.

  Using this option mean that the program does much more wrawing (the whole image has its background painted) so it is not recommended for system without a fast graphics facility.'

Screen.clear
: Will clear the screen, redrawing the borders and background. Useful to clean after some deficiencies in Fly8 where sometimes the borders are overwritten.

5.3.1 Screen.Palette Menu

A set of colors is listed. Select one for modification. You will then be presented with a number of adjustment options:

Brighter	intensify the color
Darker	reduce color intensity

New set color to a desired RBG value

Restore restore color to original value

5.3.2 Screen.Colors Menu

A list of visual componnents is listed. Select one and then the color palette menu will show - choose the color to assign for this element.

5.3.3 Screen.Stereo Menu

select a mono/stereo mode:

Screen.Stereo.Mono
: standard mono mode

Screen.Stereo.S'Scopic
: side by side stereoscopic images. Use the 'recerse' option if you prefer the cross-eye stereoscipic view.

Screen.Stereo.RedBlue
: red/blue composite (needs colored glasses)

Screen.Stereo.Alternate
: alternating left/right images (needs shutter glasses and '-s' command line option)

Screen.Reverse
: toggle reverse-stereo mode (swap Left and Right images)

Screen.Paralax
: set stereo inter-occular distance. Initialy set to 12 units. Each unit is 1meter/256 (about 4mm).

5.4 Windows Menu

A number of window configurations are offered. This defines how the screen is split into main and auxiliary display areas. After the selection the screen remains active so that you can select the ´configure' option immediately.

You can also set the foreground, background and border color for each window with this menu.

You will be prompted with a "Which window?" message, and all the windows will have a number and type in their middle. Enter the number and a selection of options for theis window will be offered.

Windows.configure
: This will call up the windows layout setup menu.

Windows.bg color
: Set the foreground color for one window. You will be prompted to select a window and then to select a color.

Windows.fg color
: Set the border color for one window. You will be prompted to select a window and then to select a color.

Windows.bo color
: Set the background color for one window. You will be prompted to select a window and then to select a color.

Windows.full
: the whole screen is one window

Windows.landscape
: a wide main view with three windows below: stores on the left, radar map on the right and a rear vision mirror in the middle.

Windows.portrait
: a square main view with a column of two windows on the right

Windows.square
: a square main view with a column of three square windows on the right

Windows.wide
: a wide main view with four square windows below

Windows.panorama
: An experimental format where the center view is accompanied by a left and right views which meet on the edges.

Windows.ether
: a new window configuration that is being developed at the moment.

5.4.1 Windows.Configure Menu

The plane has a number of on-board instruments, each one has a visual representation that can be shown on one of the active displays. This menu is used for defining which instrument is to be shown on each of the displays. You can enter 'x' as the window number and this will set the type of the external view (accessed with the 'v' command, by default is the radar).

front	this is a forward looking camera.
none	designates the HDD as unassigned
rear	a rear viewing camera
map	a map of the area from above with you at the center and north is up.
radar	as 'map' above but the plane's current heading is the 'up' direction. Gives better situation awareness.
target	a target following camera.
pan	another target following camera that is less stable (more real?)
gaze	a view of my plane from a fixed relative point
chase	a view of my plane from a point that chases my path.
follow	as 'chase' but the view is always level (never rolls).
hud	this is the raw HUD display
up-front	an alphanumeric display used for pilot interaction
panel	a digital data and warning display panel.
right	the right view of the 'panorama' configuration.
left	the left view of the 'panorama' configuration.
stores	This is a summary of the vehicle's status showing weapon selection, throttle and engine state, fuel and other engaged features (gear, brakes etc.).
lamps	a digital board of lamps that can be on, off or blinking in red or green with a legend on each. It is used by default by the Ether configuration.
mirror	Like a rear view but through a mirror. It is by default set up as a wide angle mirror.

5.5 Info Menu

Select the stats info level. This info is shown on the up-front HDD but can (optionally) be overlaid on the main view.

Info.off
: do not overlay ’info’ on the main view

Info.on
: do overlay ’info’ on the main view

Info.none
: no info

Info.timing
: only basic timing will be shown.

Info.stats
: timing and internal stats are shown, used for program testing.

Info.game
: timing and basic info for a game are shown.

  The second line will show (in order):

  — time from start of game (in seconds).
  — number of targets present (both standard targets and ground targets)
  — number of weapons used in total
  — number of hits scored

  The third line will show:

  — score (counting down!)
  — plane speed (meters/sec).

The basic ’timing’ data, which is always the first line, is a list of millisecond durations for:

— total time of frame
— graphics drawing (display list -> screen)
— 3D transformations (world -> display list)
— objects simulation (old world -> new world)
— other visual calculation (hud, text, sky etc.)
— vertical sync wait (if double buffering)
— total minus the rest; will include the auxiliary windows time and network disturbance.

5.6 Emit Menu

Various objects can be created with this menu. These objects are used as targets.

Emit.target
: create one random target

del delete all targets

Emit.gtarget
: create one random ground target

del delete all ground targets

Emit.box
: create one random box. Boxes are cubes that hop around which can be shot down.

del delete all boxes

Emit.del tgts
delete all targets, ground targets and boxes.

Emit.drone
create one random plane (drone)

del delete all drones. This will also set the number of automatic drones to zero.

Emit.drones
specify how many drones should be automatically maintained in the air. Whenever one is lost another one takes off.

Emit.killers
specify how many of the drones should be killers. These will be set to Chase and Kill mode.

5.7 HUD Menu

Various aspects of the HUD can be set. Each option is either set, reset or toggled. The default is to toggle the option but the first three menu items can be used to change this mode. This menu can be accessed from the main menu as well as directly with the 'u' command. The following selections appear on many of the sub-menus:

0 turn off	turn option off
1 turn on	turn option on
2 toggle	toggle option on/off

This top level menu will bring up a number of sub-menus which are described further down this doco.

HUD.off
turn the HUD off. When turned off, the radar symbols will still show. This is a feature of the game which is not like the real thing; it allows you to play with a very clean view. Not only will the radar stuff still show, but the symbols will now move freely across the full screen rather than being confined to the HUD area. To get rid of the reticle/TDB use the 'parts' menu.

HUD.on
turn the HUD on

HUD.type
Select HUD style. Although the styles are named after planes, each plane actualy displays many styles depending on the mode of operation.

HUD.parts
The HUD has many components. This sub-menu allows you to choose which are included in the HUD display. Selecting a HUD type will automatically adjust these to what is appropriate for that style.

HUD.options1
This (and the next) selection allows you to set some parameters which modify the appearance of the HUD. The most often used ones are in this sub-menu and the rest are in the next.

HUD.options2
See description of "options 1" above.

HUD.radar
This will configure the radar symbology on the HUD.

HUD.ils
Will let you select the ILS beacon. In the future there should be a more elaborate NAV facility instead.

HUD.hdd
: This menu is now changing as a new hdd (head down display) system is being implemented.

HUD.help
: Display the full hud setup options list.

5.7.1 HUD.type Menu

Fly8 supports a number of HUD styles. The name of the hud does not necessarily correspond with the plane type but this is what I found on the various videos that I saw. If anybody has more knowledge or can provide other detail PLEASE feel free to advice me.

HUD.type.Classic
: This one I made up before seeing any real HUD. The basic data is laid close to the edge and leaved most of the area free from obstruction. My original aiming reticule was 8 dots in a circle but I discontinued it in favour or the more common piper style. The numerals on the pitch ladder do not rotate and the fast font is used. On a slow machine this hud (especially in low detail) will perform much faster that any other.

HUD.type.FA18
: This HUD does not use tapes for the altitude and speed. The pitch ladder is narrower than usual and slanted toward the horizon. A good feature is the fact that the pitch ladder stays always in view: if the velocity vector goes off the screen (easily done on the FA-18 which has no trouble flying at high AOAs) the pitch ladder adopts (temporarily) the waterline mark. Another feature is the closure speed which is shown under the piper rather than on a radar ranging scale. Optionally, a pendulum (or what do you call it?) can be displayed which shows you your roll angle with good resolution up to 45 Degrees either way. This is a wide angle hud - 20 degrees side to side.

HUD.type.F16
: This HUD uses simple scales (no baseline). The heading scale can be at the top or at the bottom.

HUD.type.F15
: This HUD is probably used for air to air on other planes. The speed scale is upside-down. The heading scale can be set to two different positioned at the top.

HUD.type.Ether
: A new HUD type now being developed.

5.7.2 HUD.parts Menu

The various HUD symbols can be individually selected for display.

HUD.parts.ladder
: Select pitch ladder (and related) options.

HUD.parts.altitude
: Show altitude scale (or box).

HUD.parts.speed
: Show speed scale (or box).

HUD.parts.heading
: Show heading scale.

HUD.parts.border
: Show the HUD border (in gray color).

HUD.parts.vv
: Show the plane’s velocity vector. A must for accurate flying.

HUD.parts.vw
: This activates a mode that the FA18 uses: when the vv goes off screen, a waterline mark will appear and the pitch ladder will be drawn around it. This way the ladder never goes completely off screen (which can often happen when flying at high AOAs).

HUD.parts.plus
: Show a 'plus' sign at the center of the screen.

HUD.parts.pointer
: Show a small (red) mark that tracks the joystick (or mouse or whatever pointing device you use).

HUD.parts.beta
: Show the sideslip angle (beta) on FA18 style.

HUD.parts.ground
: Brings up a menu of ground proximity related options.

HUD.parts.director
: Show the flight director. This is part of the new ether HUD still under development.

HUD.parts.waypoint
: Will show the target as a diamond along with a small pointer near the FPM pointing at it. This small pointer represents a top view where the small circle is the pointer base and the line from it marks the direction of the target. When ILS is active it is that point that will be tracked by these symbols.

HUD.parts.tracers
: This shows a 'string' hanginf off your boresight, which represents the bullets position if you were firing. Horizontal marks indicate interval of 1500 feet, and a small bead marks the target distance. This tool is useful for gun shots when the radar is shut down.

HUD.parts.ghost
: The Flight Path Marker is caged. A ghost FPM is shown if the caging shifts the FPM by more than 1 degree. It looks like the FPM but without the central circle.

HUD.parts.truehead
: The Heading scale is normally calibrated to show about 30 degrees. However, you may prefer it to span directly correspond to the world view through the HUD. In this way, an object that is 10 degrees left of the HUD center will really be indicated with a 10 degrees offset on the heading scale. This option is the default for the Ether HUD.

5.7.3 HUD.parts.ladder menu

These options modify some aspects of the HUD appearance regarding the pitch ladder and associated symbols.

HUD.parts.ladder.ladder
: Enables the display of the pitch ladder.

HUD.parts.ladder.pinned
: In this mode the ladder it always attached to the HUD center (waterline mark).

HUD.parts.ladder.right
: The numbers on the ladder steps are displayed only on the right wing.

HUD.parts.ladder.erect
: The numbers are displayed erected. The default will rotate the numbers with the ladder.

HUD.parts.ladder.color
: In this mode the positive steps are blue and the negative are red.

HUD.parts.ladder.funnel
: The step tips are displayed in the middle gap instead, which gives it a funnel shape.

HUD.parts.ladder.slant
: The steps are slanted rather than flat. The slant increases with the pitch angle, reaching to about 45 degrees.

HUD.parts.ladder.zenith
: Will display a zenith/nadir marker. It is a small circle for the zenith and a similar circle with a cross inside for the nadir.

HUD.parts.ladder.under
: The numerals are displayed under the step rather than beside it.

HUD.parts.ladder.tip0
: Specifies that you want a tip to be displayed on the zero pitch (horizon) ladder step.

HUD.parts.ladder.hold
: This is a temporary option that controls the behaviour of the program when the pitch is very high such that some standard calculations cannot be carried out. By default it freezes the heading and allows the roll angle to vary.

HUD.parts.ladder.h
: This controls another aspect of the behaviour as described in the previous option. It causes the roll angle to freeze while the heading will continue to reflect your attitude.

HUD.parts.ladder.sun
: Shows a sun symbol, a small white circle which stays on the HUD edge when the sun is out of sight. It actually follows the zenith rather that the real sun and is intended as an aid in recovering your situational awareness.

HUD.parts.ladder.negtip
: This indicates that the step tips will always point toward the nadir. The default is to point toward the horizon.

HUD.parts.ladder.sizes
: Brings up a menu for setting the sizes of the ladder features.

5.7.4 HUD.parts.ladder.sizes menu

These options specify the sizes of the pitch ladder parts. The size is a relative number with a value from 0 (size zero) to 16384 (full HUD width). A resize menu will come up which will allow you to modify the size using the '+' or '-' options (increase/decrease) or using the '=' option (you can then enter a new value). Use the '*' option to restore the value to what it was at the start (but when you exit this menu the new value is final).

HUD.parts.ladder.sizes.gap
: The width of the gap in the middle of each step.

HUD.parts.ladder.sizes.step
: The width of each ladder step.

HUD.parts.ladder.sizes.horizon
: The size of the horizon step in flight.

HUD.parts.ladder.sizes.land
: The size of the horizon step when the landing gear is lowered.

HUD.parts.ladder.sizes.tip
: The size of the step tip.

HUD.parts.ladder.sizes.ndash
: The number of dashes that make a single step (one side of it).

5.7.5 HUD.parts.ground menu

These options modify some aspects of the HUD appearance regarding ground proximity.

HUD.parts.ground.gnd ptr
: The ground pointer is a marker that shows your bank (roll) angle. It is represented by an arrow-head which slides along a set of angle marks. The highest angle marked is 45 degrees.

HUD.parts.ground.Xbreak
: Show the X-break symbol if at risk of hitting the ground. This symbol is a large, blinking, X symbol in the center of the HUD. If you get even closer to impact then a PULL UP message will be flashed and a high pitch warning will sound.

HUD.parts.ground.Xvar
: In this mode the X-break symbol starts with two angle-brackets (like ´> <' that get closer towards the HUD center as the impact gets nearer. It then merges into a single X shaped symbol.

  The default mode always shows the X shaped symbol, regardless of the time to impact.

HUD.parts.ground.Xgrid
: Show the warning grid if at risk of hitting the ground. This is a red grid which will overlay the ground if you fly too low. This experimental mode is attempting to assist you in regaining awareness of your situation when there are not enough groung features in view.

HUD.parts.ground.pullup
: A pullup cue will be displayed. It indicated the dive angle at which you will hit the ground in 5 seconds. When a bomb weapon is selected a safety distance of 200 meters is set to protect you from the explosion.

  The pullup cue looks like an extra pitch ladder step (usually narrower), with upwards slanted tips.

5.7.6 HUD.options1 Menu

These options modify some aspects of the HUD appearance.

HUD.options1.heading
: The heading scale shows the planes heading. The numbers displayed are in the range 000-350 in increments of 10. This option selects between showing the full 3 digits or using an abbreviated form. The abbreviated form will only show the top two digits (09 for 90 and 27 for 270). The Classic HUD style will show the full number but without the leading zeros.

HUD.options1.knots
: Internally all data is stored in meters. This option requests that all numbers use knots/feet (as appropriate) instead. It is the default for the standard HUDs.

HUD.options1.top
: This will further modify the heading scale. The scale will show at the top or at the bottom depending on this option. For the FA18 HUD style, this option will cause a base-line to be drawn under the scale (the scale will stay at the top regardless).

HUD.options1.fine
: For some of the scales this option will show more detail. The standard detail is to show a tick every 5 units. The fine detail will show a tick every two units.

HUD.options1.xfine
: This is a further refinement of the above 'fine' level and will show a tick for each scale unit.

HUD.options1.big
: [obsolete]The name is completely wrong. This option defines the style of the ticks on the scales (for some of the HUDs only). The usual way is to have the ticks go from the base-line to the outside. In the 'big' style the scale will be along the edge with the ticks towards the inside.

HUD.options1.scale
: This defines the number of units along the scale. This affects only the Classic HUD. The more units, the longer the scales.

HUD.options1.area
: The HUD has a fixed area (measured in field-of-view degrees). You can alter this size. Note that although the HUD size changes when you zoom in/out, it still keeps the same FOV. This option defines how many degrees are from the center to the edge of the HUD (all HUDs are square).

HUD.options1.cas
: The speed show will be the 'calibrated airspeed' rather than the 'true airspeed'. A snall 'T' or 'C' will mark the type of speed shown.

5.7.7 HUD.options2 Menu

These options modify some aspects of the HUD appearance. These are the less used options.

HUD.options2.a alarm
: Enable aural alarms. If you hate the GLIMIT beeps etc. then use this option to turn these alarms off.

HUD.options2.v alarm
: This will Enable/disable the visual alarms that show on the HUD.

HUD.options2.font
: Select the font for the stroke characters used on the screen

HUD.options2.fontsize
: Select the stroke font size. A size of 8 means 'use the default' and all other sizes are relative to 8. The default is calculated from the screen resolution.

5.7.8 HUD.radar Menu

The radar symbology on the HUD is controlled with these options.

HUD.radar.corner
: Radar target data can be at the bottom-left corner of the HUD or can follow the target designator.

HUD.radar.data
: Request to show target data.

HUD.radar.distance
: Request to show target distance in intel mode (mainly used in the radar/map modes).

HUD.radar.name
: Request to show target type in intel mode (mainly used in the radar/map modes).

HUD.radar.accvect
: Show target acceleration vector as a hand inside the reticle piper.

HUD.radar.reticle
: Show aiming reticle piper.

HUD.radar.target
: Show the target designator.

HUD.radar.ross
: Use Ross's method for the aiming reticle. This mode will show a small box in front of the target where it is expected to be when a bullet hits it. If you aim the reticle at this box and shoot then you should hit the target.

HUD.radar.limit
: Unlike real HUDs, the radar symbols can be displayed all over the screen rather that just inside the HUD area. 'limit' will specify which way it should be.

HUD.radar.thick
: This is an experimental option to draw the radar reticle thicker.

HUD.radar.hidetgt
: When active, if the TD box is under the reticle then it is not shown. This de-clutters the area of interest.

HUD.radar.tpointer
: If the target is off the HUD then a line is drawn from the boresightpoint (the '+' at the view center) towards the target. The distance to the target is shown digitally: this is the angle of the target relative to the boresight direction: 180 means it is exactly behind you, 90 means it is on your side (ANY side, right, left, above or below). It should be read as: if you turn so many degrees in the direction of the pointer then you will have the target straight in front.

HUD.radar.vpointer
: This will modify the 'tpointer' such that the length of the pointer will vary with the target relative angle.

5.7.9 HUD.hdd Menu

Fly8 supports a number of HDD devices. This menu allows you to set these up with various options. It is still under construction.

HUD.HDD.instruments
: Show the instruments panel. This is a very basic (and rather useless by now) instruments depiction that will be overlayed at the bottom right of the main window.

HUD.HDD.nav
: Request to display navogation info in the panel display.

HUD.HDD.compass
: A compass will be added to the radar map display. See next options too.

HUD.HDD.square
: Selects a square or round compass

HUD.HDD.ortho
: Selects angled or orthogonal ticks around a square compass.

HUD.HDD.panel
: Request to show the panel HDD on the main window. This will show at the bottom right side as digital flight data.

5.8 Net Menu

For full details please refer to the networking chapter.

Net.ping
: find out who else is playing. A message is broadcasted and for each responding player a message is displayed.

Net.play
: join another player's game (or all players)

Net.quit
: stop playing with a player. If there are more than one players then you will be asked to choose.

Net.message
: send a message to a player (or all). You will later be notified how long it took the message to reach each player and return a notification.

Net.accept
: accept a player's request to play with you. Used in response to the Requesting message.

Net.decline
: decline a player's request to play with you. Used in response to the Requesting message.

Net.always accept
: automatically accept any requests to play.

Net.always decline
: automatically decline any requests to play.

Net.manual reply
: do not automatically respond to any requests to play.

5.9 Options Menu

Options.Version
: show program version and compile date/time.

Options.Smoke
: set/clear smoke generation. Damaged planes and craters will smoke if the option is enabled.

Options.Font
: show current stroke font. It is displayed if large on the center of the screen.

Options.Colors
: Show the current palette setup.

Options.Modes
: show current program modes setting (same as 'm' command)

Options.Sky
: paint blue sky in views.

Options.Gravity
: enable gravity (default). Will affect bullets path.

Options.Play Blues
: [sound debug] No simulator is complete without it. Actually used to test the sound generation logic which for now is operational only on the PC.

Options.Verbose
: Toggle verbose mode. Off by default. When using the menu system you will not be shown the standard 'help' screens. Use the 'm' and 'uh' commands to see the 'modes' and 'hud' help screens. The command line 'v' controls this option as well.

Options.Net Stats
: display network statistics (same as 'n' command).

Options.Limited
: Will limit the ammunition to the takeoff quantity. Normally you have limitless ammo.

Options.No stall
: This option reduced the effectiveness of a stall. Good for early practice.

Options.Paused Msg
: This option is ON by default. If set to OFF then the 'paused' message (when issuing the 'p' command) will not display. This is nice for clean screen captures.

Options.win ident
: This option is OFF by default. If set to ON then a window name is displayed at the top right corner of each.

5.10 Auto Menu

These options enable some augmentation systems which modify the behaviour of some controls.

Auto.Flaps
: Enables the flaps Control Augmentation System (CAS). This will give you better turn performance by adding flaps (or leading edge flaps) when needed).

Auto.Elevators
: This will enable the CAS which limits the Elevators sensitivity at high speed to avoid excessive load. A Stability Augmentation System (SAS) is being added which will improve dynamic pitch stability by taking some authority over the elevators.

Auto.Rudder
: Enable the rudder SAS and CAS. It is not yet implemented.

5.11 Debug Menu

Debug.debug
: Enable the general debug mode. Some programs will display internal data when in this mode. This changes a lot and cannot be documented.

Debug.Trace
: Enable debugging trace. Useful for developers only.

Debug.gp w
: General purpose debug option W is controlled by this option. It is used for debugging and cannot be documented here, see if there is anything in the readme.

Debug.gp x
: As above for option X.

Debug.gp y
: As above for option Y.

Debug.gp z
: As above for option Z.

5.12 Buttons Menu

The Buttons on your pointer (mouse, joystick etc.) are recognized by the pointer driver and reported back to Fly8. It is now up to the program to react to the button status. If a button press command is associated with a key then it will be issued when the button press status is recognized. If the button was set to release mode then when you release the button a command may be sent too.

You use this menu to define the modes that a button responds to. The actual command associated with a button press/release is defined by including, in fly.max, a line like:

```
Def Btn 0 F1
```

This will cause button 0 (the first button) to issue the F1 (weapon fire) command. Another example:

```
Def Btn 1 +
Def Brl 1 +
```

This will cause a '+' (speed brake on/off) to be issued when you press button 1, then another '+' when it is released. This way the speed brake is active while you hold button 1.

If you want the same key to have a different command depending of your game mode then you can use the virtual 'button shift modes' provided. The buttons are considered to be sensitive to the status of the Alt, Ctrl, Shift and Special buttons (which do not exist on any input). If the Btn-Alt is set then pressing button 1 will be recognized as Btn-Alt-1 and will respond to a line in fly.max like:

```
Def Btn Alt 1 r
```

to turn the radar on/off. The only way to set these button modes is through this buttons menu. However, you can program any hot key to do this for you. You can program Alt-S as your button Shift key in fly.max:

```
Def Alt S M s           # Now Alt-S toggles Btn-Shift mode
```

This allows the buttons to have 16 diffrerent modes which should be enough, however, by default, the buttons do NOT respond to the mode state. To make a button sensitive you use this menu.

All joysticks have 2 buttons, Thrustmaster FCS and CHPro have 4 (use the ":four" option in this case) and also the hat is decoded as four more buttons. Mice may have more buttons and some drivers will recognize the common three. When using a Thrustmaster WCS of FLCS you may program all the buttons to be handled through the keyboard, in which case there are NO buttons directly visible (use the ":zero" pointer option in this case).

The first thing you will see is a request to name the button to be customized. The buttons are named with a single alphanumeric character (so a maximum of 36 buttons can be handled by Fly8). Then you will be presented with the following menu.

Buttons.Alt
: Will set the Button Alt mode sensitivity. By default all buttons ignore this mode, the pointer option a=... can nominate buttons that should be sesitive to this mode.

Buttons.Ctrl
: Will set the Button Ctrl mode sensitivity. By default all buttons ignore this mode, the pointer option c=... can nominate buttons that should be sesitive to this mode.

Buttons.Shift
: Will set the Button Shift mode sensitivity. By default all buttons ignore this mode, the pointer option s=... can nominate buttons that should be sesitive to this mode.

Buttons.sPecial
: Will set the Button Special mode sensitivity. By default all buttons ignore this mode, the pointer option p=... can nominate buttons that should be sesitive to this mode.

Buttons.Debounce
: Set the debounce mode. When ON, a button press will activate a command once, until released and pressed again. When OFF, the command will be repeatedly issued as long as the button is pressed (once per frame).

  By default all buttons are debounced. The pointer option d=... can nominate buttons that should NOT be debounced.

Buttons.Release
: When ON, this option will enable issuing button release commands. When OFF the button release will not issue a command.

  By default all buttons recognize release events. The pointer option r=... can nominate buttons that should NOT recognize release. [but I am not sure what it is useful for].

Buttons.Clear
: Will clear all modes sensitivity for the button. Note that this will include the Debounce and Release modes which are normally ON. You use the Clear option when you wish to directly set a button to a known state, like: "x d r s" will set this button for Debounce, Release and Shift sensitivity.

Buttons.Cancel
: Will exit without changing the button definition.

5.13 Commands Menu

Some simple commands toggle between options. This makes it impossible to use a macro for setting these commands to a pre-determined state. The commands menu allows you to do just so. Select ’On’, Off’ or ’Toggle’ then hit another command. Now it will follow the earlier setting rather than the default ’Toggle’ behaviour.

6. Command Line Editing

When a command needs to receive a parameter which is more than one keystroke it uses a line input facility. It allows you to use history and editing. You can use the arrow keys and insert/delete keys to move about and edit your response. The up/down keys will retrieve history. Finally you will need to press Enter for the program to accept the input. If you key the start of a line and hit PgUp then a search will be done for a previous entry with the same beginning.

The history queue has 20 entries, all input requests share this same queue.

7. Aural Indicators

Sound is used to inform and warn. The sounds at the moment are simple tones or tone sequences.

A short beep will sound when:

- you fire a weapon
- you hit a target, or a plane crashes
- the radar locks onto a (new) target
- a menu selection is invalid

A low beep will sound when:

- landing
- taking off
- landing gear status is changed

Two repeating tones for:

- Emergency alarm (pull-up, eject etc.)
- Warning alarm (stall, g-limit etc.)

Repeating scales when:

- target practice has ended

8. Visual Indicators

These are highlighted words flashed onto the HUD. They vary in size and blink rate.

WAIT	You ejected and landed but your plane did not yet crash. This one does not relate to the HUD, all the others only show when the HUD is on.
STALL	You are flying the plane too slow to maintain lift or you are turning too sharply at a too high angle of attack.
GLIMIT	You exceeded the maximum acceptable G force of your body (+9G to -3G), or you exceeded the 10G plane structure limit.
FUEL	You have less than 10% fuel left. The less fuel you have the faster the message blinks, then it finally stays on.
PULL UP	You are about to hit the ground unless you pull the plane up immediately. If the danger is higher then a red ground grid is flashed to give the pilot better orientation (there is not enough scenery to build proper visual awareness).
EJECT	The plane is damaged beyond control. Shift-E to eject.

9. The Plane

This chapter explains in detail how planes are handled in this program.

The plane is controlled by your pointing device, preferably a joystick. The basic controls will have the following effect:

Left/right controls will cause the plane to roll. The roll will continue while the controls are engaged. When the joystick is centered the rolling will stop and the plane will stay in the current situation. If you want to fly level after rolling to the right then you will have to do the following:

— roll right (the horizon will roll left).

— stop rolling (the horizon will stay at a fixed angle).

— roll left (horizon rolling back to the right).

— stop rolling (when the horizon is level).

As the plane has momentum, the response is not immediate and you will have to get a feel for it.

To start climbing you will pull the stick toward you until the climb angle is what you want and then release the stick. The plane will continue climbing until you push the nose down for level flight. If you are rolled over to one side then the pulling will cause the plane to turn into that side. If you are upside-down and you pull the stick then you will start descending towards the ground. In other words: the up/down controls (elevators) are used for any change of direction, both left/right and up/down.

To turn right, first roll right, then pull the stick until the desired heading is reached, then release the elevators and roll back to level flight. Of course, due to gravity and plane dynamics any change in situation will probably cause the plane to move in a direction slighly different from what the controls suggest - you should learn to compensate for this. The flight-path-marker (the little circle with three wings) tells you where the plane is heading and this is hardly ever the direction where your plane is pointed at.

To control your engine you set the throttle with the 9/3 keys. The throttle can be set to between -100% and 100%. Reverse setting only works on the ground. Each keystroke is 5% change. The 1 key will set the throttle to 100% and the 0 key to 0%. The planes speed will pick up slowly (depends on the planes weight and the engines power). You can engage the after-burner with the '.' key. To slow down you may use the speed brakes ('+' key).

About the AfterBurner: light it with '.' (will also set throttle to 100%). Then each '.' or '3' (power-up) will add a notch. Each '9' (power-down) will take it down a notch. There are five steps (say 20% each). The throttle display will show '103' for '100% + AB3', 105 is full AB. The engine display will show thrust in % of mil thrust (full AB is about 150%-160%). If you use '1' (max throttle) or '0' (idle engine) the ab is turned off. NOTE that AB5 uses about 6 times as much fuel as MIL for 60% extra power!

Note that with the PC keyboard and the NumLock engaged, the above keys appear in a logical order.

The Classic plane is an over-simplified vehicle. It has no momentum and no aerodynamics characteristics, it goes where you point it and is a good way to get the hang of the controls. It will never crash either (you can fly underground of course). But don't get too used to it, real planes handle very differently (the Classic is more like a weightless spaceship of an arcade-game).

10. The Head Up Display

The program displays a number of HUD styles. These are named according to a plane type but this is just because I first saw this HUD on a video tape dedicated to that plane. In reality each plane has a number of HUD modes. You can change the HUD style through the hud menu regardless of the plane type.

The HUD is a piece of glass that is positioned at an angle in front of the pilot. The pilot looks through it to see the front view from the cockpit and at the same time a reflection of a video screen is seen (this CRT is in the 'dashboard' facing up). The dual-image is similar to when one looks outside through a window at night and sees a reflection of some part of the room as well as the outside. The HUD can be displayed by itself on one of the auxiliary HDDs (sometimes refered to as the 'HUD repeater'), which is useful when the HUD optics is not operational (damaged).

In practice, the HUD is a flat image superimposed on the front view, and it uses a special (usually green but you can change this) color. It does not cover the full field-of-view.

The image projected onto the HUD contains two kinds of information. One type is data that the pilot will otherwise have to look for in some cockpit instruments (thus taking his eye off the outside scene); this is simply a way of putting the most important information if front of the pilot. An example is the display of plane speed. The other kind is information directly related to the outside image and meaningfull only in relation to it, for example: a bounding box is displayed such that it coincides with a visible target that the radar is locked on.

The prominent features on the HUD are a number of scales which are usually diaplayed along the edge. Sometimes the detail of the scales can be controlled in three levels through the Hud menu (see there).

The HUD symbols will relate to flight data or to auxiliary systems (weapons, radar, fuel etc.). The data related to the flying of the plane is described first.

10.1 Heading

Your compass. It will be shown as a horizontal ruler that moves as you turn. The current heading is marked with a 'tick' or a 'V'. It can be at the top or at the bottom of the HUD (Top option in Hud menu). North is 360, South is 180, East is 90 and West 270. Some modes do not show the trailing zero (270 is shown as 27) and NO, there is no support for radians or other units...

10.2 Altitude

Your height above sea level, a vertical ruler at the right edge of the HUD. It may be accompanied by a second bar (immediately to its left) that shows your climb/fall rate. This ruler moves up and down as the plane moves, the current altitude is to be read at the 'tick' in its middle. High altitudes show in thousands (with a possible decimal point) while low ones will show exact. The FA-18 style HUD shows the altitude in a box at the right side of the HUD with the climb rate above it. Some HUDs will show a radar-range scale adjacent to (and to the left of) the altitude scale. This will indicate the distance to the target (the full scale range

is shown just above this scale) as a sliding tick while the target closure speed is shown inside the tick.

Climb rate is in meters (or feet) per minute!

10.3 Speed

Your speed is shown as a ruler at the left edge of the HUD, a tick marks the current value. The FA-18 style HUD shows it in the left box. Some HUDs do not show the trailing zero while others show have the scale run from top to bottom.

This information may be in meters/kmh of feet/knots (use the Hud Knots command to toggle). The F16/FA18 default to feet/knots.

10.4 Pitch ladder

The orientation of the plane is displayed as a ladder, each step relates to a different pitch. The steps are always parallel to the horizon. Each step is marked with a number which is your pitch angle (90 degrees is straight up, -90 is down and zero is level). The step's angle represent the planes roll. When you are upside-down the steps are too, as you roll the steps turn in the opposite way to follow the horizon. The negative-pitch steps (when you are going down) are dashed while the positive ones are solid. Small winglets at the tips of the steps point toward the ground. The zero-pitch step is larger and is your artifical horizon if you cannot see the real one. The FA-18 style HUD tries a bit harder by bending the steps toward the ground: the higher your pitch the larger the slant. It also shows a small circle at the straight up/down directions (the down one has a cross through it).

Although the pitch ladder follows the horizon (meaning the zero-step is on the horizon) there is some freedom in where on the horizon to show it. Unless you disable the velocity vector (flight path) marker (see below), the ladder will be centered on it. This means that at a high angle of attack the pitch ladder may be out of view (as will the vv).

In the case of the FA18, if the vv goes out of view then a waterline marker will appear at the center of the HUD (it is a W marker in a fixed position) and the ladder will shift (smoothly) toward it. Once the vv is back in view the ladder will return to it and the waterline mark will disappear (the transitions take about two seconds maximum). The FA18 ladder shows an extra-long zero-step while the landing gear is down.

10.5 Velocity Vector

A plane rarely moves straight ahead due to gravity and aerodynamic forces. This marker (sometimes called the 'plane symbol') is a tiny circle with wings on either side and at the top (it is a stylized shape of a plane from behind). At any time, this marker shows you where the plane is heading. You will most of the time use this marker as a reference for flying the plane. The center of the view is rather useless for flying (you can bring up a cross-hair with the 'u+' command) but can be helpful in aiming the cannon (in the absence of the aiming reticle).

The Classic plane always goes ahead, so the vv will be fixed at the center of the HUD. By default it will not be shown for this plane.

10.6 Waterline mark

This is a 'W' that shows in the straight ahead point on the front view (this is not always the physical center of the HUD). It comes on whenever the landing gear is lowered. The FA-18 HUD shows it whenever the Velocity Vector is outside the HUD.

10.7 Radar Symbology

When the radar is active, some symbols related to its operation are displayed. The main features are the target designator box and the aiming reticle (the Piper).

10.7.1 Target designator

This is a square that is centered on the target. If the target is not locked then the box will have only corners. The target should be visible inside the box, unless it is off screen. When off screen, the box has a ´+' through it and it crawls along the HUD edge showing you the direction where the target is. If the target is actualy behind you then the '+' is replaced with an 'X'.

10.7.2 Aiming Reticle

If a target is close enough (within weapon range) then an aiming reticle appears. The reticle is a circle with 12 ticks. Each tick represent a distance of 1000 to the target and the range is marked with a tick that moves along the inside edge of the reticle. A tick at 11 o'clock means a distance of 11,000 etc. You should fly the plane so that the center of the reticle (has a dot) is on the target and then shoot (actually, the cannon/radar computer will display 'shoot' above the reticle when you have a good aim). If this sounds simple it is because it is a simple procedure; the problem is that in order to get the target in the reticle you will NOT be flying the plane directly toward it. In practice you forget about where the plane should go and play a game of follow-the-target with the reticle (just try and not hit the ground).

The F16 will also show a 'hand' inside the circle which indicates the direction and magnitude of the target acceleration (this one is very jiterry at times). You can turn this hand on/off with the Hud menu "acc vect" command.

The FA18 HUD shows the closure speed outside the lower right side of the reticle. The F15/F16 shows the same information on the radar range scale (beside the altitude) marked with a large '>' symbol. The closure speed measures how fast you are catching up (positive) with the target.

However, in order to complicate the situation we have some variations possible:

There is an alternate piper: Ross's reticle. This is a different aiming method altogether. A square reticle is shown with only the corners visible. It is ahead of your target at all times on its projected position. You have to aim the piper at the box and then shoot. With this one you do not care where the real target is because the aiming box replaces it. The piper will be fixed at the center of the HUD. You may want to turn off the target designator with 'ut'.

The target designator and aiming reticle are part of the HUD display, however you may choose to ignore this and request that these use the full screen. Use the 'uL' command to limit these to inside the hud area or use the whole screen.

10.7.3 Radar Range

The radar measures the distance and relative (closure) speed of the target. The range is shown as an extra scale on the right side on the HUD while the closure speed is shown beside the '>' mark on it. The FA18 HUD does not show this scale but shows the closure speed under the piper with a 'Vc' mark.

10.7.4 Digital data

When a target is selected, some digital information may be displayed (it can be disabled by the hud/radar menu). This data shows at the left bottom corner of the HUD and has the following items:

- distance to target (units or k's with one decimal)
- closure speed (meters/knots)
- time to meet (seconds with one decimal).
- target type or pilot name

When the target is in range the time shown is bullet time-to-impact rather than plane flight time.

If the Corner option is not selected (uC) then this data will show under the target designator box. If the target is too close to the bottom then the data may show above it.

If you activate the Intel mode (i) then all visible targets get a box with the following data (the MAP and RADAR diplays always have this mode):

— distance to target (units or k's with one decimal)

— target type or pilot name

You can use 'un' to disable the display of the second line.

10.8 Other Features

In addition to the above features, the HUD may show the following:

The FA18 type HUD shows as standard, on the left low edge, the angle of attack (aoa), the mach number and the pilot's vertical Gs. The selected weapon (and available units) is displayed at the low center of the HUD.

The F15/F16 HUD shows the aoa at the top right above the altitude scale. The weapon selected is shown at the top of the data list ('XXX' means none selected)

11. The ILS

The Instrument Landing System (ILS) is a system that provides enough information about your approach to guide you to the touchdown point with great accuracy. The system comprises two separate facilities: the Localizer beam which tells you how well you are aligned with the runway and the Glide Path beam which monitors your descent rate. The two components measure your approach error and display it as two bars.

The Localizer deviation bar is a vertical line that moves accross the HUD and indicates which way of the correct line you are. If the bar is left of center then this means that you are to the right of the Localizer beam, so you should correct your approach to the left. When the bar is right of center you will need to move to the right too. You are correctly aligned when the bar is at the center. You can judge the bar's position by noting the number of ticks along the horizontal bar. The larger middle one is where you want to be. The bar is at full deflection when your error is 2.5 degrees.

The Glide Path deviation bar is a horizontal line that indicates where the correct descent line is. If the line is above center then the you should be flying higher (your descent is too rapid or you are descending toward a point on the ground too short of the runway); you should gain some height or reduce your descent angle. In the same way, when the line is below HUD center you are above the correct path. The bar is at full deflection when your error is 0.75 degrees.

Note that the ILS system does not know where you are heading, it just tells you how close you are to the correct approach path. The system does not even know if you are coming or going! So make sure that you approach the runway from the correct end or the ILS Localizer deviation bar will show reverse reading and the Glide Path will direct you to land at the far end of the runway.

Real ILS systems have very narrow beams and will only operate when you are reasonably aligned. These systems will tell you when you are out of range. The one in Fly8 is active within a radius of about 25 kilometers around the runway.

When the ILS is operating a marker along the heading scale will direct you to the airport; use it for the general approach but then identify the correct runway carefully.

To select the desired runway use the ILS menu and choose from the list. There are now two airports (A is home and B is for the drones) and each has two runways: 18 (approach at heading 180) and 27 (approach at 270). You can turn the ILS off and it will still remember the last active runway which will be offered when you use the command again. These ILS aids are defined in fly.nav and you can change these. The 'H' command option is used to designate your home runway, otherwise it is the first one defined (the drones use the last one in the list).

12. Radar and targets

The radar in this program does NOT try to simulate a real radar. The real thing has many types and modes of operation. This one just cheats to get its data.

When enabled ('r') the radar measures distance, direction and speed of possible targets.

In the basic mode, the radar constantly selects the closest target. This may cause it to 'jump' between targets as they change distance. You can put the radar in 'locked' mode ('l') which will make it stick to the selected target. In this mode, when a target is destroyed, the nearest target will be selected and stay locked. Use 'l' to release the lock (or you can turn the radar off/on with 'rr').

There are 3 other acquisition modes controlled by the 'f' command.

0 pick closest target (old way, as described above).

1 3.3deg circle: boresight. Only targets inside the small circle are detected.

2 20deg circle: HUD. Any target inside the large circle (which covers most of the HUD area) is detected.

3 5.3deg wide by 60deg high: vertical. Targets inside the narrow band (+-5.3 degrees wide but 60 degrees tall) are detected.

In modes 1-3, a target is highlighted when it is within the designated area. The limits of the modes 1-3 are drawn on the HUD.

If you are in locked mode then the first detected target will immediately be locked, otherwise you will have to hit 'l' to lock the highlighted target. Only when a target is locked you get the aiming reticle (if it is close enough).

It should be made clear that the 'locked' mode is set/reset by you with the 'l' command. Once engaged, there is no need to lock on targets because the first one to qualify will immediately be locked on. If you want then to select another target then use 'l' to release the target and then later 'l' to lock on the new one.

Once a target is locked, the selection markings disappear and the piper shows (or if it is still far then only the target designator box shows).

A target is identified with a box around it (the target designator). When the target is out of view the box has a large '+' crossing it which changes to a large 'X' when the target is actually behind you.

If you issue Shift-C then the plane will chase the current target (there must be one or the plane will just patrol round the runway). Use the ´k' command to allow automatic firing ('k' works even when not in Chase mode, and is useful if you want to practice the chase and let the auto-pilot do the shooting).

When the reticle goes off-screen it gets a '+' inside it.

You can shoot at various objects. Use the 'emit' menu to create these objects.

When you hit something it gets damaged and fragments fall off. When enough damage is done the object is officially HIT. It blinks red/white and starts falling toward the ground. Practice targets are destroyed immediately. These fragments are lethal and can hit any other object! Normally you can fly through any object EXCEPT a bullet - so don't stay behind a broken plane or you may be hit by the falling fragments.

13. Networking

As others said before, playing with oneself is fun but you don't make as many friends (they said you'll go blind too).

The program will let you play with others using a variety of communication media. Once networked, objects are shared between the players. The number of players is only limited by the capability of the

network medium.

Below here, numeric parameter values can be given in C format, i.e. if it starts with 0x it is hex, starts with 0 is octal, else it is decimal.

13.1 msdos

13.1.1 serial

At the bottom of the pile is the PC to PC serial connection. Only two players can combine in this way. You can choose two drivers for this:

```
dncom.N:baud:parity:bits:stop:xmode:inbuf:outbuf
```

Direct control over the com port. It can handle any speed, but slow machines will drop charaters if you go too fast. Slow machines should NOT use output buffering.

Positional parameters:

N	com port number: 1...4
baud	the line speed, up to 115200.
parity	e, o or n
bits	8
stop	1
xmode	xon or xoff; do not specify!
inbuf	4000 is enough. experiment.
outbuf	very little needed if at all. Very fast machines will benefit from output buffering, slow ones will choke unless the baud rate is low. At 115200 most machines cannot cope with output buffering. My 486/66 gets a major speedup with output buffering at 38400.

Other parameters can follow, the parameter name MUST BE GIVEN:

irqN	irq number (use 4/3 as usual)
baseNNN	port hardware address (use 0x3f8/0x2f8 as usual)

Example:

```
dncom.2:115200:n:8:1::4000::irq3:base0x2f8
```

Note that no ’xmode’ was given and no ’output buffering’.

If you prefer, you can use SLIP when you have SLIP8250:

```
slip8250 0x65 -h SLIP 3 0x02f8 38400 10000
fly8 r dnslip.1:int=0x65
```

This is the same as

```
fly8 r dnslip.1
```

because the driver will then be searched for.

Slip is not too fast, you need both machines to be at least 386DX for full speed and even then it is touch and go. It does no output buffering.

13.1.2 network-based

If you have a ’real’ network then install your favourite packet driver and do

```
fly8 r dnpkt.1:pack=1408
```

In this mode you can have as many players as you wish. The program talks packet level. You can also use the EtherSlip driver to play head to head with this driver, however direct com access is more efficient.

If you are runnning a unix system with the fly8udp server on it then you can use the msdos udp over packet driver. This will allow you to join the other unix players. The unix server MUST be on the same subnet for this driver to work.

```
fly8 r dnpcudp.1:pack=1408:ip=192.0.2.4:sip=192.0.2.5
```

Note how you must give your own ip (ip=) and the server ip (sip=). Your own ip can be anything but your sysadmin must allocate it, while the server ip is the actual ip for the host that is running the fly8udp server.

By default this driver uses port 0xf8f9 (change with port=) and expects the server to be on port 0xf8f8 (use sport= to change it).

The djgpp version accepts a parameter 'nbufs=' that specifies the number of receive buffers to allocate. The default is 20. Fly8 accepts packets very often and it is unlikely that you will need to increase this parameter under normal circumstances.

13.2 unix

13.2.1 FIFO

The FIFO driver will allow communication using FIFOs as well as a tty serial line.

This driver provides a head-to-head capability through FIFOs or other similar steam oriented facility. For one, it will allow connecting through a serial port (or other '/dev/tty' type connections).

Basically, you nominate an input file (if=) and an output file (of=) and Fly8 will use these. For example, this is how you start a two player head-to-head on a pair of FIFOs:

```
fly8 r N1 Tone dnfifo.1:if=ff81:of=ff82:pack L1.log
fly8 r N2 Tone dnfifo.1:if=ff82:of=ff81:pack L2.log
```

The two fifos can be created as

```
mkfifo ff81
```

or as

```
/usr/etc/mknod ff81 p
```

Note that the two programs run with different user names (N*) but share the same team (T*). This means that they cannot fight each other. Use a different team name for dueling.

Also note that a different log file is used by each (L*). This is needed only if both run from the same directory.

To play accross a serial line, simply start this on both machines

```
fly8 r N1 Tone dnfifo.1:if=/dev/ttyS1:of=/dev/ttyS1
```

/dev/ttyS1 is the serial line connection name on this end. You will use a different user name (and maybe team name) on the two ends too.

You will first need to set the serial line up, for example:

```
stty speed 38400 raw crtscts </dev/ttyS1
```

This setup can connect to the msdos serial driver. On msdos use these parameters:

```
fly8 r N2 Tone dncom.2:38400:n:8:1::4000:4000
```

This will use com2 (use 'com.1' for com1).

13.2.2 UDP-level

This driver is based on a server (fly8udp) program to which each player connects. The server will re-distribute all messages as necessary.

The order of events is as follows:

Start the server:

```
fly8udp &
```

Put the host name where the server runs in the fly.ini parameter. Here is an example for when the server is running on the local host:

```
r
dnudp.1:server=localhost:pack=1408
```

The 'r' will enable the Fly8 networking in general.

The ':pack' option tells Fly8 to collect multiple logical packets into one physical packet, please always use it with this driver.

The above option can be put on the command line too, have a look at the sample script 'flyudp'.

Now start Fly8 as usuall. Once running, check to see who is attached to the server with the ping request 'Esc n p' command. If someone replies then you can connect with the play request 'Esc n y'; answer '*' (all) to the prompt or select a specific user.

Note that if you do not talk to the server for 10 seconds you will be purged from it's table. This means that if you are the first user then you will not be able to keep your registration, the first two players need to connect (both) within this 10 seconds time window.

To control the server use the 'udpmgr' program. Start it as

```
udpmgr localhost
```

where 'localhost' is the default host where the server is running, subtitute the correct name otherwise. Use this command to shut the server down:

```
shutdown
```

and use this to stop the manager itself (if it does not drop out by itself at this point)

```
end
```

You can request the server to print it's stats with

```
stats
```

13.3 usage

OK, you started the program with networking. It really enjoys it but then it is a computer. You want to enjoy it too. Here is how. The Net menu has a set of commands for managing your connections with other players.

Some of the requests in this menu will need you to identify a player. A list of players is displayed and you should select one by entering their name. If you Enter '*' then you will select all of them. If you Enter ´+' then you will select all of your team members. If your response is null then you select none. If there is only one player then it wil be selected automatically. Otherwise, you can enter the start of the name and team so that the first partial match will be selected.

A message 'no net' means that the program was started without the net option (or possibly all net connections are inaccessible).

A message 'no player' may be displayed which means that there are no players in the needed status (eg. you try to 'quit' but you are not currently playing with anyone).

ping
: This is a broadcast ping to all players. All other programs will respond with their identification. Now you know who is active on the net. How do you tell who's who? the parameter -Nxxx supplies your handle (nickname, etc.) and you will be known by it in this game.

play
: Request to play with another player. If there is only one then the program will go ahead and establish connection. Otherwise you will be prompted to select a player from a list. To have the program know who is playing you should have used the P command first.

quit
: Quit playing. If there are more players then you will get a list to choose from. An empty response will assume you want to quit all players.

message
: Send a message to a player. Right now, an echo message will also be sent and the turnaround trip time (in milliseconds) will be shown. This time is end-to-end including program delay.

In addition to the above commands which you will use from time to time, there are a number of options on this menu which you may want to set at program startup (or wish to continue to handle manually).

accept
: If you are in 'manual reply' mode then you will get messages about players who want to play with you. Use this option to accept them.

decline
: If you are in 'manual reply' mode then you will get messages about players who want to play with you. Use this option to decline their request.

auto accept
: Set the game to always accepy playing requests.

auto delcine
: Set the game to always decline playing requests.

manual reply
: Set the program to ask you to accept/decline playing requests.

auto connect
: This option will attempt to connect to all net players continuously. It will also accept all play requests automatically. If you plan to use any communication then it is a good idea to always enable this option - just add it to the startup macro in fly.max.

When you exit the program it will automatically quit all players and notify the net of your exit.

If long delays are observed then a player may be automatically timed out. You will see messages about this. Use 'play' to re-establish contact. Proper use of 'quit' and 'play' will let you pop out of sight in danger and re-join in a more favourable position :-)

> IMPORTANT NOTE: the comms at this point is one-on-one. If you connect to three other players then they all show in your world and you show in their but they do not know of each other unless they establish connections too. If a group plays and each one joins with a global ´play' then everyone will know about everyone else.

There is a need for proper 'game' management (co-ordination), now completely lacking.

14. Files

14.1 fly.ini

This file serves as an extension to the command line options. The command-line option 'I' can set a user selected file name, this option MUST be on the command line.

Fly8 acquires parameters from three sources:

The ini file
: One parameter per line. Leading blanks are skipped, then first blank terminates the parameter. '#' denotes a comment line.

  The file is searched for in the current directory, then in your home directory (using the HOME variable) then in each directory along your PATH.

The environment variable FLY8
: Parameters are separated by a semicolon.

The command line options
: The usual rules apply, but the traditional '-' is NOT required.

These sources are processed in the above order and later parameters override earlier ones. The parameters consist of one letter followed by a value. Some options are grouped, and then two letters identify the parameter.

Numeric values are expected as hex (0x*) octal (0*) or decimal (otherwise).

h
: Help - prints a list of options. This option was removed, and it now tells you to read the doco...

FFilesPath
: Path where all files are to be found. Should be used only on the command line.

IIniFile
: Specifies the name of an init file.

  The default is 'fly.ini'. The file is searched for in the current directory, then in the home dir (uses HOME env. var.) then the PATH directories are checked. Only the first file found is used. Should be used onlye on the command line.

LLogFileName
: The file to use as the Fly8 log. It is appended to.

MMacroFile
: Name of keyboard macros file. [default is 'fly']

XNavFile
: Name of nav data file. [default is 'fly']

YLandFile
: Name of landscape file. [default is 'fly']

dpPointerDriver
: The pointing device can be one of: keypad, mouse, astick, bstick. But each version (port) of fly8 has different ones.

  Example of pointer parameters:

  dpAstick:hat:sh=3:d=046

  This means "we have an analog hat (FCS) and it is on channel 3 (2nd stick X). we want to not debounce buttons 0 (trigger) 4 and 6 (hat up down) - these will fire continuously.

The Fly8 X axis controlls the roll while the Y axis controls the pitch. The rudder is on X2 and the throttle on Y2. A Thrustmaster analog HAT is also on Y2. These assignments can be changed using the following format: ":sh=3" will use analog channel 3 (X2) as the source for the throttle. You can swap the X and Y using ":sx=2:sy=1". You can change the direction of an axis using this format: ":st=-4" which means the throttle should be interpreted in reverse. This has the same effect as appending a hyphen to the 'ttl' option.

Each pointing device can have more parameters. When a list of buttons is expected you can use a simple list or include ranges, for example ":a=1-4" will make buttons 1, 2, 3 and 4 sensitive to the Btn-Alt mode. If the '-' is at the start then a '0' is assumes before it while at the end a 'z' is assumed. This means that ":a=-" will set all of the buttons.

common options

d=
: Do not debounce these buttons - make them repeating. Example 'd=0' will cause button 0 to repeat its associated command while pressed.

r=
: Do NOT issue key release command for these buttons. Example 'r=13' will disable issuing key-release on buttons 1 and 3.

a=

c=

p=

s=
: Set the buttons to respondto Alt, Ctrl, Special and SHift modes respectively.

linear
: The 'x' and 'y' inputs will be interpreted as a linear scale. The default is to apply a log scale such that the sensitivity of the stick increases as you move away from the reference point (center).

keypad
: (No optional parameters are defined)

mouse

smx=
: Sets the sensitivity of the mouse x axis. Default is 10.

smy=
: Sets the sensitivity of the mouse y axis. Default is 10.

stick

ian=
: The size (in % of active range) of the region where the reading may be unstable. This 'idle' parameter nominates the area around the center of the joystick and the edges of it (and of the throttle) where the reading will be treated as it is was still at the center (or the edge).

  The 'an' above nominate the axis and stick number as: ix= main stick x, iy= main stick y, ir= rudder (2nd stick x), it= throttle (2nd stick y), ih= FCS hat (2nd stick y).

hat
: Use y2 as the FCS hat. The positions are called buttons 4 (up), 5(right), 6(down) & 7(left). Do not use this option for the CHPro, see the special 'chpro' option later.

ttl
: Use y2 as throttle. if you use ":ttl-" then the throttle will be read such that full-range is zero-throttle and zero-range is full-throttle (which is how the CH and WCS do it).

rdr
: Use y1 as rudder. if you use ":rdr-" then the rudder will be read such that the two ends are reversed.

zero
: Read no buttons. This is used when the buttons are delivered through the keyboard using a WCS or FLCS.

four
: Read all 4 buttons. These are called 0,1,2 & 3. Do not use this option for the CHPro, see the special 'chpro' option later.

chpro
: Used to indicate that you are using a CH Pro Flightstick which has four buttons and a hat and has a different encoding scheme than the TM FCS.

rd=
: Read the joystick this many times instead of once. See also next option.

dly=
: Wait some time between multiple reads. The last two options are usefull if there is a lot of interruptions (network, multiuser system etc.). The delay may be necessary if doing multiple reads since some joystick ports are slow to reset.

count
: Do not use the timer when reading the stick, just run a loop counter. Usefull if your machine/joystick-card combination is so fast that the timer does not have enough resolution. This method is however more sensitive to interruptions, use 'rd=' and 'dly=' to overcome these. Also, under mswin the timer may not function properly.

game
: Do not use the <joystick.h> device but use the <game.h> device driver.

gp
: Use the serial line /dev/gp0 to read a Colorado Spectrum Gameport (Workstation) joystick port.

dvVideoDriver
: The software video driver. These vary between machines.

- MSDOS: grQc (default), grFast or grVESA (VESA/VBE). There may be a grbgi if compiled with borland but it is too slow.
- DjGpp: grAsm (using fast assembly level graphics), grDJ (using distribution DjGpp graphics library).
- MSWIN: grGDI, grBitBlt, grWinG
- UNIX: grx (PixMap based), gri (Image based) or grSVGA (svgalib).
- AMIGA: gramiga [amiga port not available!]

dnNetDriver
: Network access through driver 'NetPort'. Both msdos and unix have network drivers at the moment. Look earlier at the networking section.

dkKeyboardDriver
: Keyboard device name. Usually there is only one defined so leave this parameter out.

dsSoundDriver
: Sound device name. Usually there is only one defined so leave this parameter out.

dtTimerDriver
: Allowes you to select, and pass parameters to, the timer device. You will probably never use this option.

PMainObject:options

DDroneObject:options
: The main and drone object type. If absent then drone type defaults to the main type. While any object name can be used (for most *.vxx present in the Fly8 directory) currently only "plane" responds to controls and it supports a dynamics type option: classic, basic or f16. The default is "plane:basic".

  An experimental Car type (no options) is being put together a an example of how to add new types.

VVideoModesFile
: The video modes file to use, e.g. ’Vgrvesa’ means to use the file ´grvesa.vmd’. [default is ’fly.vmd’]

mVideoMode
: The video mode. This is one of the modes defined in the .vmd file you use.

iString
: Initialization keystrokes. The string is a list of macros to execute at startup. The default is Ctrl-A. The string ’iCaezAb’ will execute, in order: Ctrl-a Ctrl-e Ctrl-z Alt-b. Only Ctrl- and Alt-type macros can be used here. Macros are defined in the *.max file.

bn
: Windows configuration. The ’n’ is the same letter as used on the ’Windows’ menu.

r
: Activate network playing (used to be ’support Remote players’).

NHandle
: You will be known as ’Handle’.

TTeam
: Your team’s name.

HHome
: Your home runway name (selected from fly.nav file).

o
: This is used to prefix a list of options (separately described as ’oq’ etc. later). You can embed ’-’, ’+’ and ’^’ to set, clear or toggle an option respectively. For example "o-qv+l" will turn off sound and verbose, and turn on landscape.

oc
: Use the block-Clear function of the graphics driver between frames rather than erase the old frame by retracing it in background color. The retrace way is often much quicker with Fly8 (which is mostly wireframe) but some modern graphic devices can erase the screen fater than that.

ol
: Add some landscaping (very rudimentary).

oq
: No sound

os
: Start the program in "solid sky" mode, see "Screen.Solid Sky" in the menus chapter.

ot
: See Options.Trace in the menus chapter.

ov
: See Options.Verbose in the menus chapter.

zNDrones
: Screen-blanker mode (with 'NDRONES' drones). You are put insto auto-pilot auto-kill mode, window borders are not shown.

naAutoConnectRate
: The (milliseconds) between network auto-connections. This is effective only if net auto connect option is active. Default is 5 seconds.

nbLineBufferSize
: Max num of segments in the display list. Default 5000. Used to reduce memory shortage on msdos. Should be set to higher in environments that have no stupid memory limitations.

nfFrameRate
: Minimum time (milliseconds) between frames. This allows you to limit the frame rate, and is useful if you do not have double buffering. Default is 10ms and it is also the lowest allowed.

niDynamicsStep
: Max time for single step plane dynamics. Used to reduce simulation errors by breaking long periods into a sequence of shorter ones. Default is 100ms.

nkSkyLines
: How many lines to use to show the sky. Default is 50.

nlLogFlushTime
: Time in seconds for log file flush after a log message. use 'nl0' to force all logs to go to disk asap.

nmMacros
: Max number of keyboard macros. Default is 256.

nrRecallListSize
: Number of entries in the history recall buffer. Default is 20.

ntTimeLimit
: Time limit in seconds to auto shutdown (use in batch demos).

nuUpdateRate
: Minimum time (milliseconds) between network status update. By default Fly8 sends update information for each object at a regular 60m interval. If you use a slow network (modem) with many users then you may want to reduce the update rate (e.g. set it to 125ms to 8 updates/sec).

cColor
: Sets any one of the color definition. The color name is the same as used on the color menu. The value is as RGB, each componnent is 8 bits. Example: ch0x60c060 set hud-low ('h') color to light green (R=60, G=c0 B=60, all in hex).

14.2 fly.log

This file logs the activity of the program, problem messages and final stats. The command-line option -L can set a user selected file name.

14.3 fly.mac

This file is read at program startup and written at program shutdown. It is the list of keyboard macros. Use the 'mac2max' program to list the contents of this file. At the moment there is no macro editor so you should use the redefine-macro for updating. The command-line option -M can set a user selected file name. It is easier to maintain the fly.max (see later) manually, the fly.mac file may be removed in the future.

14.4 fly.max

This file is read on program startup and defined keyboard macros. If it does not exist then fly.mac is read, if there is none then no macros are defined (at this point). During shutdown all keyboard macros are written to fly.mac (will OVERWRITE the original!).

fly.max is a text file while fly.mac is a binary format.

It used to be that you could only define macros at run time using the F7 facility. However, now you will probably edit fly.max instead. However, if you do add new macros interactively then you can use:

```
mac2max fly.mac >new.max
```

and now with an editor copy from new.max what you want into fly.max. If you run the program again then fly.mac will be overwritten with the content of fly.max.

If you find yourself running fly8 and forgot to save the fly.mac (into fly.max) then shell out (with the ! command) and make a copy of fly.mac. The shell only works if you have enough memory. If the screen goes blank on shell then assume that it worked and type (blind) something like:

```
copy fly.mac xxx
exit
```

and the program will resume. Good luck.

fly.max is a list of key definitions. Each definition starts with the Def keyword and is followed by the key being defined and a list of keystrokes (the macro itself).

Things can continue on multiple lines. A # will cause the rest of a line to be ignored.

A key name is a list of shifts followed by a key name. The recognized shifts are:

Shift

Ctrl

Alt

Btn
: Button press. This is the usual button definition.

Brl
: Button release. Will only be effective if the button was defined to report key releases (the default, but see ":r=" in pointer parameters).

A key name is either a single character (which represents itself and IS CASE SENSITIVE) or one of the special names (not case sensitive):

F1

F2

F3

F4

F5

F6

F7

F8

F9

F10

F11

F12

Left

Right

Up

Dn
: alias Down

PgUp
: alias PageUp

PgDn
: alias PageDown

Home

End

Ins
: alias Insert

Ctr
: alias Center. Unshifted numeric keypad 5 key.

Del
: alias Delete

Sp
: alias Space

Bell
: alias \a

Bs
: alias Ro, Rubout, \b

Esc
: alias \e

Ff
: alias \f

Enter
: alias Ent, Cr, Ret, \n

Nl
: alias \r

Tab
: alias \t

Vt
: alias \v

\\
: A single '\'

Note that some keys have both a long and a short name and some have an escape form too.

You can use string notation:

```
Def Alt g "\esph=\e40c080\n\e\e" # set HUD to green
Def Alt y "\esph=\e808000\n\e\e" # set HUD to yellow
```

strings can continue on the next line.

Buttons are named 0-9 and a-z. The usual drivers will assign them in the following manner (use wcs.adv and flcs.b50 to configue these devices).

0
: trigger, mouse right button (FLCS TG2).

1
: other button on a two button pointer (FLCS S2).

2,3
: rest of buttons on a 4 button joystick (FLCS S1 and S3).

4,5,6,7
: main hat positions: up, right, down, left.

8
: FLCS TG1 (partial trigger) button.

9
: FLCS red paddle (S4) button.

a,b,c,d,e,f
: WCS bottons 1, 2, 3, 4, 5 and 6.

g,h
: WCS rocker up and rocker down.

i,j,k,l
: FLCS hat 2 up, right, down, left,

m,n,o,p
: FLCS hat 3 up, right, down, left,

q,r,s,t
: FLCS hat 4 up, right, down, left,

Note that these names depend on the pointer driver that you use. The mouse uses only 0 (left) and 1(right).

14.5 fly.nav

Lists the navigation waypoints which have ILS beacons. Should really be merged with the landscape file. Maybe one day. It is created from the fly.nac file using an awk script (in the 'parms' directory).

This is a very basic file that nominates the navigation points. Fly8 is not strong in this area, however you can activate the ILS system which will use one of these beacons.

The file structure is as follows:

First you nominate the number of points you are defining (I know, I am lazy).

Next come the beacon information:

```
"A18",
   V(0), V(0), V(0), /* [x,y,z] meters */
   149*60+10, -(35*60+20), /* longitude, latitude minutes */
   0, -500,          /* localizer relative [x,y] meters */
   0, 1500,          /* glide-path relative [x,y] meters */
   0,                /* heading of forward beam angle */
   D(3),             /* pitch of beam angle */
```

The name is whatever you call it (not case sensitive). The position is like in the object positioning in the .lnc file. The long/lat data allows the program to track your position, it is NOT checked against the previous line!

The next line specifies the localizer position (assumed to be level with the object) as [east, north] and then the same for the glide slope origin point. These points are where the transmitting antenna is located.

The next line gives the heading of the localizer beam (0 means north), and the next is the glide slope angle (3 means 3 degrees slope).

An empty name must close the list.

14.6 fly.lnd

Defines the contents of the landscape. It is created from the fly.lnc file using the C pre-processor and an awk script.

It is mechanically built from fly.lnc. It describes the contents of the landscape.

This file is a list of objects and their placement. You can place the standard objects (defined in separate .vxx files) as well as define new objects. The new objects will not be animated are are used for the fixed landscape.

And object placement has the following format:

```
O_RUNWAY, CC_DEFAULT,   /* runway A */
   V(0), V(0), V(0), /* position x, y, z */
   D(0), D(0), D(0), /* heading, roll, pitch */
```

The first line names the object and assigns a color. CC_DEFAULT will let the object keep it’s standard color.

The second line gives the objects position as x (east) y (north) and z (up) in meters (you can use fractions).

The third line gives the objects orientation in terms of heading (north is zero) roll (zero means level, 90 means a quarter turn right and 180 means upside-down) and pitch (zero means level, 90 mens straight up while -90 is straight down).

To define a local object use the format:

```
O_DEFINE+1, V_METERS,   /* river */
   6,
   V(-4000), V(-2000), V(0), V_MOVE,
    ...
```

The first line is the object name (each object is defined once and is assigned a number, this one is ’1’) and the data resolution: V_METERS means that the data is to be kept in meters (no fractions) while V_FINE means that internal data representation is in 1/16 of a meter.

Next you nominate the number of vertices (must me right!) and following are the vertices. Each is an (x, y, z) trio and an indication as to the visibility of that vertex.

To place a local object use a name of the format O_LOCAL+1 (for the object in the above example).

14.7 *.vxx

This file can mechanically built from *.vx. It defines the shape of an object.

The .vxx file is a simple list of numbers. The first line gives the number of vertices in the object and nominates the resolution of the following data: 2 means 1=1meter while ’1’ means 1=1/16meter.

The data is a list of (x, y, z) points followed by 1 (line not visible) or 2 (line visible).

No punctuation marks are used on this file and data cannot be moved from the nominated lines (2 numbers of the first line and 4 on the rest).

This file can be built from a more flexible file (usually stored as a .vx) which is then processd by the C preprocessor an an expression evaluator which builds the .vxx file.

Some of the .vx files look line C initializers - this is because objects used to be internal definitions. There is no point in making this any nicer since the whole object representation will at some stage be completely upgraded to a full polygon based structure.

14.8 *.prm

These files define the various aircraft parameters. Different parameters are used by different flight models, but if you modify a parameter file it is best to set ALL parameters so that all model will work.

The file that you create is the *.prc, which is then processed into a *.prm file (which is then read by fly8). The fly8 options ’P’ and ’D’ select which parameter file will be read (so ’fly8 Pf16’ will read the ´f16.prm’ file).

The *.prc file has short comments against each line which should give some idea about each value. Note that the data is explicily integer. This means that fractions are kept as fixed point. Also, other data (line angles and lengths) have a fraction part; however, you should not worry about this, just make sure that you specify the data using the correct format. If you see a drag coefficient specified as F(0.02) then keep using the F() format - this will convert the fraction to the fly8 representation. Following are the types recognized (look at ’parms.h’ for the definitions if you wish):

F(x)
: Stores x as a fraction. x should be kept in the range -1.0 to 1.0.

F10(x)
: Stores x as a scaled fraction. x should be kept in the range -10.0 to 10.0.

V(x)
: Length representation. Will be stores as meters with four binary fraction bits (just over 6cm).

VV(x)
: Special format for high resolution lengths. x should be kept to a reasonably small value (-128.0 to 128.0) but the resolution is to 8 bits (about 4mm). This is used in giving delicate landing gear dimentions.

G
: Constant of gravity (9.810). Useful for defining forces.

D(x)
: Stores x degrees in fly8 format.

DV(x)
: Stores x degrees/sec in fly8 format.

I10(x)
: Stores a large integer with -1 decimal point (that means that a weight of 12876 is stores as ’1287’).

In all of the above ’x’ can be any number (integer or floating point) or any expression of these. The

expression evaluator understand most of the standard C operators. Also, you can use the C preprocessor syntax in this file since it is being run through it anyway.

Note that fly8 expects the parameters in the correct format so never change the format notation in the *.prc file, just change the values.

The most important parameter is the first item in the ’10 options’ list (towards the end), it selects the flight model program.

[much more stuff needed here]

14.9 fly.vmd

This file defines the parameters for the available video modes. In windowed environments this is the startup mode and you can then resize the window. The command-line option V can set a user selected file name. The readme should itemize the available mode files for the different environments.

The file is a list of mode lines. Each line has the following format:

name
: The name of mode, any string (should be unique in this file).

int10 value
: For pc/grfast: the supplied text files should have enough info to set these up for the supported cards.

  For pc/grqc: this number is the c library id for the mode. [this driver is obsolete]

  Not used on other platforms.

number of colors
: Must be 256 on pc/vga.

min x
: The physical offset of the left margine of the screen. 0 on pc/vga.

min y
: The physical offset of the top margine of the screen. 0 on pc/vga.

size x
: Screen width in pixels

size y
: Screen height in pixels

number of pages
: How many full pages (WxH) will fit in your memory. Fly8 may use 2 for double buffering.

physical screen width
: Actual screen measurment (e.g. millimeters) for aspect ratio adjustment.

physical screen height
: Actual screen measurment (e.g. millimeters) for aspect ratio adjustment.

font width
: 8 (obsolete)

font height
: 8 (obsolete)

flags
: This is a general purpose flags word used differently by each video driver. For example, msdos/grfast (and many others) uses the ’1’ bit to indicate a need to wait for vsync on page flipping.

14.10 data directory

This directory is where one build the *.lnd, *.prm and *.nav files from the free form input. The 'data' directory can be located anywhere but I find is easy to keep is as a subdirectory of the Fly8 game directory.

The files here are:

*.lnd
: built from *.lnc, this is the landscape desciption.

*.prm
: built from *.prc, this is the plane parameters file.

*.nav
: built from *.nav, this file gives some navigation data.

The steps to follow in building the files are described below. The 'build' batch file is available for msdos but a similar one should be done for unix etc. It simple automates a rather simple process.

1
: Edit the source file (lnc/prc/nac). Use whatever editor you prefer, these are plain text files.

2
: Do 'build File Type' where File is the name of the file you edited (without extension!) and type is lnd/prm/nav as appropriate. This script will run your file through the C pre-processor and then through an awk script (expr.awk).

3
: Copy the new file (lnd/prm/nav) to your game directory. Or you can give the directory on the build 'build File Type CopyToDirectory'.

4
: Change to the game directory and play. The new file will now be used.

15. Acknowledgements

Ross Johnson (rpj@ise.canberra.edu.au)
: Ideas, testing, networking know-how, X11 help. Doco typesetting and review.

Mike Taylor (miket@pcug.org.au)
: Amiga and Windows ports.

Paul Thomas (paul.thomas@uk.ac.ox.eng)
: Contributed a number of plane shapes.

Chris Collins (ccollins@pcug.org.au)
: Contributed much of the Win95 port, as well as multimedia items.

16. Misc Notes

When you eject ('E') you will find yourself on a parachute. When you land, you will move to a new plane. You can accelerate the descent with ´D' but if the plane did not yet crash then you will (have to) wait on the ground until it crashes and a new one is provided.

The IBMPC uses a timer chip with three counters. Fly8 reads counter 0 to get high resolution interval timing and writes counter 2 to generate speaker sound. Counter 0 is often set to the wrong mode by various programs (some examples: Landmark 1.14 sets the mode to 3 while version 2.00 sets it to 2) . The standard is mode 3, but it is sometimes set to 2. Fly8 needs to have the timer in the mode 2 or 3 or it reads bad

timing information.

The program 'gettimer' will report the current mode. If it is 36/34 or b6/b4 then the mode is correct. Fly8 was upgraded to support the most common two modes and can actually operate with non-standard timer settings (I should document these special parameters somewhere).

The program settimer will set the timer to mode 3.

17. Known Problems

- On a 486DX50 the serial driver fails at 115200 but is ok at 57600. Output buffering at high speed looses the comms. It is now fixed with a kludge in the comms driver.
- On fast machines attempting to read the joystick twice in a row produces unexpected results. A special delay option was introduced to get around this problem.
- The stroke character generator will not handle stroke sizes above 128 pixels. May be a problem if you try running at very high resolutions (say 1600x1200).
- On slow machines the program may fail with a divide overflow. It also may happen (rarely) on fast machines. This will be fixed gradually as sensitive areas of the program get cleaned.

Table of Contents