(2)

AD-A222 614

WRDC-TR-90-2005

PDWAP - A PRELIMINARY DESIGN AND WEIGHTS ANALYSIS PROGRAM FOR AEROSPACE VEHICLES

L. R. Carreiro, 1Lt, USAF

Analysis and Applications Branch
Advanced Propulsion Division

January 1990

Final Report for Period October 1987 to December 1988

Approved for puplic release; distrubution unlimited

AERO PROPULSION AND POWER LABORATORY
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6563

LOUIS R. CARREIRO, 1LT, USAF
Aerospace Development Engineer
Analysis and Applications Branch

JOHN L. LEINGANG, Chief
Analysis and Applications Branch
Advanced Propulsion Division

FOR THE COMMANDER

WILLIAM G. BEECROFT
Director
Advanced Propulsion Division
Aero Propulsion and Power Laboratory

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Unlimited Distribution Approved for Public Release
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-2005	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION WRDC Aero Propulsion and Power Laboratory 6b. OFFICE SYMBOL (If applicable) WRDC/POPA	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB OH 45433-6563 PHONE: (513) 255-4171	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.
62203F	3012	08	93

11. TITLE (Include Security Classification)

PDWAP - A Preliminary Design and Weights Analysis Program for Aerospace Vehicles

12. PERSONAL AUTHOR(S)

L. R. Carreiro

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
FINAL	FROM Oct 87 TO Dec 88	January 1990	90

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

Weight Analysis, Preliminary Design, Aerospace Vehicles, Hypersonic Vehicles, Single-Stage, Earth-to-Orbit Vehicle Design.

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This is the User's Manual & Formulation Report for the program PDWAP (Preliminary Design and Weights Analysis Program). The program runs on XT/AT compatible microcomputers. It is designed to provide a complete vehicle size summary and weight statement based on very rudimentary user inputs, in a very short runtime (i.e.: less than 3 minutes). The preliminary design phase is based on current trends in hypersonic-aerospace vehicles. The weights analysis, based on the WAATS code, can be used separately, and can handle a wide range of vehicles, such as fighters, transport, boosters, shuttles, etc. The program is interactive and menu driven, providing results in minimum time and investment.

Key words:

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Louis R. Carreiro, 1Lt, USAF	22b. TELEPHONE (Include Area Code) (513) 255-4171 22c. OFFICE SYMBOL WRDC/POPA

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

FOREWORD

This report describes the use and development of an aerospace vehicle preliminary design and weights analysis program, PDWAP. The program is primarily intended for aiding in the study of aerospace and hypersonic vehicles, although it can be used to perform weights analysis on virtually any flight vehicle. The program uses an interactive menu system, and an easy to use file system to provide the user with the answers in the shortest possible time.

Detailed instructions are given for the computer program and it's formulation and algorithms are discussed. Example cases are included in the user's manual. The program is intended to be used on an IBM or compatible XT/AT class microcomputer and is available on a 5¼", 360 Kb floppy disk from the author. Requests from non-government agencies should include a completed copy of the Software Release Form, included at the end of this report.

The flight vehicles used as examples in this report, unless otherwise stated, are not intended to represent any known or project vehicle or level of technology. No inference should be made that the example vehicles typify current, or projected capabilities.

This work was performed in the Analysis and Applications Branch, Advanced Propulsion Division of the WRDC (Wright Research and Development Center) Aero Propulsion and Power Laboratory. This work was done under in-house work unit 30120893. The work was begun in October 1987 and completed in December 1988.

The author, 1Lt Louis R. Carreiro, wishes to thank John L. Leingang and Wayne A. Donaldson for their help during the development, documenting, and testing of PDWAP.

Table of Contents

Table of Contents (cont.)

List of Figures

Abbreviations and Symbols

A_i	Adustment coefficient array, i is the index
A_{ctr}	Thrust scaling factor
A_{ICapt}	Total capture area of intakes
A_{Ratio}	Area ratio of rocket engine
b	Geometric wing span
b_{st}	Structural span of wing along half cord
F_{Mach}	Mach factor
Gal	Number of gallons of storable fuel
G_{Fct}	Geomtric factor
h	Vehicle height
L_{Body}	Length of fuselage
L_{Inlet}	Length of inlet
L_{Ramp}	Length of vari-ramp
N_{Intake}	Number od intakes
N_{re}	Number of rocket engines
N_{rj}	Number of ramjet engines
N_{trj}	Number of turbo-ramjet engines
N_{Tanks}	Number of fuel tanks
P_{high}	Upper engine design pressure
P_{low}	Lower engine design pressure
P_R	Pressure ratio at design conditions
P_{t2}	Total pressure in engine intake
P_{t0}	Total freestream pressure
q_{max}	Maximum dynamic pressure

Abbreviations and Symbols (Cont)

r_b	Radius of body
r_f	Radius of forward cone
r_t	Radius of LOX tank
S_{Body}	Surface area of vehicle
S_{cyl}	Surface area of body halh-cylinder
S_{Fair}	Surface area of aero fairings
S_{fc}	Surface area of forward cone
S_{FuTk}	Total surface area of fuel tank(s)
S_{FuTk1}	Surface area of forward fuel tank
S_{FuTk2}	Surface area of aft fuel tank
S_{Horz}	Surface area of horizontal control surfaces
S_{OxTk}	Total surface area of oxidizer tank(s)
S_{rc}	Surface area of rear cone
S_{TPS}	Surface area of thermal protection system
S_{Vert}	Surface area of vertical control surfaces
S_{Wing}	Surface area of wing (planform)
T_{Root}	Wing thickness at root
T_{TOT}	Total thrust of all engines
T_{re}	Thrust of rocket engine
T_{rj}	Thrust of ramjet engine
T_{trj}	Thrust of turbo-ramjet engine
T_{Fct}	Temperature function
V_{fb}	Volume of forward body
V_{fc}	Volume of fuselage center section
V_{fu}	Volume of fuel

Abbreviations and Symbols (Cont)

V_{FuTk}	Total volume of fuel tank(s)
V_{OxTk}	Total volume of oxidizer tank(s)
V_{rb}	Volume of rear body
W_{a}	Weight flow of air in intake
W_{ACS}	Weight of attitude control system
W_{ACSFu}	Weight of attitude control system fuel
W_{ACSOx}	Weight of attitude control system oxidizer
W_{ACSP}	Weight of attitude control system propellant
W_{ACSRe}	Weight of attitude control system reserves
W_{ACSTk}	Weight of attitude control system tankage
W_{ARef}	Reference weight flow of air in intake
W_{Aero}	Weight of aerodynamic control surfaces
W_{Avonc}	Weight of avionics systems
W_{BPump}	Weight of boost pump
W_{Basic}	Weight of basic body structural
W_{Body}	Weight of body structure
W_{CProv}	Weight of crew provisions
W_{Cont}	Weight of contingency allowance
W_{Cover}	Weight of TPS cover panels
W_{Crew}	Weight of crew and life support equipment
W_{Dist1}	Weight of fuel distribution system 1
W_{Dist2}	Weight of fuel distribution system 2
W_{Drain}	Weight of fuel drainage system
W_{Dry}	Dry weight of vehicle

Abbreviations and Symbols (Cont)

W_{Elect}	Weight of electrical system
W_{Empty}	Empty weight of vehicle
W_{EngMt}	Weight of engine mounts
W_{Entry}	Entry, or insertion weight of vehicle
W_{FCont}	Weight of fuel control system
W_{FLoss}	Weight of fuel losses due to boil off, etc.
W_{FResv}	Weight of reserve fuel
W_{FTrap}	Weight of trapped fuel (ullage)
W_{Fair}	Weight of aerodynamic fairings
W_{FuCnt}	Weight of fuel container for non-integral
W_{FuSys}	Weight of fuel system
W_{Fu}	Weight of fuel
W_{FuelM}	Weight of main fuel
W_{Gear}	Weight of launch and landing gear
W_{Gimbal}	Weight of rocket engine gimbal system
W_{Horz}	Weight of horizontal surfaces
W_{HyPnu}	Weight of hydro-pnuematic system
W_{IDuct}	Weight of inlet ducts
W_{InFuT}	Weight of fuel tank insulation
W_{InOxT}	Weight of oxidizer tank insulation
W_{Inlet}	Weight of inlet(s)
W_{InsFT}	Weight of integral fuel tank insulation
W_{InsOT}	Weight of integral oxidizer tank insulation
W_{Insul}	Weight of TPS insulation

Abbreviations and Symbols (Cont)

W_{LGear}	Weight of landing gear
W_{Land}	Landing weight of vehicle
W_{Launch}	Weight of launch gear
W_{O2}	Weight of oxidizer
W_{OLoss}	Weight of oxidizer loss
W_{OResv}	Weight of reserve oxidizer
W_{OTrap}	Weight of trapped oxidizer (ullage)
W_{Ornt}	Weight of orientation and control system
W_{OxCnt}	Weight of oxidizer control system
W_{OxSys}	Weight of oxidizer system
W_{OxidM}	Weight of main oxidizer
W_{PLoss}	Weight of propellant lost
W_P	Weight of propellant
W_{PayLd}	Weight of of payload
W_{PrSys}	Weight of pressurization system
W_{PropM}	Weight of main propellant
W_{PropU}	Weight of propulsion system
W_{PwrSy}	Weight of power generation system
W_{re}	Weight of rocket engines
W_{rj}	Weight of ramjet engines
W_{ReFul}	Weight of reserve fuel
W_{ResP}	Weight of reserve propellant
W_{Resid}	Weight of residual propellants
W_{Seal}	Weight of fuel system seals

Abbreviations and Symbols (Cont)

W_{SecSt}	Weight of secondary structure
W_{Sep}	Weight of seperation system for multi-stage
W_{Spike}	Weight of intake spike
W_{Surf}	Weight of aero surfaces
W_{TO}	Gross take off weight of vehicle
W_{TPS}	Weight of thermal protection system
W_{trj}	Weight of turbo-ramjet engines
W_{ThrSt}	Weight of engine thrust structure
W_{VRamp}	Weight of veri-ramp system
W_{Vert}	Weight of vertical control surfaces
W_{Wing}	Weight of the wing
X_{LF}	Ultimate load factor
X_w	Length of wing
X_c	Length of forward cone (crew area)
X_f	Length of forward part vehicle
X_m	Length of midsection
X_r	Length of rear section

Abbreviations and Symbols (Cont)

f	Overall fuel to oxidizer ratio
f_{ACS}	Fuel to oxidizer ratio for ACS engines
$\backslash_{LE}$	Sweep angle of leading edge
π	3.1415926...
ρ_F	Weight density of fuel
ρ_{H2}	Weight density of liquid hydrogen
ρ_{O2}	Weight density of liquid oxygen
σ_f	Half cone angle of forward cone
σ_r	Half cone angle of aft cone (boat tail)

1 User's Manual

1.0 How top Use this Manual

This part of the report, the user's manual, is aimed at helping anyone to use PDWAP. It assumes that the average experience of the user is somewhere between computer avoider and computer wizard. This manual is geared toward the lower range of that spectrum, mostly because real computer wizards wouldn't read the user's manual anyway. If you are a new user, try reading through the manual. If you are a little more comfortable, probably a glance through the sections on preliminary design and weights analysis would help just to point out what the program can do and what input it needs.

The manual is divided up based on how you would go about using PDWAP. The first sections are concerned with getting the program and its files loaded onto a working disk, and then getting it started. The following sections describe how to use PDWAP in the order that a normal session would proceed.

The last section of the user's manual shows a sample case. The sample case is based on the Incremental Growth Vehicle (IGV). It is part of a study that McDonnell Douglas did to investigate the use of a high speed research vehicle for propulsion system testing. The vehicle was selected randomly of those available with weights documentation so a comparison could be drawn between the documented weights data and PDWAP's results. This vehicle was not run on PDWAP before the development of the documentation and serves as a good test of the program.

1.1 Introduction to PDWAP

PDWAP is a system of three programs that can be used to provide a weights analysis of any of a large group of flight vehicles. The user adjustable data file can be tailored to represent fighter and transport jets, rocket powered boosters, single stage Earth-to-orbit vehicles, as well as other possible configurations. The output of the program is a detailed vehicle summary and weight statement which can be used to check the effects of various configurations, and missions on the vehicle under study.

The PDWAP system is comprised of three programs. The first is the preliminary design program. This is used to scale a generic design vehicle (GDV) to match your supplied inputs. The geometry of this vehicle is then used to calculate the areas, volumes, etc. that are used later. The program then writes the calculated data to a file to be used by the second program.

The second program in the package is a modification of WAATS (Weights Analysis for Advanced Transportation Systems) by Glatts. It uses a large data file, created by using the first program or written by you, to perform a weights analysis using a component build-up technique based on historical data. The results are then be sent to a data file for viewing later.

The third program is actually a collection of routines that allows you to view either the data file or output file as well as send it to the printer for hard copy. It also has a file converter that updates your version 2.0 files to the version 2.1 format. Also, a DOS shell has been added to make it easier to use other programs while in the PDWAP system.

All three programs are contained in one large package, about 72K, and are controlled by a user friendly menu system. This allows you to move from one program to another without returning to DOS. Also, since the programs are combined in one package that is loaded in at start-up, there is no lag time involved when going from one program to another.

For a graphic description of the PDWAP system see Figure 1.

1.2 Requirements

1.2.1 Required Files

Included on the PDWAP disk are the following files:

PD.EXE	Program executable code
PD.DAT	Initializing data file used by PDWAP
PD.DOC	Documentation file for PDWAP
PD.BAS	TurboBASIC source code
PD1.COM	Used to view files
PDI.BAT	Installs needed files on hard drive

To run the program, PD.EXE, along with the files PD.DAT, PD1.COM are required. The other files are there only for your benefit. The source code is included so that you can modify the program to fit your needs. I would greatly appreciate hearing of any modifications as well as any suggestions for changes to the program.

1.2.2 Required Hardware

PDWAP will run on any IBM PC/XT/AT compatible machine. It can operate with or without the 8087/80287 math chip. The program can operate in a system with as little as 128K. Since the program reads and writes to the disk a few times during its operation, a hard drive or ram drive will speed it up.

PDWAP "Roadmap"

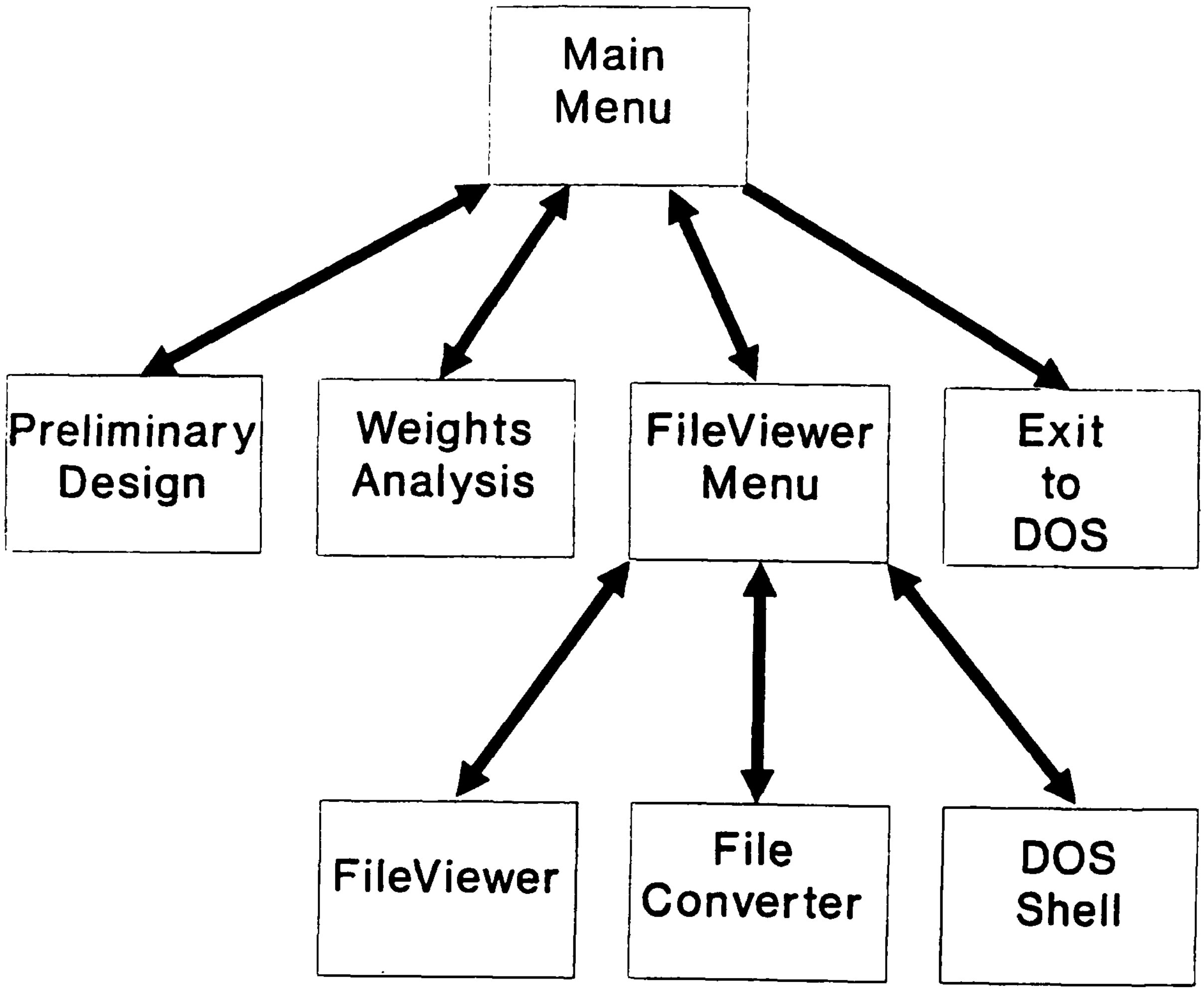


Figure 1

1.3 Getting Started

1.3.1 Installing PDWAP

Getting started with PDWAP is as easy as typing PDI at the DOS prompt:

A:>PDI [*path*]**<RETURN>**

where *path* indicates the drive and directory that you want to put PDWAP. This will make a working copy of PDWAP by copying PD.EXE and the other needed files to your working floppy or hard disk.

1.3.2 Running PDWAP

As mentioned above, running PDWAP is simple. At the DOS prompt type PD and then hit the <RETURN> key. You should then see the introduction page on your screen.

The program will then load into the machine and your screen will then look like Figure 2. This is the Main Menu screen of PDWAP. From here, you can use the functions of PDWAP just by using the menu, instead of having to remember numerous commands.

1.3.3 Using the Menu Interface

The menu system used on PDWAP was selected because of its ease of use, along with the speed it has over manual selection of the programs that make up the PDWAP system. To use the menu just select the PDWAP system. To use the menu just the PDWAP system. To use the menu just select the operation the PDWAP system.

To use the menu just select the operation that you wish to perform by using either the arrow keys or the corresponding function key. The title for the function will then be highlighted with brackets, like choice number 1 in Figure 2, and you just hit <RETURN> to activate that program. After using the function you have selected the program will return to the Main Menu.

Main Menu Screen

PDWAP ver 2.1

Preliminary Design and Weights Analysis Program

Main Menu

‹‹‹F1. Initial Estimation›››

F2. Weights Analysis

F3. FileViewer

F4. Exit to DOS

Select number or use arrows

Then hit ‹ENTER›

Figure 2

1.4 Preliminary Design

The preliminary design function of PDWAP provides you with a way to make a complete weights analysis of a vehicle when only a little information is known. The program makes certain assumptions based on known vehicles and then sizes a Generic Design Vehicle (GDV) to match the requirements specified by the user. From this sized vehicle all the geometry needed to perform the weights analysis is written to a data file. The data file can then be adjusted by the user to make it conform to the vehicle in mind.

1.4.1 Description

To perform the initial weight estimation, simply select that choice from the menu. This is done by either using the arrow keys to place the highlite marks around it or by hitting function key <F1>. Then, hit the <ENTER> key. The program will go to the Preliminary Design routine and the title screen will appear.

The computer will then prompt you for the name of the initialization file you want it to use. PD.DAT is the default data file and you should just hit <RETURN> to use it. If you have a special file you wish to use, or if PD.DAT is on a directory other then the current one you will have to enter the complete file name, including path, if any, at the prompt.

While the data file is loaded in, a message will be displayed on the screen to "please wait." Once this is done, the program will ask for several inputs. The information that is generated by the program, based on these inputs will then be stored. At that point, the program will ask if there is another estimate to be done, if not, it will return to the main menu.

1.4.2 User Inputs

To perform the initial estimation, the program will obviously need some information from you. Since the aim of this program was to provide as much information as possible with the smallest amount of inputs, the amount of inputs will be kept to a minimum.

First, the program asks the name of the output file to use. The extension DAT will automatically be added on to the name, or you can specify any file name you want. If the file is to be written to a different directory or disk this should be included as part of the file name. At the prompt, enter the path and filename and hit the <RETURN> key.

The main factor in determining the size, and in turn weight, of the vehicle is the fuel storage. Therefore, the first input, and the one with largest impact is the amount of fuel to be stored. At the prompt, input the number of pounds of fuel that will be on board the vehicle and hit <RETURN>.

Since it's possible to use different fuels, PDWAP needs to know the density of the fuel you are working with. Since the fuel normally used in this type of vehicle is liquid hydrogen, it is the default, just hit <RETURN> to enter it. If you would like to use a different fuel, enter the fuel's weight density in pounds per cubic foot.

The next input is the oxygen to fuel ratio (f) for the vehicle. This term is the ratio vehicle. This term is the ratio for the whole vehicle. Therefore, although the rocket engines might have f=6 if there is an airbreathing system as well, then the hydrogen to be used for them must be taken into account as well. This will lower the f for the vehicle. The amount of oxygen needed is then calculated from the value input for f and helps to size the vehicle. Input the value for f and then hit <RETURN>.

Another factor in the weight of the vehicle is the number of crewmembers on board. This number is used to determine the weight of accommodations and supplies needed on board the vehicle. Input the number of crewmembers and passengers and hit <RETURN>.

Next, you must indicate the number of each type of engine on board the vehicle. If none of a particular type is used, then you should enter zero (0). If you specify a non-zero number of engines, then the program asks for the thrust in pounds that the engine produces. The amount of thrust is for one engine only, not for the entire number of engines of that type.

The program then asks whether the vehicle will take off vertically or horizontally. This is used to determine coefficients for the landing gear, as well as whether the control surfaces are sized for takeoff or landing weight. The initial data file already contains the information for a horizontal takeoff, so if that is selected, the information is just transferred to the output file. Otherwise, the program makes several modifications, including using gimbaled engines, separation systems, etc. Indicate whether the vehicle will be using a vertical or horizontal takeoff with a V or H, then hit <RETURN>.

The next item to be input is the amount of payload to be carried. This is used only to add to the weight of the vehicle and is not used to size the GDV. Enter the weight of the payload in pounds and then press <RETURN>.

To better identify the vehicle's weight statement PDWAP needs to know the vehicle's name. Enter the name of the vehicle at the prompt and hit <RETURN>. If you hit <RETURN> without entering a name, the name "No Name" is added to the file. The last input is a description of the vehicle. If you hit <RETURN> without entering a description the program will enter "NONE" in the data file. You can enter any description up to 65 characters in length.

The program will now build the output file and save it to disk. This output file will then become the input file for the next phase in the weights analysis. Once this is done the program will ask if another preliminary design is to be done. Select Y or N, hit <RETURN> and the program will continue with your selection.

If you've followed this so far your screen should look like Figure 3.

1.4.3 Calculations & Assumptions

The calculations involved in the preliminary design phase of PDWAP center around finding the right size vehicle in response to your inputs. The main factor is the weight of the liquid hydrogen and oxygen on board. This determines the volume required for the fuel tanks, which in turn defines the size of the vehicle. The GDV used as the model vehicle, has its shape defined by several equations using certain ratios, based on the volume of the propellant. Once the volume required has been calculated, the program then computes the various lengths, areas, and volumes needed to define the scaled GDV. These values are then written to the data file.

Preliminary Design Screen

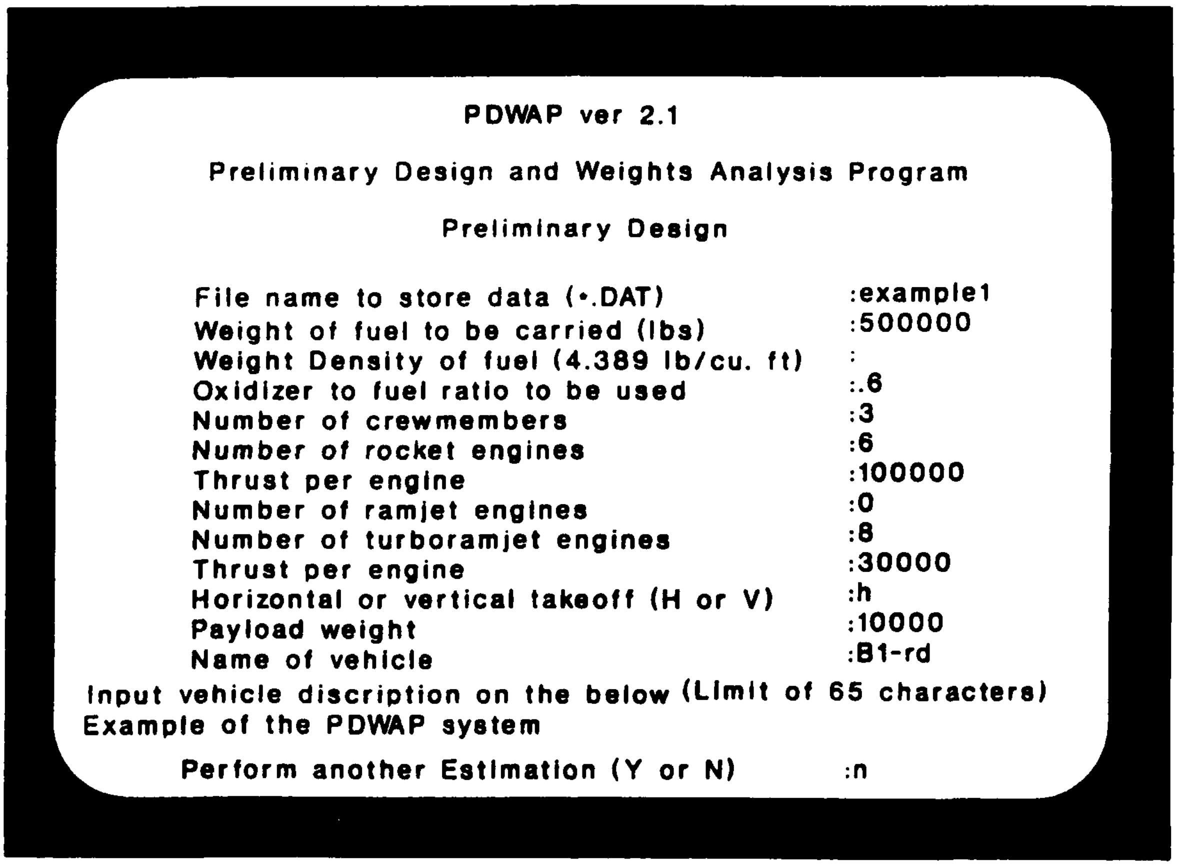


Figure 3

Generic Design Vehicle (GDV)

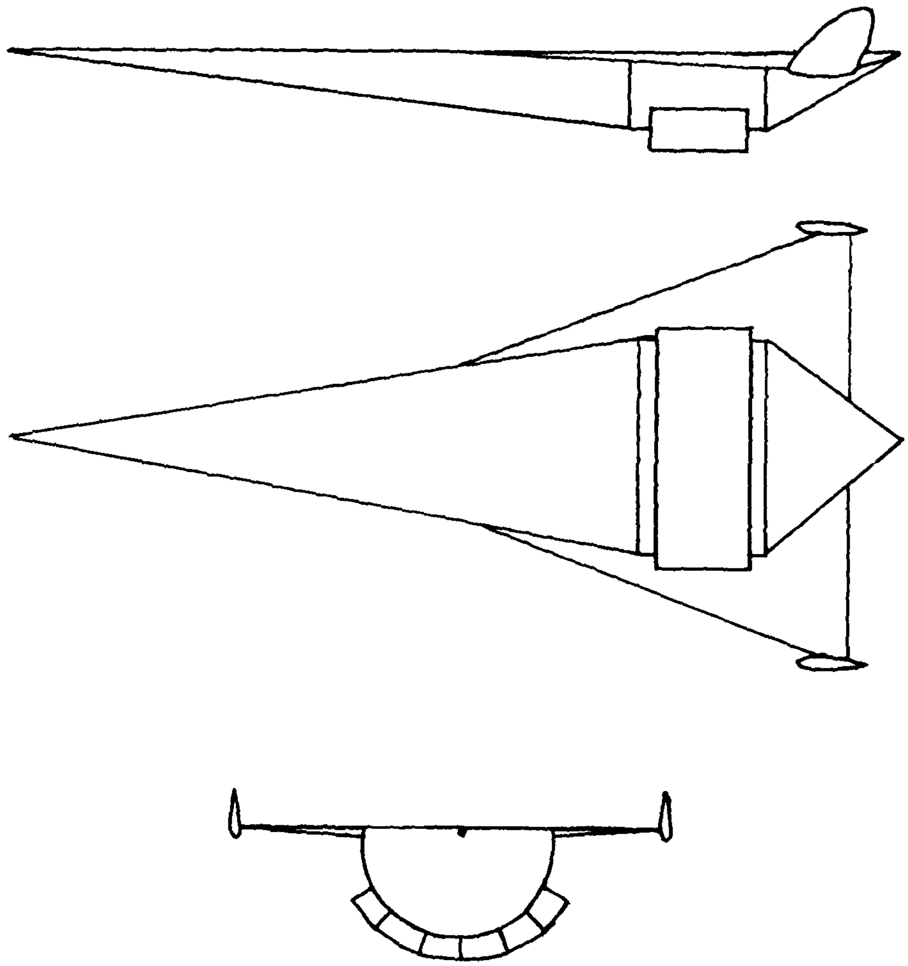


Figure 4

Since the GDV is only a basic model reflecting a single stage to orbit vehicle, the data generated may require some modification to suit the vehicle under study. It will, however, provide a useful starting point when little information is known about the vehicle.

The calculations and geometry used in the Preliminary Design routine are covered in greater detail in the Technical Reference Manual (§ 2.3.3).

In the process of coming up with the GDV quite a few assumptions had to be made. First, the general layout of the vehicle was based on what seems to be the current trend of designs in single stage to orbit vehicles (See Figure 4). The specific details of the design, and the reasons behind them, are discussed in the section covering the GDV in the Technical Reference Manual (§ 2.3.3).

The main assumption that would affect the user is that the preliminary design routine bases vehicle size on internal storage of fuel. In a vehicle such as the space shuttle, where its size is defined by the cargo and crew areas, this assumption would give poor results. In a case such as this, the user would have to calculate the required parameters, build the data file, and run the weights analysis routine.

1.5 Weights Analysis

The weights analysis portion of the program is based on the algorithm in Weights Analysis for Advanced Transportation Systems (WAATS) by C. L. Glatts, NASA CR-2024. You are referred to this source for a more complete discussion of the algorithm, equations, and the coefficients used. The algorithm was modified from WAATS in several places.

The first major modification was that WAATS allowed only one type of engine at a time. This was not practical for the type of vehicles that could be studied. For example, a vehicle might use rockets and turboramjets for take off, cruise and climb with turboramjets only, and then use rockets only for escape to orbit. This could not be handled by the original WAATS code.

1.5.1 Description

This portion of the program uses a data file, either generated by the preliminary design program or by the user, to perform a detailed weights analysis.

The program reads in the data file specified by the user and proceeds to make the calculations to find the weights of the various components of the vehicle. The components are then summed to find the landing and take-off weights. The take-off weight is then checked against the initial estimate and if it is within the tolerance the solutions are output. Since many of the component weights are determined by the take-off or landing weight, the program usually iterates several times to find the correct weight that will satisfy the system of equations. Usually this iterative procedure converges on the answer in a short period of time and is not noticeable.

1.5.2 User Input

Since the data necessary to run the program is in the data file, the program does not need any user input except for the name and location of the file. The extension DAT is used as the default for the input filename. If the data file is stored under another name, or if it is not located in the current path, input the filename, including drive or directory name if needed. The program will then ask for the name of the output file. Again, a default is used for the filename. If you hit <RETURN> the output file will be sent to the old filename but with the extension OUT added. Otherwise, you can enter any file name and extension you like. Your screen should look like Figure 5.

The program will calculate the weights and the output the results as directed. Once it is done the program will ask if there is another evaluation to be performed. If so enter Y and the program will continue. If not enter N and the program will return to the Main Menu.

1.5.3 Program Output

The output from the weights analysis routine consists of two parts. The first is a vehicle summary which lists the values having the most impact on a vehicle design, such as geometry, engine data, and the standard ratios used to compare vehicles. The second part is a detailed weight statement which lists all the system and subsystem weights. A sample weights statement from PDWAP is shown in Figures 6 and 7.

1.5.3.1 Vehicle Summary

The top part of the PDWAP output is the vehicle summary. It contains the vehicle name, a short description, as well as information about the data and output files. This information was included to help keep straight what output was for which vehicle.

The next section is the meat of the vehicle summary. It contains the geometrical values which define the shape and size of the vehicle. All values are in feet for lengths and square feet for areas. This section provides a good check on the sizing of the vehicle if it was produced with the preliminary design routine. In some cases it might require some modification to better suit the actual vehicle's constraints.

Next, the engine data is shown. This section indicates the number of each engine type. Below the engine type and number the maximum thrust for each engine is printed. The values have the units of pounds.

Another item in the summary is the fuel information. The amount of fuel stored aboard the vehicle is displayed in pounds. Also the fuel's weight density is shown. The density used in this program is pounds per cubic foot. The amount of liquid oxygen stored, if any, is also given.

The next section gives a brief look the weights of the vehicle. This includes Gross Take Off Weight (GTOW), weight of payload carried, dry, landing, and entry or insertion weights. These are all expressed in pounds.

Finally, the standard ratios used to compare vehicles are printed. Included are wing loading (W/S), Aspect ratio, thrust to weight ratio, and thrust loading. These were output so it would give the you an indication of how your vehicle measures up to the usual rules of thumb.

1.5.3.2 Weight Statement

The bulk of the output from the weights analysis routine is the vehicles's weight statement. This statement lists the weights for all of the systems and subsystems on the vehicle. The format used for the weight statement provides you with a detailed look at the vehicle.

Weights Analysis Screen

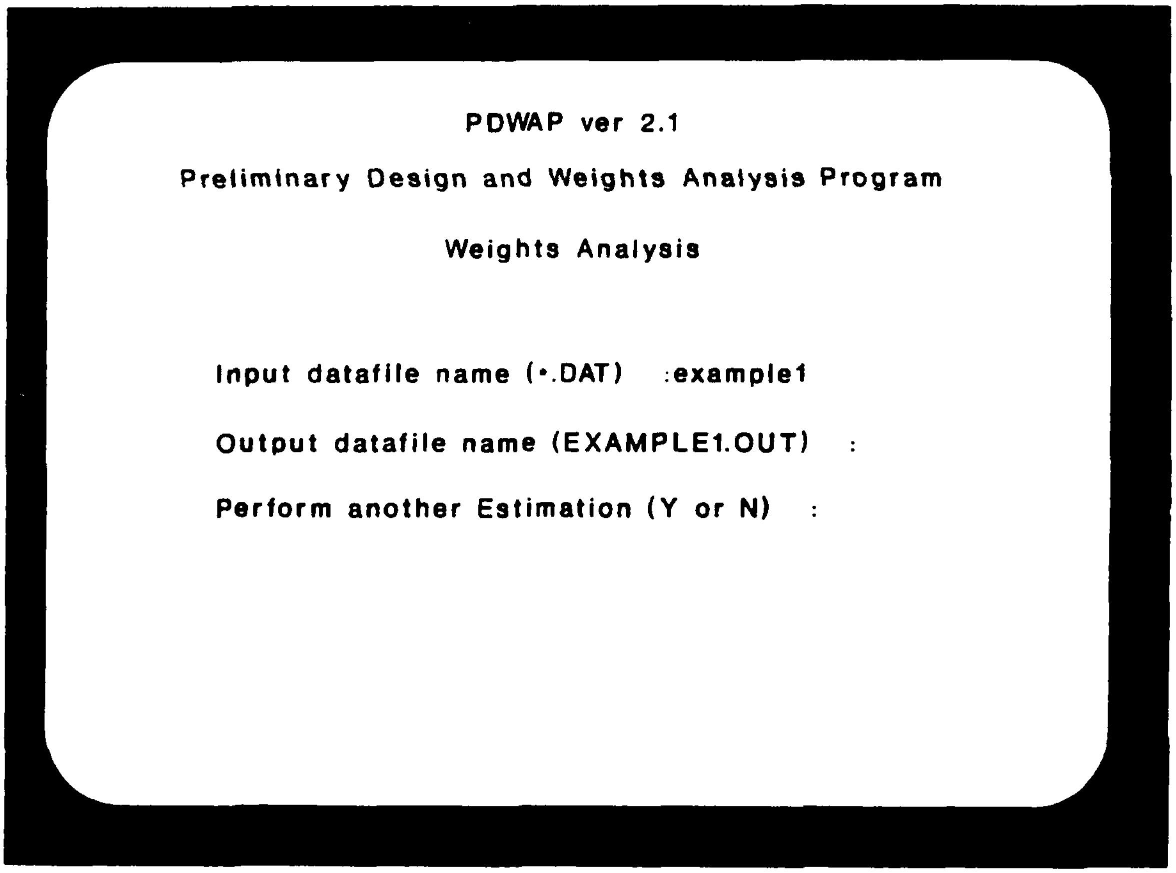


Figure 5

PDWAP Output Page 1

```
                              Vehicle Summary

                         Vehicle Name: Test1
Vehicle Discription: This vehicle is the SSTO demo for ETO

Data File = TEST1.DAT                          Output File = TEST1.OUT
     Date =  04-27-1989                               Date = 04-27-1989
     Time =  10:45:59                                 Time = 10:46:13

Length =       129   Height =       14   Span  =       57   TRoot =     9.2
 SBody =     4,003    SWing =    1,633   SVert =      163   SHorz =       0

 Number of: RamJets =  6         Rockets =  0        TurboRamJets =  4
 Thrust of:         =   10,000          =        0              =    15,000

Fuel =      60,000          Fuel Density =   4.389          LOX =          0

Weights:   GTOW =     133,010      Payload =          0    Dry =     71,305
        Landing =      72,290        Entry =     72,770

Ratios:  GTOW/S =    81.47     AR =  2.017    T/GTOW =  0.90    T/S =  73.50

                              Weight Statement
===========================================================================
   Group 1:   Aero surfaces                                     3,458
                       Wing                          2,166
                       Vertical                      1,291
                       Horizontal                        0
                       Fairing                           0

   Group 2:   Body structure                                   28,425
                       Basic body                   16,245
                       Secondary                     3,922
                       Thrust                          819
                       Integral fuel tanks           7,438
                       Integral Ox tanks                 0

   Group 3:   Thermal Protection System                         7,164
                       Vehicle insulation            7,164
                       Cover panels                      0

   Group 4:   Launch and Recovery  Gear                         4,004
                       Launch gear                     333
                       Landing gear                  3,672

   Group 5:   Propulsion                                       19,713
                       Rocket engines                    0
                       Ramjets                         840
                       Turboramjet                  10,890
                       Nonstructural fuel tank           0
                       Nonstructural Ox tank             0
                       Fuel tank insulation            497
                       Ox tank insulation               87
                       Fuel system                     572
                       Oxidizer system                 286
                       Pressurization system         6,152
                       Inlets                          350
```

Figure 6

PDWAP Output Page 2

Weight statement for:Test1 — Page 2

Group 6:	Orientation Control System		1,881
	Engine gimbal system	0	
	Attitude control system	199	
	Aerodynamic controls	1,282	
	Seperation system	399	
	ACS tankage	0	
Group 8:	Power supply		1,489
	Electrical System	1,435	
	Hydraulic/Pneumatic Sys	54	
Group 10:	Avionics		4,696
Group 14:	Crew Provisions		475
	Vehicle Dry Weight		71,305
Group 17:	Crew		535
Group 18:	Payload		0
Group 21:	Residual Propellant		450
	Trapped fuel	450	
	Trapped Oxidizer	0	
	Landing Weight		72,290
Group 22:	Reserve Propellants		480
	Fuel	480	
	Oxidizer	0	
	ACS fuel	0	
	ACS oxidizer	0	
	Entry Weight		72,770
Group 23:	Inflight Losses		240
	Fuel	240	
	Oxidizer	0	
Group 25:	Main Propellants		60,000
	Fuel	60,000	
	Oxidizer	0	
	Gross Weight		133,010

Figure 7

The general style uses the format outlined in MIL-M-38310B (USAF), Mass Properties Control Requirements for Missiles and Space Vehicles. This was done to standardize the output. The subcategories under each group are provided to give more detail to the sub systems, although different from the MIL Spec, they more closely match the components of the vehicle.

The entries labeled as "Gross Weight", "Entry Weight", etc., are the summation of the weights printed above them. They were placed in their positions so that you would have a feel about which groups or subsystems were included in which weight.

It must be stressed that the results of the weights analysis routine are only approximations based on your inputs. While exceptionally close results (Errors less then 0.1%) have been obtained, this is not always the case. All results must be taken with some caution, as with any analytic approximations.

1.6 File Viewer

1.6.1 Description

What started out as a rather crude file viewing routine has turned out to be a catch-all for PDWAP. The first thing to notice about File Viewer is that it is menu driven, using the same system as the PDWAP Main Menu. The screen should look like Figure 7.

The choices from this menu are:

1) File Viewer
2) File Converter
3) DOS Shell
4) Return to Main Menu

Each of the choices will be covered in detail below. Since number four is self explanatory we'll leave it out of the following descriptions.

1.6.2 File Viewer

As mentioned above File Viewer started out as a rather crude file display routine. So crude in fact, that one co-worker refused to use it. Realizing that there were better methods available, File Viewer started its change.

File Viewer is the only part of PDWAP that is not loaded in at run time. It is activated by using a DOS shell and then accessing another program that performs the file viewing process. When the program is finished it exits the DOS shell. This returns you to the PDWAP environment.

To use File Viewer, select it from the menu and hit <RETURN>. At the prompt enter the file name of the file you wish to see. Include any drive or directory prefixes as needed. And then follow the directions in the File Viewer program.

File Viewer Menu Screen

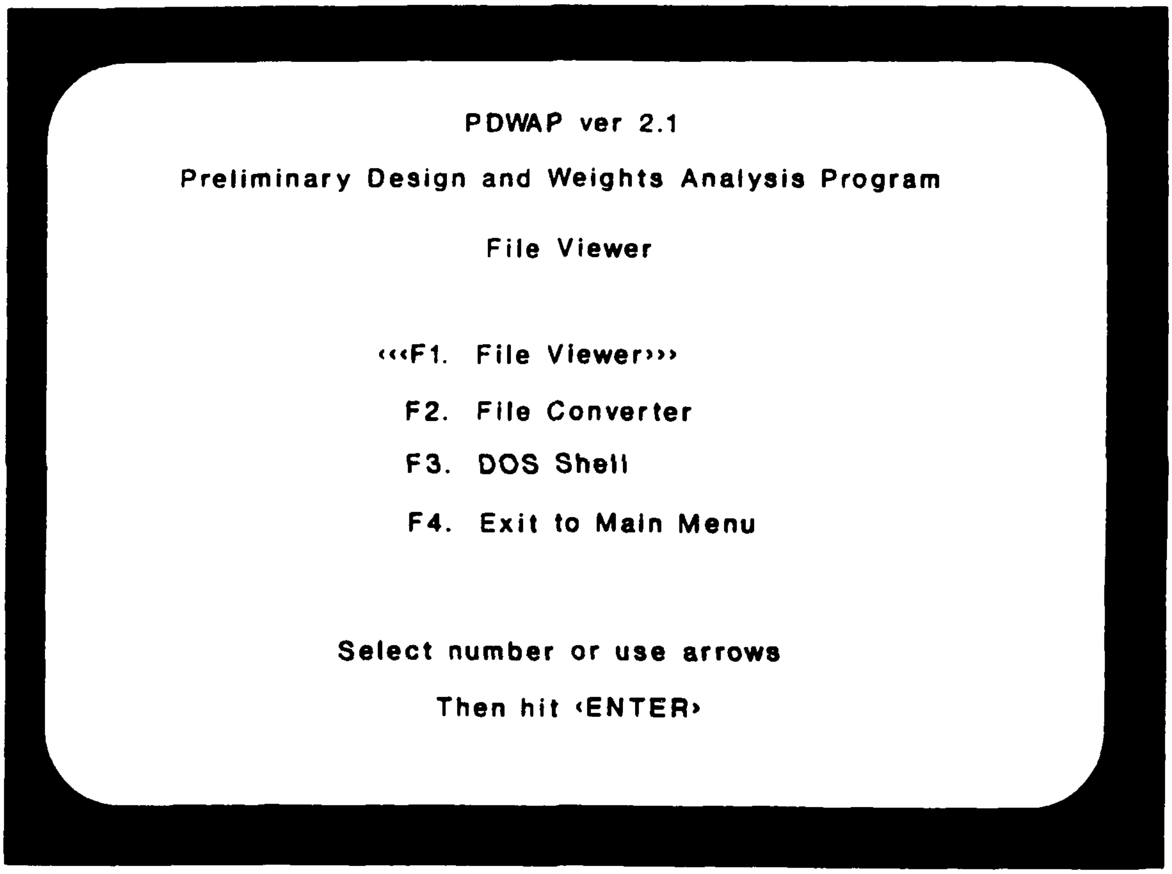


Figure 8

1.6.3 File Converter

As you can tell from the program version number, there has been more then one version of PDWAP. Since programs must change in order to maintain there usefulness, there probably will be more versions in time. This creates a potential problem. If the program is to change and grow, its data structure could possibly change as well. If this should happen what do you do with old, but still needed data files.

That was what happened with the change from version 1.0 to version 2.0. Unfortunately those files had to be either manually rewritten or discarded. When work was started on version 2.1 this was not going to happen again. A File Converter was added. This actually involves only a few minor corrections, but if not done the program won't work at all.

The File Converter changes files from the version 2.0 format to the 2.1 format. To use the File Converter just select it from the menu and hit <RETURN>. This will activate the File Converter.

At the prompt enter the file name you wish to convert. The program will then ask for the new file name you wish to use. The old file name will be the default. This will replace the old file with the new formatted file. If you would like to keep the old file unchanged enter a different file name. After it is finished the program will return you to the File Viewer Menu.

1.6.4 DOS Shell

Since the File Viewer worked through a DOS shell it seemed only logical to include it as on of the options in the program. This idea stemmed from the way we used PDWAP ver 2.0 in this office. We would run a vehicle through PDWAP, then fly it in a trajectory program, then back to PDWAP for changes, etc. While going through these iterations it would have been nice to have one of the programs create a DOS shell so the other one could be run. Also while editing the data file, which is covered later, it would be much simpler to be able to leave PDWAP active and invoke a text editor. This is why the DOS shell was added to PDWAP.

The DOS shell allows you to temporarily leave PDWAP and enter a second level of DOS. From here you can perform any of the normal DOS functions like checking directories, formatting disks, etc. After you are done enter EXIT at the DOS prompt. This will return you to the File Viewer Menu in PDWAP.

One word of caution, if you change directories during the shell you should change back to the one in which PDWAP is in before exiting the DOS shell.

1.7 The PDWAP Data File System

Obviously a program such as PDWAP needs some way of storing values, especially when it is comprised of several routines linked together put acting separately. For example, the preliminary design phase does not communicate directly with the weights analysis routine. Instead all values are stored in a data file which is open and read by the next program.

The data file also had to be easily read, understood, and modified by a human. This meant that a simple and clear method had to be used to represent the data. Preferably some system that was self documenting and understandable by someone with little extra training.

This explains the reasoning behind the PDWAP data file system. It allows you to read through the file and edit it without the cryptic and often confusing systems normally used for data files. The disadvantage to this system is that the file is rather large, about 13K. This disadvantage was felt to be acceptable given the ease of use of the file's structure.

1.7.1 File Description

The PDWAP data file consists of two main sections the header, and the body. The header contains information which helps to identify the vehicle being studied. This includes the vehicle's and file's name, date and time the file was prepared, and a description of the vehicle.

Next, the body contains all the numeric information which is used to describe the vehicle to the weights analysis routine. The body is divided into three columns. In the first are the values themselves. Second, is a description of what the value represents and what units it should be in. The last column contains the variable name that is used in the program. This is included to help someone modify the source code. Each column is defined by a label at the top of the body to further clarify their roles.

The body is further divided into two halves. The top half has, with a couple of exceptions, all the values representing the vehicle. These include the geometry, powerplants, capacities, etc. The top half, therefore, contains most of the values that may need checking or modification.

The second half is made up of the adjustment coefficients. There are 134 adjustment coefficients, each designated with A and their identification number in parenthesis: ie. A(102). These are used to adjust the historical data base that the weights analysis is based on.

They represent various variables in the equations which approximate the weight of your vehicle. For more information on their use see the Technical Manual (§ 2.5)

1.7.2 Input File for Preliminary Design

The input file for the Preliminary Design routine is basically a template. The routine reads it in makes the required changes based on the information that you input, and writes it to another file. This technique, while cumbersome allows you to use another data file as the template. This can be useful when the historical data stored in the initial data file doesn't match the vehicles you are studying. If the values had been coded into the program then the source code would have to be modified in order to change the historical base.

1.7.3 Output File from Preliminary Design

As mention above, the output is actually a copy of the initial data file with the required changes made to match your input. This file will then be used by the weights analysis routine to produce the weight statement.

1.7.4 Modifying the Data File

The main reason for modifying the data file is that it does not adequately reflect the vehicle you are try to simulate. This usually means that only the vehicle defining variables, located at the top half of the data file, will need changing. If more serious changes are needed, such as changes to the adjustment coefficients then you should refer to the Technical Reference Manual (§ 2.5).

The data file was constructed so that making changes would be easy. All that you need to do is load the file into a text editor that will support ASCII files. The easiest way to tell if your editor will work with ASCII files is to use the DOS TYPE command on a file you have written with it. If the screen looks like what you had in your file then you are OK. If it looks like garbage and beeps at you it is not an ASCII file.

To change any of the values put your cursor over the value. With your editor in an OVER-TYPE mode enter the new value. This should replace the old value with the new. If your editor does not support OVERTYPE then you will have to delete some characters from the old value so the file lines up properly.

Now save the file and you are ready to continue on with the weights analysis of your new vehicle.

This process can be done quickly by using the DOS Shell function of the File Viewer Menu. This allows you to exit PDWAP, edit and save your file, and then return to PDWAP for the next analysis.

1.7.5 Creating a Special File

Some vehicles are just not compatible with the GDV used in the Preliminary Design routine, one notable example is the Space Shuttle. This does not mean that you cannot perform a weights analysis on it. You must just manually do what the Preliminary Design routine does.

This involves finding all of the geometric and other parameters used to define a vehicle. Most of this information can be obtained directly from the vehicle's documentation, such as wing area, span, overall height, etc. Other values require more work. Fuel tank surface area is probably the worst case, but with a few assumptions even this can be solved.

Once you have all of the necessary values, you can use an existing data file as a template to write your own special file. As mentioned above, the editor you use must be able to save the data file in an ACSII format.

1.8 A Sample Case: The IGV

The IGV, or Incremental Growth Vehicle, was a study vehicle developed by McDonnell Douglas to test high Mach (6 - 9) engines and materials. Because of its available design data it can be used to show PDWAP's ability to perform a sizing and weights analysis on a documented vehicle.

The IGV is a single seat vehicle designed for a air launch from a B-52 mothership. It used rocket power to boost up to it's test speed and altitude and then used whatever airbreathing engine was installed for testing.

We will now go through the steps nessecary to do a first cut analysis of the IGV using only rudimentary data. It is assumed that PDWAP is running and you are seeing the Main Menu.

Step	Action
1	Select 'Initial Estimation' from the menu.
2	Hit <ENTER> to accept PD.DAT as the initialization file.
3	Enter the filename, IGV.
4	Enter the weight of fuel onboard, 1875.3 lbs.
5	Hit <ENTER> to accept the density for LH_2.
6	Enter oxidizer to fuel ratio, 6.
7	Enter the number of crewmembers, 1.
8	Enter the number of rockets, 2.
9	Enter the thrust of each engine, 17000.
10	Enter 0 for the both the number of ramjets and turboramjets.
11	Since this is an air launch, enter V so the vehicle will be sized for landing.
12	Enter the payload weight, 1500.
13	Enter the vehicle's name, IGV.
14	Enter a description of the vehicle.
15	Enter N for 'Another Estimation? (Y/N)'.

You should now be back at the Main Menu. The vehicle has been defined and that information is stored in the file IVG.DAT. That information will be used in the next step, performing the weights analysis

16	Select 'Weights Analysis' form the main menu.
17	Enter 'IGV' at the prompt for the input file name.
18	Hit <ENTER> to accept the output filename.
19	Enter N for 'Another Estimation? (Y/N)'.

You should once again be back at the Main Menu. The complete vehicle analysis is no done. All that remains is to view the output and check it against the known vehicle. Of course, if you were doing this on your own vehicle you wouldn't have the ability to check answers. This is being done to show the capability of PDWAP to produce reliable results.

20	Select 'View Data File'.

21 Select 'File Viewer'.

22 Enter 'IGV.OUT'.

The output file should now be displayed. You can see the entire file by hitting the <PgUp> and <PgDn> keys or using the arrows. The following table gives some of the important values calculated by PDWAP and as given in the report on the IGV, and also the percent error that is present in the calculations.

Description	PDWAP	Documented	% Error
GTOW	28,931	27,463	5.35
Entry	15,751	14,336	9.87
Landing	15,646	14,011	11.67
Dry	13,763	11,961	15.07

It is noteworthy that this vehicle was selected at random and was not part of the test suite of vehicles used to test and verify PDWAP. The program was in no way groomed to work with this particular vehicle. Also, I should point out the if you were following along in this example please check the vehicle size values. The are not in as good an agreement as the weights. The lengths are nearly identical, 47 ft for PDWAP versus 45.25 for the vehicle, however the other values don't agree well. This is because of the assumptions made by PDWAP about the vehicle. It is set up to define the vehicle as a wave rider type with more of a blended body. The IGV is more of a conventional aircraft structure, therefore, the values for height, width, etc. tend to be off by several feet.

With some grooming of the data file a better representation of the vehicle you are working on can be obtained. For example, wing areas, fuel tanks etc. can all be manually defined in the data file. The adjustment coefficients can all be adjusted to better fit the vehicle type you are trying to represent. These changes, while being simple in themselves would take too much time and space to explain here, plus they vary on a case by case basis. The coefficients used in the initialization file PD.DAT have been found to give the best average results for the types of high speed vehicles currently of interest.

2 Technical Reference Manual

2.0 How To Use This Manual

The technical reference manual for PDWAP is broken down into sections based on the individual subroutines. These sections are listed in the same order as the subroutines appear in the source code. It is felt that this system of dividing the technical manual aids the person who needs it the most, the one who is trying to modify or just follow the code as it is in the source listing. One can follow the source code and the manual in the same way.

The sections for each subroutine are further divided into three sections. The first describes the purpose of the subroutine. Basically this covers the 'Why' of the subroutine. Knowing this up front it should be easier to describe the operation of the subroutine, and will hopefully help the reader understand the methods and techniques used.

The next section describes the tables and/or data used by the subroutine. This should give you some feel for what data is needed by the subroutine and how it will be modified.

The last section about the subroutine is perhaps the most revealing, the processing performed. Here the subroutine is described by the way it operates - basically the algorithm used is explained and any mathematics used are covered. These sections were written not to describe the code line-by-line, but to describe the process the code attempts to perform. To the person with a background in other languages, or no language at all this will provide a much better understanding of the code. To the programmer who can understand the TurboBASIC language, this section will perhaps provide the answer to any questions looking at the code will bring up, after all no two programmers will code the same algorithm in the same way.

This section, as well as this entire report should help clear up any questions, and help get you running and using PDWAP.

2.1 MAIN

2.1.1 Purpose

The MAIN routine provides the central control of the program. It is used to start the program and display the main menu. Based on the user's choice it then invokes the appropriate routine.

The MAIN routine uses the MenuDisplay (§ 2.13) subroutine to display the main menu and get the user's choice. Other routines used by the MAIN routine are PD (§ 2.2), used to perform a preliminary design, WAATSb (§ 2.4), for the weights analysis, and FileViewer (§ 2.9), to view a file and perform other file related functions.

2.1.2 Tables and Data Referenced

The MAIN routine assigns the strings needed by the MenuDisplay (§ 2.13) subroutine to the array sel$. These are then used to display the program's main menu.

2.1.3 Processing Performed

Outside of controlling the flow of the program, no processing is performed by the MAIN routine.

2.2 PD

2.2.1 Purpose

The only purpose of the routine PD is to provide the front end for the preliminary design part of PDWAP. It sets some constants which are needed by the routine PD1 (§ 2.3) and then turns over control to PD1 (§ 2.3).

This routine functions inside of a loop which is exited only after the user has selected that she does not wish to run the routine again.

2.2.2 Tables Referenced

None.

2.2.3 Processing Performed

None. Provides program control for the PD1 routine, as well as sets several constants before PD1 is called.

2.3 PD1

2.3.1 Purpose

PD1 is the core of the preliminary design part of PDWAP. It makes the calculations and assumptions that determine the size and relative shape of the Generic Design Vehicle (GDV) that is used as the basis for the weights analysis to follow.

This part of the code is used when the user does not have enough specific information on a vehicle to provide the data required by the weights analysis code WAATSb (§ 2.4). Provided with only some rudimentary information and some assumptions based on the current state of hypersonic vehicle design, it scales the GDV to match the user's inputs.

2.3.2 Tables Referenced

The routine PD1 uses the initialization file PD.DAT or the file specified by the user. This file is read in line by line and essentially copied to the user named output file. The changes that are made to the file during the copying process reflect the size and shape of the GDV specified by the user.

There are also several flags and constants that are changed based on whether the vehicle is using storable or cryogenic fuels and its take off mode, either horizontal or vertical.

The routine reads in the line as the value and the string containing the value's description. It will then change the value as needed and output the value and string to the output file.

2.3.3 Processing Performed

PD1 performs three types of processes when it is called. First, it queries the user with a general set of questions about the vehicle. Next, it goes through a set of mathematical steps defining the geometry of the vehicle. Last, it outputs a data file based on the initialization file, the user's inputs, and its calculations.

The following is the algorithm used to define the GDV. It contains all of the geometric assumptions and relations used in the process.

To start the process some constants must be assumed. These are defined in PD (§ 2.2).

Constants:

$\pi = 3.1415926$

$\rho_{H_2} = 4.389$ lbs/ft^3

$\sigma_f = 8°$

$\rho_{O_2} = 71.025$ lbs/ft^3

$\sigma_r = 30°$

$W_{TO} = 3{,}000{,}000$ lbs

$\lambda_{LE} = 82°$

Input from user:

W_{fu}

N_{trj}

ρ_{fu}

T_{re} (If $N_{re} \neq 0$)

f

T_{rj} (If $N_{rj} \neq 0$)

N_{crew}

T_{trj} (If $N_{trj} \neq 0$)

N_{re}

W_{pay}

N_{rj}

With these constants known some basic quantities can be calculated:

$$W_{O_2} = W_{fu} f \qquad \text{Eqn 2-1}$$

$$V_{FuTk} = \frac{W_{fu}}{\rho_{fu}} \qquad \text{Eqn 2-2}$$

$$V_{OxTk} = W_{O_2} \qquad \text{Eqn 2-3}$$

$$N_{Intake} = N_{rj} + N_{trj} \qquad \text{Eqn 2-4}$$

$$W_P = W_{O_2} + W_{fu} \qquad \text{Eqn 2-5}$$

$$AI_{capt} = 6\, N_{Intake} \qquad \text{Eqn 2-6}$$

$$T_{TOT} = N_{re}T_{re} + N_{rj}T_{rj} + N_{trj}T_{trj} \qquad \text{Eqn 2-7}$$

$$W_{Land} = W_{TO} - W_P \qquad \text{Eqn 2-8}$$

Since the vehicle has the following shape (see Figure 9) it can be evaluated rather easily. The trick is to get the equation for volume into a form which can be solved for one of the unknown dimensions.

To start with, the body's volume can be found by treating it as two half-cones attached to a half-cylinder. Since there is no fuel stored in either the midsection or forward cone they can be left out, for now, and use the fuel volume as the known quantity. This can be written mathematically as,

$$V_{fu} = (V_{fb} - V_{fc}) + V_{rb} \qquad \text{Eqn 2-9}$$

Using the half cone volume formula, where r is the radius of the cone's base, and h is the height of the cone:

$$V = \tfrac{1}{2} (\pi r^2 h/3)$$

and Eqn 2-9 yields:

$$V_{fu} = \tfrac{1}{2} \left[(\pi r_b^2 x_f/3 - \pi r_f^2 x_c/3) + \pi r_b^2 x_r \right]$$

$$= \frac{\pi}{6} (r_b^2 x_f - r_f^2 x_c + r_b^2 x_r) \qquad \text{Eqn 2-10}$$

Making some assumptions and observations, let:

$$x_c = \tfrac{1}{4} x_f \qquad \text{Eqn 2-11}$$

$$r_b = x_f \tan(\sigma_f) \qquad \text{Eqn 2-12}$$

$$r_f = x_c \tan(\sigma_f)$$

$$= \tfrac{1}{4} x_f \tan(\sigma_f) \qquad \text{Eqn 2-13}$$

$$x_r = r_b / (\tan(\sigma_r))$$

$$= x_f \tan(\sigma_f)/\tan(\sigma_r) \qquad \text{Eqn 2-14}$$

GDV - Dimensional Detail

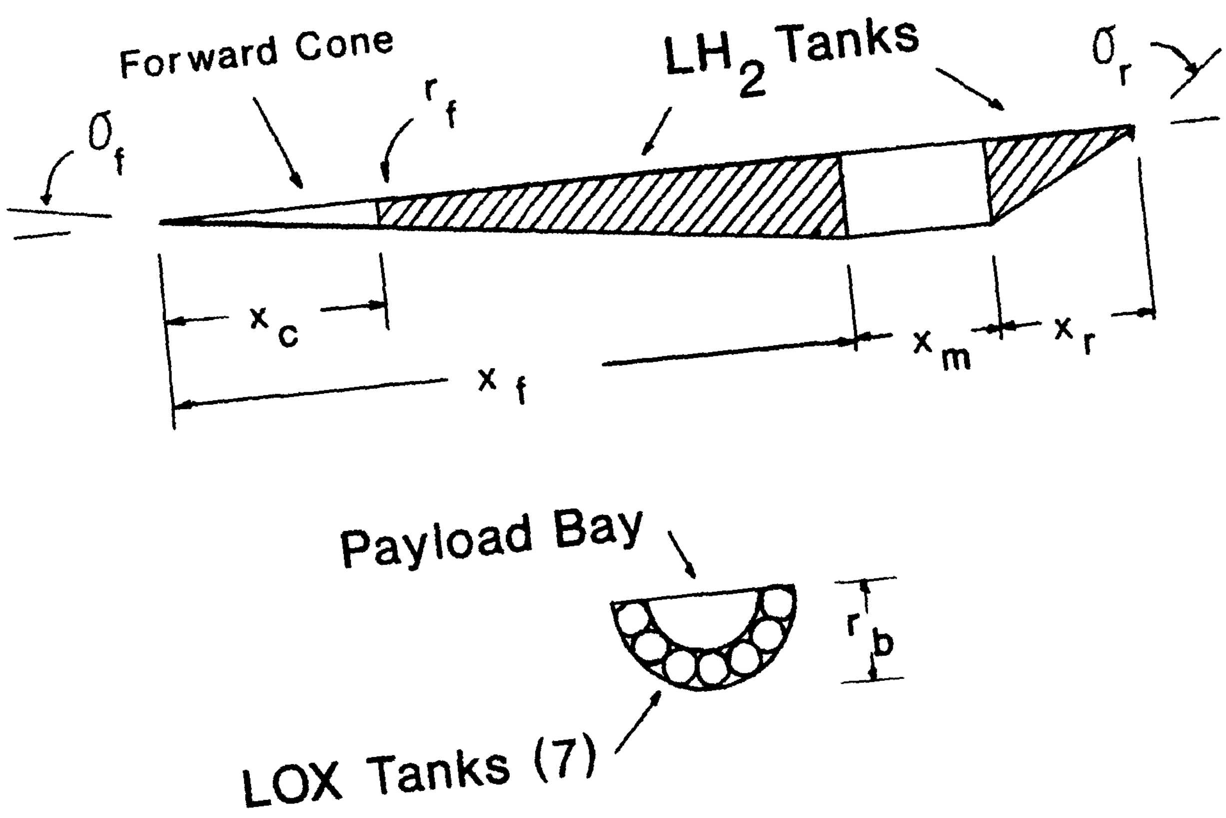


Figure 9

At this point it is obvious that x_f is the variable to solve for in the volume equation. Substituting Eqns 2-11 through 2-14 into 2-10 we have:

$$V_{fu} = \frac{\pi}{6}\left[[x_f \tan(\sigma_f)]^2 x_f - [¼ x_f \tan(\sigma_f)]^2 (¼ x_f) + [x_f \tan(\sigma_f)]^2 x_f \frac{\tan(\sigma_f)}{\tan(\sigma_r)} \right]$$

$$= \frac{\pi}{6}\left[x_f^3 \tan^2(\sigma_f) - \frac{1}{64} x_f^3 \tan^2(\sigma_f) + x_f^3 \frac{\tan^3(\sigma_f)}{\tan(\sigma_r)} \right]$$

$$= x_f^3 \frac{\pi}{6}\left[\frac{63}{64}\tan^2(\sigma_f) + \frac{\tan^3(\sigma_f)}{\tan(\sigma_r)} \right]$$

This can be neatly solved for x_f:

$$x_f = \left[V_{fu} \frac{6}{\pi} \left[\frac{63}{64}\tan^2(\sigma_f) + \frac{\tan^3(\sigma_f)}{\tan(\sigma_r)} \right]^{-1} \right]^{\frac{1}{3}} \quad \text{Eqn 2-15}$$

Now that x_f is known we can backsolve for x_c, r_b, r_f, and x_r using equations 2-11 through 2-14.

Since the two cone sections are defined all that is left is the mid-section. This can be solved by relating the length to the oxygen volume.

The vehicle is assumed to have the oxygen stored in seven cylinders with hemispherical ends. These tanks are located in a semicircle around the midsection (see Figure 10) Using:

$$V = ½ \left[\frac{4}{3} \pi r^3 \right] = \frac{2}{3} \pi r^3$$

for a hemisphere, and:

$$V = \pi r^2 \ell$$

for a cylinder, the oxygen volume can be defined as:

$$V_{OxTk} = 7 \left[\frac{2}{3} \pi r_t^3 + \pi r_t^2 (x_m - 2r_t) \right]$$

Simplifying:

$$V_{OxTk} = 7\pi\left[\frac{2}{3} r_t^3 + r_t^2 x_m - 2r_t^3 \right]$$

$$= 7\pi\left[r_t^2 x_m - - r_t^3 \right]$$

$$= 7\pi\left[\frac{r_b^2}{16} x_m - \frac{r_b^3}{48} \right]$$

and solving for x_m:

$$x_m = \left[\frac{V_{OxTk}}{7\pi} + \frac{r_b^3}{48} \right] \frac{16}{r_b^2} \qquad \text{Eqn 2-16}$$

Now that the major dimensions have been found some surface areas can be calculated. The surface of the oxygen tanks is found in the same manner as the volume; a cylinder with two hemispherical end caps.

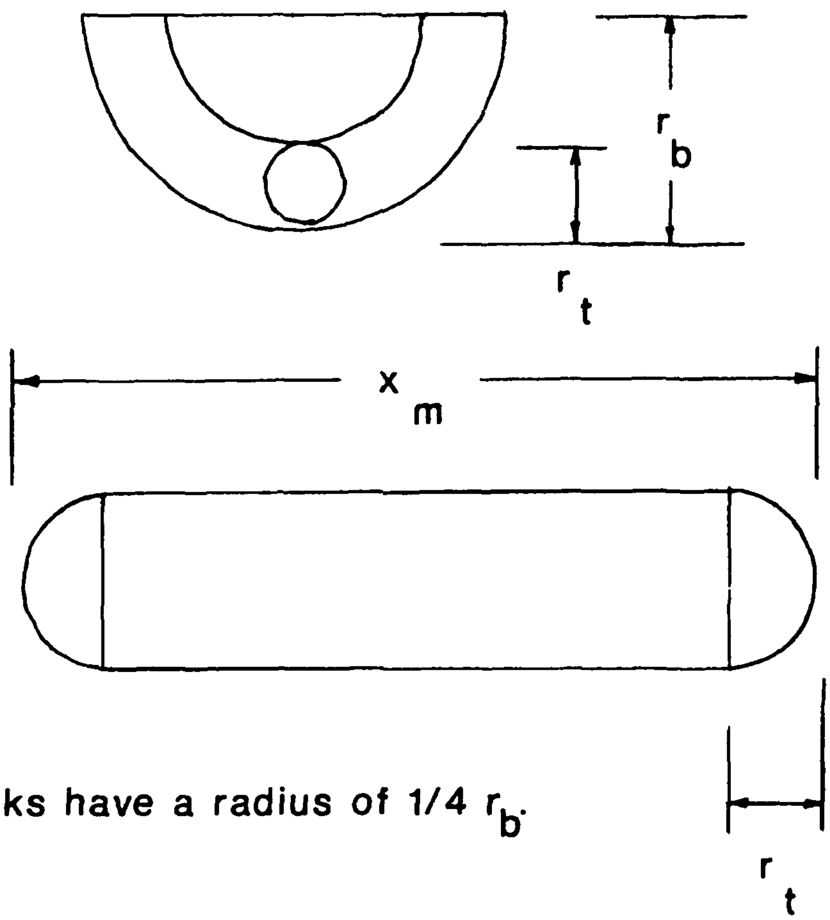
$$S_{OxTk} = 7[2S_{cap} + S_{cyl}]$$

$$= 7\left[2\left[½ 4\pi r_t^2\right] + 2\pi r_t\left[x_m - 2r_t\right]\right]$$

$$= 7\left[4\pi r_t^2 - 2\pi r_t^2 + 2\pi r_t x_m\right]$$

$$S_{OxTk} = 14\pi\left[r_t x_m + r_t^2\right] \qquad \text{Eqn 2-17}$$

GDV - LOX Tanks Detail



The tanks have a radius of 1/4 r_b.

Figure 10

The surface area of the fuel tanks can be found by treating the forward tank as half of a truncated right circular cone (See Figure 11). Therefore the area of the forward tank is the sum of the area of the half cone, the half discs where the cone was cut, the area of the top surface, minus the area of the forward half cone. This can be written as:

$$S_{Tk1} = \tfrac{1}{2}\left(\pi r_b \sqrt{r_b^2 + x_f^2} - \pi r_f \sqrt{r_f^2 + x_c^2}\right) + \tfrac{1}{2}(\pi r_b^2 + \pi r_f^2) + \tfrac{1}{2}(2 r_b x_f - 2 r_f x_c)$$

$$= \tfrac{1}{2}\pi\left(\left(\sqrt{r_b^2 + x_f^2} - r_f\sqrt{r_f^2 + x_c^2} + r_b^2 + r_f^2\right) + r_b x_f - r_f x_c \qquad \text{Eqn 2-18}$$

The second tank is easier. It's just the half-cone, triangle on top, and half-disc at the end.

$$S_{Tk2} = \tfrac{1}{2}\pi r_b \sqrt{r_b^2 + x_r^2} + \tfrac{1}{2}(2 r_b x_r) + \tfrac{1}{2}\pi r_b^2$$

$$= \tfrac{1}{2}\pi\left(r_b\sqrt{r_b^2 + x_r^2} + r_b^2\right) + x_r r_b \qquad \text{Eqn 2-19}$$

Now to finish the body geometry, the length of the vehicle is:

$$L_{Body} = x_f + x_m + x_r \qquad \text{Eqn 2-20}$$

and assuming:

$$x_w = 2/3\ L_{Body}$$

$$b = 4 r_b$$

$$h = r_b$$

The wetted area of the body is found by summing the components of the fuselage.

$$S_{Body} = \tfrac{1}{2}\left(\pi r_b \sqrt{r_b^2 + x_f^2} + \pi r_b \sqrt{r_b^2 + x_r^2}\right) + \tfrac{1}{2}(2\pi r_b x_m) + \tfrac{1}{2}(2 r_b) x_f + \tfrac{1}{2}(2 r_b) x_r + x_m(2 r_b)$$

$$= \tfrac{1}{2} r_b\left(\left(\sqrt{r_b^2 + x_f^2} + \sqrt{r_b^2 + x_r^2}\right) + r_b(\pi x_m + x_f + x_r + 2 x_m)$$

GDV - Hydrogen Tanks Detail

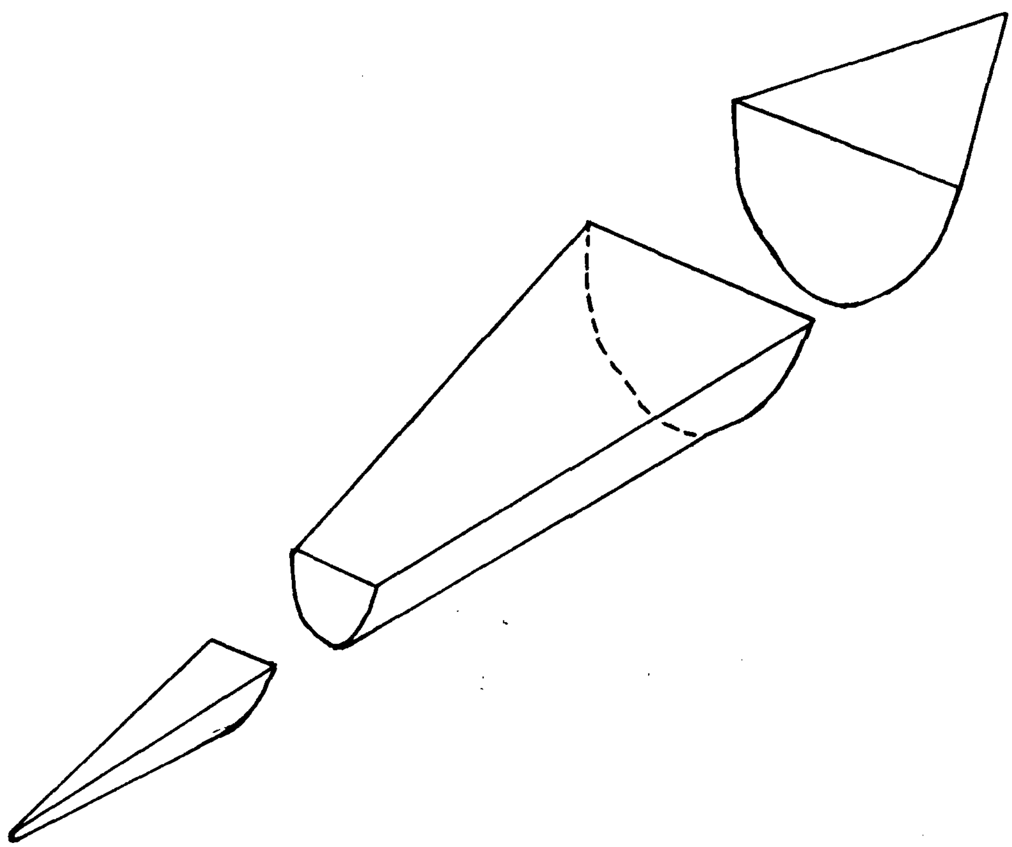


Figure 11

$$S_{Body} = r_b\left[\tfrac{1}{2}\pi\left(\sqrt{r_b^2 + x_f^2} + \sqrt{r_b^2 + x_r^2}\right) + x_f + x_r + (2 + \pi)x_m\right] \qquad \text{Eqn 2-21}$$

The wing for the GDV is basically a delta configuration and can be approximated by a triangle. The wing is assumed to have a length of two thirds of the fuselage. Also, the span is four times the body radius. This gives the vehicle a fineness ratio of two.

$$S_{Wing} = \tfrac{1}{2}x_w b \qquad \text{Eqn 2-22}$$

The area of the vertical fins is defined as ten percent of the wing area for stability reasons.

$$S_{Vert} = 0.1\ S_{Wing} \qquad \text{Eqn 2-23}$$

The structural span at half cord, shown in Figure 12, is given by:

$$b_{st} = \sqrt{(\tfrac{1}{2}x_w)^2 + (\tfrac{1}{2}b)^2} \qquad \text{Eqn 2-24}$$

The next value needed to describe the wing is the wing thickness at the root. For this it is assumed that the wing will be a thin airfoil with a thickness to cord ratio of nine percent, similar to a NACA 0009.

$$T_{Root} = 0.09\left(\tfrac{1}{2}b\ \tan(\lambda_{LE})\right) \qquad \text{Eqn 2-25}$$

The last value needed to define the vehicle for PDWAP is the thermal protection system's area. This area is made up of the lower surface of the fuselage and wing areas.

$$S_{TPS} = S_{fc} + S_{rc} + S_{cyl} + S_{Wing}$$

$$= \tfrac{1}{2}\left[\pi r_b\left(\sqrt{r_b^2 + x_f^2} + \sqrt{r_b^2 + x_r^2}\right)\right] + \tfrac{1}{2}(2\pi r_b x_m) + S_{Wing}$$

$$= \tfrac{1}{2}\pi r_b\left(\sqrt{r_b^2 + x_f^2} + \sqrt{r_b^2 + x_r^2} + 2x_m\right) + S_{Wing}$$

Eqn 2-25

At this point the preliminary design portion of PDWAP is complete. All of the values needed to describe the vehicle to WAATSb (§ 2.4) are known. This data is then written to file and can either be modified by the user to more closely match a desired vehicle, or used directly by WAATSb.

GDV - Wing Detail

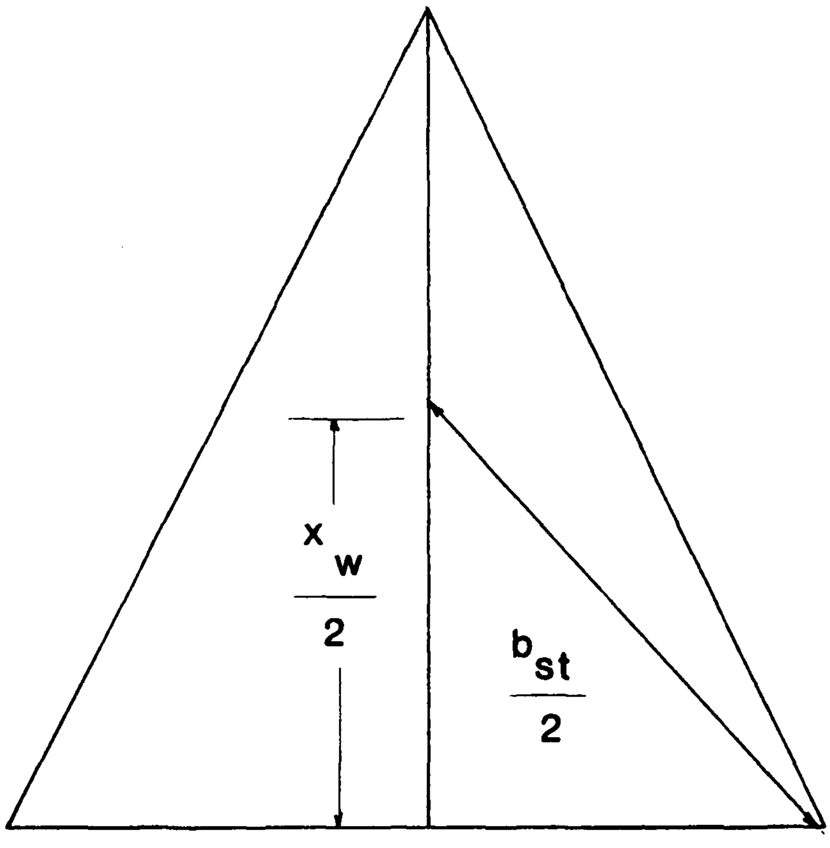


Figure 12

2.4 WAASTb

2.4.1 Purpose

The routine WAATSb is the front end to the subroutine CalcWts (§ 2.5). It gets the input and output data file names from the user and controls the execution of CalcWts.

2.4.2 Tables Referenced

None.

2.4.3 Processing Performed

The only processing that WAATSb performs is in the way of interfacing with the user.

2.5 CalcWts

2.5.1 Purpose

CalcWts is the core of the weights analysis part of PDWAP. It reads in the data from the user specified file and goes through a series of calculations based on the historical weights analysis technique.

NOTE: WAATS, the algorithm on which CalcWts and WAATSb are based, is clearly defined by C. R. Glatt in NASA CR-2420 . This system of calculations is basically a modified version of the original WAATS program. It has been intentionally kept in the same form as the original, at least as much as the two different languages would allow, so that the documentation for WAATS could be used to help describe how WAATSb (§ 2.4) and CalcWts function. With this in mind the user is strongly urged to obtain a copy of the WAATS documentation.

2.5.2 Tables Referenced

CalcWts uses the data file specified by the user and read in WAATSb (§ 2.4). When the weights analysis is complete the output is sent to a data file with the name specified by the user in WAATSb (§ 2.4).

2.5.3 Processing Performed

The first thing that CalcWts does is to calculate the total number of engines on the vehicle and the total thrust.

$$N_{Eng} = N_{re} + N_{rj} + N_{trj}$$

$$T_{TOT} = N_{re}T_{re} + N_{rj}T_{rj} + N_{trj}T_{trj}$$

The number of gallons of fuel on board is also calculated based on the fuel's volume in cubic feet.

$$Gal = V_{FuTk}\ (7.481\ gal/ft^3)$$

The program can now begin to calculate the vehicle's weight based on individual components. A problem arises in that most of the weight equations use either gross or empty weight as one of the parameters. This means that the solution will have to be an iterative one. However, there are still a few weight equations that can be solved directly, and the routine CalcWts will start with those.

The first to be solved is weights of the aerodynamic fairing and vertical surfaces. These will only be calculated if the shape flag IShape has been set to a non-zero value in the input data file. The equations for these two weights are:

$$W_{Fair} = A_8 S_{Fair} + A_9$$

$$W_{Vert} = A_4 S_{Vert}^{A_{89}} + A_5$$

The other aero-surfaces, such as the wings and horizontal surfaces are based either on take-off or landing weight, and are therefore calculated during the iterative part of the program.

The body structural weight can also be calculated now. This includes the weight of the basic body, secondary, and thrust structures. The remaining structural items, are the integral fuel and oxidizer tanks, if used, are calculated later in the program during the iterative loop.

$$W_{Basic} = A_{14} S_{Body} + A_{15} \left\{ \left[\frac{L_{Body} X_{lf}}{H_{Body}} \right]^{0.15} q_{max}^{.16} S_{Body}^{1.05} \right\}^{A_{81}} + A_{16}$$

$$W_{SecSt} = A_{17} S_{Body} + A_{18}$$

$$W_{ThrSt} = A_{19} (T_{rj} N_{rj} + T_{trj} N_{trj}) + A_{10} T_{re} N_{re} + A_{11} + A_{20}$$

The next set of weights to be calculated is the weight of the thermal protection system (TPS). The weight of the TPS is divided into two components, the insulation and the cover panels. The equation for each are:

$$W_{Insul} = A_{21} S_{TPS} + A_{76}$$

$$W_{Cover} = A_{22} S_{TPS} + A_{77}$$

And the total weight of the system is:

$$W_{TPS} = W_{Insul} + W_{Cover}$$

The next system to be analyzed is the main propulsion system. This is divided up into the following components:

Turboramjet
Ramjet
Rocket
Intake
Cryogenic fuel system
Storable fuel system

The turboramjet calculations start by finding the pressure ratio, P_r, and total pressure, P_{t0} at the design Mach number and altitude. These values are gotten from the subroutine Atmos (§ 2.8). The values returned are then used to find the total pressure in the intake, P_{t2}.

$$P_{t2} = P_r P_{t0}$$

Next, the air flow, W_a, must be calculated using:

$$W_a = W_{aRef} A_{ctr}$$

Now the weight of the engine mount is calculated:

$$W_{EngMt} = A_{102} T_{trj} + A_{103}$$

The total weight of all turboramjet engines, and their mounts on the vehicle is given by the relation:

$$W_{trj} = \left[A_{32} e^{(A_{33} W_a)} \frac{P_{t2} - P_{high}}{P_{low} - P_{high}} + A_{34} e^{(A_{35} W_a)} \frac{P_{t2} - P_{low}}{P_{high} - P_{low}} + W_{EngMt} \right] N_{trj}$$

Similarly, the weight of the ramjet mount and total system on board is given by:

$$W_{EngMt} = A_{102} T_{rj} + A_{103}$$

$$W_{rj} = (A_{82} T_{rj} + A_{83} + W_{EngMt}) N_{rj}$$

The last of the engine systems, the rocket, is calculated in the same manner.

$$W_{EngMt} = A_{12} T_{re} + A_{13}$$

$$W_{re} = A_{28} T_{re} + A_{29} T_{re} A_{Ratio}^{A_{30}} + A_{31} + W_{EngMt} N_{re}$$

The weight for the inlets is found using the following formulae:

$$W_{IDuct} = A_{53} \left[\sqrt{L_{Inlet} N_{Inlet}} \; (A_{ICapt} / N_{Inlet})^{.333} P_{t2}^{.667} G_{Fct} F_{Mach} \right]^{A_{54}} + A_{105}$$

$$W_{VRamp} = A_{106} \left[L_{Ramp} N_{Inlet} \sqrt{A_{ICapt} / N_{Inlet}} \; T_{Fct} \right]^{A_{107}} + A_{108}$$

$$W_{Spike} = A_{109} N_{Inlet}$$

$$W_{Inlet} = W_{IDuct} + W_{VRamp} + W_{Spike}$$

which is used only if the vehicle has airbreathing engines ($N_{trj} \neq 0$ or $N_{rj} \neq 0$).

The last of the main propulsion system to be calculated is the fuel system. This is divided into two parts, one for cryogenic fuels (ie. H_2 & LOX) and one for storable fuels (ie. JP-4). To decide which is used a flag called ICry is used. If the flag is set to 1 in the data file then the vehicle is using cryogenic fuels, and the following calculations are made.

First the weight of the insulation of the fuel and oxidizer tanks are found:

$$W_{InsFt} = A_{40} S_{FuTk} + A_{41}$$

$$W_{InsOt} = A_{42} S_{OxTk} + A_{43}$$

Next, the weight of the pressurization and distribution system are calculated:

$$W_{PrSys} = A_{50} V_{FuTk} + A_{48} V_{OxTk} + A_{52}$$

$$W_{FuSys} = A_{44} T_{Tot} + A_{45} L_{Body} + A_{46}$$

$$W_{OxSys} = A_{47} T_{Tot} + A_{48} L_{Body} + A_{46}$$

If the tanks are integral their weight is calculated based on the following equations:

$$W_{InFuT} = A_{130} V_{FuTk} + A_{131}$$

$$W_{InOxT} = A_{132} V_{OxTk} + A_{133}$$

If they are not integral tanks, different coefficients are used and the equations are

$$W_{InFuT} = A_{38} V_{FuTk} + A_{37}$$

$$W_{InOxT} = A_{38} V_{OxTk} + A_{39}$$

If the fuel is storable then a totally different set of equations are used to find the fuel system weight. The weight of the fuel container is:

$$W_{FuCnt} = A_{36} \left(\frac{Gal}{N_{Tanks}} \right)^{0.6} N_{Tanks} + A_{37}$$

The pressurization system, boost pump, and fuel distribution system weights are given by:

$$W_{PrSys} = .009\ T_{Tot} N_{Tanks}$$

$$W_{BPump} = .001\ T_{Tot} (1.75 + .266\ N_{Eng})$$

$$W_{Dist1} = N_{Eng} A_{104} \left[\frac{T_{Tot}}{N_{Eng}} \right]^{0.5}$$

$$W_{Dist2} = .255\ Gal^{0.7} N_{Tanks}^{0.25}$$

The fuel control, refueling system, drains, and seal system weights are found using:

$$W_{FCont} = 0.169\ N_{Tanks} \sqrt{Gal}$$

$$W_{ReFuel} = N_{Tanks}(3 + .45\ Gal^{0.333})$$

$$W_{Drain} = .159\ Gal^{0.65}$$

$$W_{Seal} = .045\ N_{Tanks}(Gal/NTanks)^{0.75}$$

The weight of the complete storable fuel system is then the summation of the components calculated above.

$$W_{FuSys} = W_{BPump} + W_{Dist1} + W_{Dist2} + W_{FCont} + W_{ReFuel} + W_{Drain} + W_{Seal}$$

The final calculation of the main propulsion system is to sum all the values found so far:

$$W_{PropU} = W_{trj} + W_{RamJ} + W_{REng} + W_{FuCnt} + W_{OxCnt} + W_{InsFT} + W_{InsOT} + W_{FuSys} + W_{OxSys} + W_{PrSys} + W_{Inlet}$$

this gives the total weight for the main propulsion system.

At this point we have enough information to calculate the weight of the body structure it self. This is done by summing the components that form the structural part of the fuselage. This includes the integral fuel and oxidizer tanks if they are used.

$$W_{Body} = W_{Basic} + W_{SecSt} + W_{ThrSt} + W_{InFuT} + W_{InOxT}$$

Next, the weight of the crew and their life support system is calculated. This will use the following equation. Note that if the vehicle is unmanned that the equation is still used but should be equal to zero, unless the user has input a fixed weight for A_{73}.

$$W_{Crew} = A_{72}\ N_{Crew} + A_{73}$$

The last thing to be calculated before the iterative section is the fuel data. The weights to be calculated are for the main, reserve, lost, and residual propellants. The main fuel and oxidizer are calculated as follows:

$$W_{FuelM} = W_{PMain} \ /\ (1 + f)$$

$$W_{OxidM} = W_{FuelM} /$$

$$W_{PropM} = W_{FuelM} + W_{OxidM}$$

The reserve fuel is that which is held onboard for a later maneuver or for contingency reasons. The reserve fuel is found with these equations:

$$W_{FResv} = A_{84} W_{FuelM} + A_{85}$$

$$W_{OResv} = A_{86} W_{OxidM} + A_{87}$$

$$W_{Resp} = W_{FResv} + W_{OResv}$$

The lost fuel is the fuel that is lost during flight either through boil off with cryogenic fuels, or leakage with storable fuel. This part of the fuel weight can be calculated using the following equations.

$$W_{FLoss} = A_{116} W_{FuelM}$$

$$W_{OLoss} = A_{116} W_{OxidM}$$

$$W_{PLoss} = W_{FLoss} + W_{OLoss}$$

Residual fuel is the fuel that is unable to be removed from the tank during operation. This is basically trapped fuel or ullage. The equations used to find the trapped fuel are:

$$W_{FTrap} = A_{92} W_{FuelM} + A_{93}$$

$$W_{OTrap} = A_{94} W_{OxidM} + A_{95}$$

$$W_{Resid} = W_{FTrap} + W_{OTrap}$$

At this point all of the weights that can be calculated in a close-form method have been found. The rest of the vehicle's weight is dependant on functions that are based on either take-off, landing, or dry weight. Therefore, some iteration scheme is needed.

The method used starts with an approximate value of take-off and landing weights. The weights of all the remaining components are calculated and then summed to form a new estimate at the final weights. This series of steps continues until the percent error between the last estimate and the present one is lower then 0.01%.

The first of the equation sets needing the final weights are the equations for the aero surfaces. These equations are used only if the flag IShape is set to 1 which indicates that the vehicle does have wings and horizontal surfaces.

The weight of the wing has three basic parts. The first is based on the take-off weight, and the second on landing weight. They are used to base the weight of the wing on the amount of load it will encounter. The equation works by having the coefficient, either A_1 or A_2, for the part that is not being used set to zero. This would either be done during PD1 (§ 2.3) or by the user creating a custom data file. The remaining part is the weight based solely on surface area of the wing and the constant wing weight. These represent, basically, weight penalties that can be set by the user.

$$W_{Wing} = \frac{A_1}{10^6}\left[\frac{W_{TO}X_{LF}b_{st}S_{Wing}}{T_{Root}}\right]^{A_{78}} + A_2 S_{Wing} + A_3 + \frac{A_{117}}{10^9}\left[\frac{W_{TO}X_{LF}b_{st}S_{Wing}}{T_{Root}}\right]^{A_{118}}$$

The next equation is the one for horizontal surface weight. It is similar to wing weight, in that it has parts for both the take-off and landing conditions.

$$W_{Horz} = A_6\left(\left[\frac{W_{TO}}{S_{Wing}}\right]^{0.6} S_{Horz}^{1.2}\, q_{max}^{0.8}\right)^{A_{90}} + A_7 + A_{199}\left(\left[\frac{W_{TO}}{S_{Wing}}\right]^{0.6} S_{Horz}^{1.2}\, q_{max}^{0.8}\right)^{A_{120}}$$

At this point all of the individual items that make up the aero surfaces have been calculated. The only thing left to do is to sum them up to get the total weight of the aero surfaces.

$$W_{Surf} = W_{Wing} + W_{Vert} + W_{Horz} + W_{Fair}$$

The next weight group to be found is the weight of the orientation controls and separation system. The orientation controls are made up of the engine gimbal system used to deflect the rocket engine to change direction, orbital maneuvering system (OMS) engines, and vernier engines. Also, any aero control surfaces used to provide guidance and control, such as aeilerons, trim tabs, etc.

The following equations give the weights for the gimbal system, OMS engines, fuel and oxidizer, tankage, aero controls, separation system, and reserve OMS fuel.

$$W_{Gimbal} = A_{55}T_{Del}^{A_{110}} + A_{56}$$

$$W_{ACS} = A_{57}W_{TO}^{A_{58}} + A_{59} + A_{124}W_{Entry}^{A_{125}}$$

$$W_{ACSFu} = A_{96}W_{Entry} + A_{97}$$

$$W_{ACSOx} = W_{ACSFu}\, f_{ACS}$$

$$W_{ACSTk} = A_{64}(W_{ACSFu} + W_{ACSOx}) + A_{65}$$

$$W_{Aero} = A_{60}\left[W_{TO}^{0.667}(L_{Body}+b)^{0.25}\right]^{A_{111}} + A_{61}$$

$$+ A_{112}\left[W_{Entry}^{0.667}(L_{Body} + b)^{0.25}\right]^{A_{123}}$$

$$W_{Sep} = A_{62}W_{TO} + A_{63}$$

$$W_{ACSP} = W_{ACSFu} + W_{ACSOx}$$

$$W_{ACSRe} = A_{115}W_{ACSP}$$

Summing up the weights of these components we have:

$$W_{Ornt} = W_{Gimbal} + W_{ACS} + W_{ACSTk} + W_{Aero} + W_{Sep}$$

The next group to calculated is the power supply, conversion, and distribution system. This system is broken down into two systems, the electrical and hydraulic, also hydro-pneumatic system.

$$W_{Elect} = A_{66}\left[\sqrt{W_{TO}}L_{Body}^{0.25}\right]A_{112} + A_{67} +$$

$$A_{126}\left[\sqrt{W_{Entry}}L_{Body}^{0.25}\right]A_{127}$$

The hydro-pneumatic system's weight can be found using this equation:

$$W_{HyPnu} = A_{68}\left[\left[S_{Wing} + S_{Horz} + S_{Vert}\right]\frac{q_{Max}}{1000}\right]^{0.334}$$

$$+ \left[\sqrt{L_{Body}b_{st}}\,T_{Tail}\right]^{A_{113}} + A_{69} + A_{128}W_{TO} + A_{129}W_{Entry}$$

The power system is just the total of these two subsystems.

$$W_{PwrSy} = W_{Elect} + W_{HyPnu}$$

The remaining systems yet to be described are relatively simple. The weight of the avionics is given by:

$$W_{Avonc} = A_{70}W_{TO}^{A_{114}} + A_{71}$$

The aircraft's crew systems, such as life support, rescue, etc. is found using the following formula.

$$W_{CProv} = A_{74}W_{TO} + A_{80}N_{Crew} + A_{75}$$

On many vehicle designs there are parts of the vehicle that are used during launch or separation, and/or landing. Examples of this would be the landing gear on horizontally launched or recovered aircraft. The gear does not have to be used during both launch and recovery. One example that clearly illustrates this is the Space Shuttle, its gear are sized for landing only, they don't have to support the weight of the entire vehicle. The savings in strength is directly translated into a weight savings.

The following equations provide the weights of the launch and recovery equipment. The equation for the gear has two parts one is for landing weight and the other for weight at take-off. the coefficient for one of the two should be set to zero. The is taken care of in PD1 (§ 2.3.3), but if you are using a user written file care must be taken.

$$W_{Launch} = A_{23} W_{TO} + A_{24}$$

$$W_{LGear} = A_{25} W_{TO}^{A_{101}} + A_{26} W_{Land}^{A_{121}} + A_{27}$$

$$W_{Gear} = W_{Launch} + W_{LGear}$$

The vehicle's dry weight can now be found. This is the weight of the vehicle's structure and the inanimate objects that would be found secured inside. This does not include payload or fuel, and is similar to the weight of the vehicle as delivered or when put into storage.

$$W_{Dry} = W_{Surf} + W_{Body} + W_{TPS} + W_{Gear} + W_{PropU} + W_{Ornt} + W_{PwrSys} + W_{Avonc} + W_{CProv}$$

A factor often included in the weights analysis is the design contingency. This is based on the dry weight of the vehicle and is a figure of how much growth is designed into the vehicle. Often this number is about 5% of the vehicle's dry weight.

$$W_{Cont} = A_{98} W_{Dry} + A_{99}$$

The empty weight of the vehicle is the dry weight of the vehicle plus the contingency weight. On many vehicles designed for a particular purpose these weights would be the same.

$$W_{Empty} = W_{Dry} + W_{Cont}$$

The landing weight of the vehicle is the next greater and is made up of the empty weight plus the weight of what ever is brought back from the mission. In this case the payload is assumed to be brought back. At first glance this may seem like a rather foolish assumption, but in the case of an experimental payload, or recconasance package it begins to make sense. Also, the weight of the crew, and any residual fuel are included in the landing weight.

$$W_{Land} = W_{Empty} + W_{PayLd} + W_{Crew} + W_{Resid} + W_{ACRe}$$

Adding the weight of the reserve and OMS propellants to the landing weight gives the entry, or insertion weight.

$$W_{Entry} = W_{Land} + W_{ACSP} + W_{ResP}$$

The final weight is the gross take-off weight. This is the sum of the entry weight and the total amount of propellant aboard.

$$W_{TO} = W_{Entry} + W_{PMain} + W_{PLoss}$$

Once the take-off weight is calculated it is compared with the value calculated on the last pass through the iterative part of the routine. If it is not within the tolerance then the process repeats itself with the new values of the gross take-off, entry, and landing weights and recalculates the other weights until the tolerance is achieved.

2.6 GetData

2.6.1 Purpose

This subroutine handles the actual I/O for WAATSb(§ 2.4). It gets the filenames for the user and reads in the input data file. It also checks for any DOS errors and sends program control to the error handling routine ErrChk (§ 2.21). The reason for having a separate I/O routine was to enable the code to be easily modified for different systems as the need arouse.

GetData also sets several constants and the data required for the atmospheric subroutine, Atmos (§ 2.7).

2.6.2 Data Referenced

GetData uses the input data file specified by the user. It reads in the values and assigns these to there appropriate variables. The subroutine also reads in the data stored in the atmospheric table. This table contains the base height, lapse rates, base pressure (in mm Hg), base temperature (in deg K), and an indicator for which pressure model to use.

2.6.3 Processing Performed

The processing which GetData does is limited to getting the file names for the input and output data files, reading the input files and setting the necessary constants which will be used by CalcWts (§ 2.5).

2.7 Atmos

2.7.1 Purpose

Atmos is an atmospheric modeling subroutine based on the 1962 Standard Atmosphere. It provides the data required to size and approximate the weight of the airbreathing engines in CalcWts (§ 2.5).

2.7.2 Data Referenced

Atmos does not directly access any data tables. All data which it uses has been stored in arrays by GetData (§ 2.6).

2.7.3 Processing Performed

The subrutine Atmos first finds the index for the constants that must be found in the arrays mentioned above. This involves finding the isotherm or lapse rate, and base temperature which corresponds to the given geopotential altitude. The subroutine then uses that information to calculate the temperature.

Next the pressure is calculated based on a base pressure and the change in temperature from the base temperature. This information is then used to calculate the density at the given altitude.

2.8 Display

2.8.1 Purpose

Display is the routine that provides the final output for the weights analysis routine, WAATSb (§ 2.4). The output is sent to the file specified by the user during the routine GetData (§ 2.6).

The output format was chosen so the user would have a good look at the overall configuration of the vehicle, as well as the complete weight statement. To do this the output is divided into two sections. The first section contains a description and file data, all of the vehicle's main dimensions, and some of the more important ratios.

The second part of the output is the vehicle's weight statement. The style of the statement was based on MIL-M-38310B (USAF), Mass Properties Control Requirements for Missles and Space Vehicles. This defines the group assignments for the specific items and systems. The vehicle weights such as dry weight, entry weight, etc., where placed in the output where it was felt the user would have a sense of what systems or components where added to obtain that weight.

2.8.2 Data Referenced

None.

2.8.3 Processing Performed

Display is called by WAATSb (§ 2.4). The function of Display is straight forward. It prints all of the data to the file named by the user. Once this is done control is passed back to WAATSb (§ 2.4).

2.9 FileViewer

2.9.1 Purpose

FileViewer originally started out as a utility to allow the user to see the data and/or output files. In time this routine became a separate sub-menu. This was done to allow more options for the user to select from. Essentially, this has routine serves the same function as Main (§ 2.1), it provides a way to direct and control the flow of the program when it is in the FileViewer sub-menu area.

2.9.2 Data Referenced

None. The only data used is the assignment of the appropriate option names to the array sel$ to be used by MenuDisplay (§ 2.13).

2.9.3 Processing Performed

As mention above in the purpose section, the processing that FileViewer does is manage the flow of the program. Control is passed to it by Main (§ 2.1), and then based on the user's choice control is temporarily passed to one of the subroutines SOut (§ 2.10), Convert (§ 2.12), or DOSShell (§ 2.13). When the user selects to return to the main menu program control is then passed back to Main (§ 2.1).

2.10 SOut

2.10.1 Purpose

The reason for SOut is so the user can see the files she is working on without having to leave the program PDWAP.

2.10.2 Data Referenced

None.

2.10.3 Processing Performed

The subroutine SOut is called by the routine FileViewer (§ 2.9). The first thing the subroutine does is ask the user what file he would like to see. If the user has input a filename the subroutine then clears the screen and creates a DOS shell. The program then calls the COM file PD1 with the filename given by the user as an input parameter.

The user can now view his file, and after exiting PD1 wil be returned back to PDWAP's File Viewer menu.

2.11 DOSShell

2.11.1 Purpose

The subroutine DOSShell was added to the PDWAP system to allow the user to exit to a temporary DOS shell to perform regular DOS functions, or run a second program without having to exit PDWAP, memory permiting.

2.11.2 Data Referenced

None. This function makes use of the DOS functions already present in the operating system. To create the DOS shell, however, DOS must have access to the file COMMAND.COM to set up the shell. If this file is not available, then DOS will print a message to that effect and control will revert back to PDWAP. At that point the File Veiwer menu will be dispalyed.

2.11.3 Processing Performed

In essence, none. This subroutine simply passes temporary control to the operating system. Once the user is finished and types EXIT at the DOS prompt she will be returned to PDWAP.

2.12 Convert

2.12.1 Purpose

The purpose of the subroutine Convert is to convert existing version 2.0 data files to the new 2.1 format. This, of course, was done as a means to conserve the work that was done using the older version of PDWAP. This also is a means of assuring new users that their time invested in using PDWAP is a concern of the author, and any new releases of PDWAP will have this feature as well. This will enable users to upgrade without the pain of having to either lose data or redue alot of work.

2.12.2 Data Referenced

Convert use the filename given by the user as the old file and reads that in as it makes the required corrections and additions.

2.12.3 Processing Performed

The subroutine Convert after having asked the user for the old filename and the filename he wishes to use for the new file. If the user wishes to keep the same filename he can enter it as the new name as well. PDWAP will create a temporary file, write the data to it and then kill the old file and then rename the temporary file to replace it. The subroutine then askes the user for the vehicle name and description and then starts to make the additions to create the new data file.

2.13 MenuDisplay

2.13.1 Purpose

This subroutine provides the initial screen for all of the routines, plus does the menus when the are needed. To provide the menu function of moving the highlite bar, the subroutine uses several very small subroutines which are straight forward and particular to the Turbo-BASIC language. The will not be covered in this manual.

2.13.2 Data Referenced

None, with the exception of the array containing the menu choices, if needed.

2.13.3 Processing Performed

In general this subroutine just does the screen I/O for PDWAP. The user interface is very rudimentary, this was done intentionally so it could operate on a wider variety of machines. Since the processing that is done is specific to the PC class machine and the language used here, and also since it has no bearing on the calculations performed by the program, it will not be covered in depth here.

2.14 ErrChk

2.14.1 Purpose

This subroutine is a general purpose error handling routine. It is used to alert the user to any possible system errors. This mainly includes disk access and printer errors. The use of this subroutine cuts down on the possibility of a systen crash or the program bombing because of a minor DOS error.

2.14.2 Data Referenced

The only data used by this subroutine is the DOS error code returned in the variable err. This value is assigned by the system when an error condition is encountered.

2.14.3 Processing Performed

This subroutine just has a set of logical checks that are performed to determine the value of the error code, and therefore the cause of the error. It then alerts the user, and returns control to the location in the code that the error ocurred on. Now, if the user has cleared the error causing condition (ie: put paper in the printer for an 'Out of Paper' error), the program will continue.

References

The following is more a list of selected reading on the subject of weights analysis and preliminary design then a reference list. During the process of writing the code that became PDWAP much reading was done to gain some insight into the "black magic" field of weights analysis.

The main source of information that I used was the report by Glatt on the program WAATS. This is suggested as a companion reference to the PDWAP User's and Technical Manuals. In order to keep the program as near to WAATS as possible, I purposely used the same variable names, where they made sense, in PDWAP a in WAATS. Hopefully this will allow the user to follow the code and procedures used to implement this program.

Franciscus, L. C. and J. L. Allen: Upper-Stage Space-Shuttle Propulsion by Means of Seperate Scramjet and Rocket Engines. NASA TN D-6762. May 1972. (NOTE: Used to style the GDV.)

Glatt, C. R.: WAATS - A Computer Program for Weights Analysis of Advanced Transportation Systems. NASA CR-2420. September 1974.

MacConochie, I. O. and R. W. Kilch: Techniques for the Determination of Mass Properties of Earth-to-Orbit Transportation Systems. NASA TM-78661. June 1978.

MacConochie, I. O. and R. W. Lemessurier: Weight Trends for a Fully Reusable Single-Stage Shuttle. NASA TM-85656. September 1983.

MacConochie, I. O., J. A. Martin, C. A. Brenier, J. A. Cerro: Weights Assesment for Orbit-on-Demand Vehicles. Society of Allied Weight Engineers, Paper No. 1674.

PDWAP v2.1
Technical Reference Manual
Appendix A - Program Listing

This appendix contains the complete listing and variable cross-reference for the program PD.BAS. The program is written in Turbo BASIC by Borland International. The line numbers at the right of the code are for reference only and are added by the crossrefernec-ing program to aid in identifying portions of code. Any duplications of line numbers indicates a continuation line, and is to be regarded as being part of the same logical line.

Following the program listing is a variable listing. This list all the variables used in the program by name and the line number(s) they were used on.

```
option base 1
dim A(134),HB(8),PB(8),ALM(8),TMB(8),JSWA(8),sel$(4)
'                                  Title Screen
cls
print "        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%"
print "        %                          PDWAP 2.1                        %"
print "        %                                                           %"
print "        %       Preliminary Design and Weights Analysis Program     %"
print "        %                              by                           %"
print "        %                     Lt Louis R. Carreiro                  %"
print "        %                                                           %"
print "        %               Analysis and Applications Branch            %"
print "        %                 Advanced Propulsion Division              %"
print "        %              Aero Propulsion and Power Laboratory         %"
print "        %             Wright Research and Development Center        %"
print "        %                   Wright-Patterson AFB, OH                %"
print "        %                                                           %"
print "        %     For documentation on PDWAP see WRDC-TR-89-XXXX        %"
print "        %  Any questions or comments about this program can be sent %"
print "        %                      to the author at:                    %"
print "        %                                                           %"
print "        %              WRDC/POPA (Attn: Lt Carreiro)                %"
print "        %              Wright-Patterson AFB, OH 45444-6563          %"
print "        %              Phone: (513) 255-4171                        %"
print "        %                                                           %"
print "        %                   Hit any key to continue                 %"
print "        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%";

a$=""
while len(a$) = 0
    a$=inkey$
wend
cls

'                        Main Menu Routine
'
'
do
    clear
    sel$(1) = "F1.  Initial Estimation"
```

```
    sel$(2) = "F2.  Weight Analysis"
    sel$(3) = "F3.  View Data File"
    sel$(4) = "F4.  Exit to DOS"
    Title$ = "Main Menu"
    gosub MenuDisplay
    on menu gosub PD, WAATSb, FileViewer
loop until menu = 4

cls
end
'======================================================================
PD:
'              Preliminary Design Routine
'
PI = 3.1415926
SIGf = 8 * PI / 180
SIGr = 30 * PI / 180
LAMle =  82 * PI / 180
RhoH2 = 4.389
RhoO2 = 71.025
Title$ = "Preliminary Design"
sel$(1) = ""

do
   gosub PD1
   input "         Perform Another Estimation (Y or N)              :",again$
loop until ucase$(again$) <> "Y"
return

PD1:
do
    gosub MenuDisplay
    locate 8,1,0
    Input "             Initialization data filename (PD.DAT)  : ",InFileN$
    if len(InFileN$) = 0 then InFileN$ = "PD.DAT"
    errf = 0
    on error goto errchk
    open InFileN$ for input as #1
loop until not errf

do
    locate 8,1
    errf = 0
    input "         File name to store data (*.DAT)                  :",OutName$
    if not instr(OutName$,".") then OutName$ = OutName$ + ".DAT"
    open OutName$ for output as #2
loop until not errf

a$ =  "         \                                              \:"
print using a$;"Weight of fuel to be carried (LBS)";
input, WFu
```

```
print using a$;"Weight Density of fuel (4.389 lb/cu. ft)";
input, RhoF$
if RhoF$ = "" then RhoF = RhoH2 else RhoF = val(RhoF$)

print using a$;"Oxidizer to fuel ratio (f) to be used";
input, f

print using a$;"Number of crewmembers";
input, NCrew

print using a$;"Number of rocket engines";
input, Nre
if Nre then
    print using a$;"Thrust per engine";
    input, Tre
end if

print using a$;"Number of Ramjet engines";
input, Nrj
if Nrj then
    print using a$;"Thrust per engine";
    input, Trj
end if

print using a$;"Number of Turbo-Ramjets engines";
input, Ntrj
if Ntrj then
    print using a$;"Thrust per engine";
    input, Ttrj
end if

while ucase$(TOA$)<>"H" and ucase$(TOA$) <>"V"
    print using a$;"Horizontal or vertical takeoff? (H or V)";
    input ,TOA$
wend

print using a$;"Payload weight";
input, WPayLd

print using a$;"Name of Vehicle";
input, VName$
PRINT "Input Vehicle Discription on the line below (Limit of 65 Characrters)"
input ,VDisc$

VDisc$ = left$(VDisc$,65)
if VDisc$="" then VDisc$ = "None"
if VName$ = "" then VName$ = "No Name"

WTO = 3000000
WO2 = WFu * f
```

```

VFuTk = WFu / RhoF
VOxTk = WO2 / RhoO2

Xf = ( VFuTk * 6/PI /(63/64*tan(SIGf)^2 + tan(SIGf)^3/tan(SIGr)))^.33333
Xc = .25 * Xf
rb = Xf * tan(SIGf)
rf = 0.75 * Xf * tan(SIGf)
Xr = rb / tan(SIGr)

rOT = rb / 4

rt = 0.9 * rb

Xm = (VOxTk / 7 / PI + rb^3/48) * 16 / rb^2

SFuTk1 = .5 * PI * (rb * sqr(Xf^2 + rb^2) - rf * sqr(rf^2 + Xc^2) _
          + rb^2 + rf^2) + rb*xf - rf*Xc
SFuTk2 = .5 * PI (rb * sqr(rb^2 + Xr^2 ) + rb^2) + Xr*rb

LBody = Xf + Xm + Xr
Xw = .66 * LBody
b = 4 * rb
h = rb
NIntake = Nrj + Ntrj

SBody = rb*(.5*PI * (sqr(rb^2+Xf^2) + sqr(rb^2+Xr^2)) + Xf + Xr + (2+PI)*Xm)

SFuTk = SFuTk1 + SFuTk2
SOxTk = 14*PI * (rt*Xm + rt^2)
SWing = 1/3 * Xw * b
STPS = .5 * PI*rb*(sqr(rb^2+Xf^2) + sqr(rb^2+xr^2) + 2*Xm) + SWing
bSt = sqr((.5 * Xw)^2 + (.5 * b)^2)
SVert = 0.1 * SWing
TRoot = 1/4 * b * tan (LAMle) * .09
WPMain = WO2 + WFu
WLand = WTO - WPMain - WPayLd
TTOT = Nre * Tre + Nrj * Trj + Ntrj * Ttrj
AICapt = 6*NIntake

line input#1, a$
     print #2, "Data file name: "; OutName$
line input#1, a$
      print# 2, "Date = ";Date$
line input#1, a$
      print# 2, "Time = ";Time$
line input#1, a$
      print# 2, "Vehicle Name = ";VName$
line input#1, a$
      print #2, "Discription = ";VDisc$
line input#1, a$
```

```
          print #2, a$
line input#1, a$
          print #2, a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; AICapt;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; NCrew;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; LBody;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; LInlet;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; LRamp;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; f;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; b;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; h;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; Nre;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; Nrj;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; Ntrj;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
input #1, a, a$
          print #2, Using "########.#####"; a;: print #2, ,a$
```

```
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; SBody;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; SFuTk;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; SOxTk;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; STPS;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; bSt;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; SVert;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; SWing;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; NTanks;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; Tre;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; Trj;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; Ttrj;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; TRoot;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; VFuTk;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; VOxTk;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; WLand;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; WPayLd;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; WPMain;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; WTO;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; NIntake;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
input #1, a, a$
        print #2, Using "#######.#####"; a;: print #2, ,a$
```

```
        for I = 1 to 134
           input #1, a, a$
           if I = 1 and ucase$(TOA$) = "V" then a = 0
           if I = 23 and ucase$(TOA$) = "V" then a = 0.0025
           if I = 25 and ucase$(TOA$) = "V" then a = 0
           if I = 26 and ucase$(TOA$) = "V" then a = 0.00916
           if I = 44 then a = 0.009649*EXP(-0.00001152*TTOT)
           if I = 45 then a = 0.1121 + 0.01724 * TTOT/1000
           if I = 47 then a = 0.01425*EXP(-0.000025*TTOT)
           if I = 48 then a = 0.739 + 0.00675 * TTOT/1000
           if I = 57 and ucase$(TOA$) = "V" then a = 0
           if I = 58 and ucase$(TOA$) = "V" then a = 0
           if I = 60 and ucase$(TOA$) = "V" then a = 0
           if I = 66 and ucase$(TOA$) = "V" then a = 0
           if I = 78 and ucase$(TOA$) = "V" then a = 0
           if I = 80 then a = RhoF
           if I = 101 and ucase$(TOA$) = "V" then a = 0
           if I = 111 and ucase$(TOA$) = "V" then a = 0
           if I = 112 and ucase$(TOA$) = "V" then a = 0
           if I = 117 and ucase$(TOA$) = "V" then a = 1624
           if I = 118 and ucase$(TOA$) = "V" then a = 0.584
           if I = 121 and ucase$(TOA$) = "V" then a = 1.124
           if I = 123 and ucase$(TOA$) = "V" then a = 0.903
           if I = 124 and ucase$(TOA$) = "V" then a = 78.5
           if I = 125 and ucase$(TOA$) = "V" then a = 0.079
           if I = 126 and ucase$(TOA$) = "V" then a = 1.167
           if I = 127 and ucase$(TOA$) = "V" then a = 1.0
           if I = 128 and ucase$(TOA$) = "V" then a = 0
           if I = 129 and ucase$(TOA$) = "V" then a = 0.00114
           print #2, Using "#######.#####"; a;: print #2, ,a$
        next I
close
return
'============================================================================
'                                 WAATSb
'                               VERSION 1.0
'
'      Weights Analysis for Advanced Transportation Systems (BASIC)
'
'         Based on the algorithm in the program WAATS as outlined
'         in NASA CR-2420 by C. R. Glatts.
'
'         Modified by L. R. Carreiro for use with TurboBASIC.
WAATSb:
do
       GOSUB GetData
       GOSUB CalcWts
       GOSUB Display
       print
```

```
    input "        Perform Another Estimation? (Y or N) : ",again$
loop until again$ <> "Y" or again$ <> "y"
return
'================================================================

CalcWts:
'              Calculation of weights based on historical data

    NEng = Ntrj + Nrj + Nre
    TTOT = (Nrj*Trj + Nre*Tre + Ntrj*Ttrj)
    Gal = 7.481# * VFuTk


'   Aerodynamic Surfaces
    if IShape then
         WFair = A(8) * SFair + A(9)
         WVert = A(4) * SVert ^ A(89) + A(5)
    end if

'   Body Structure
    WBasic = A(14) * SBody
    WBasic = WBasic + A(15) * ((LBody*XLF/h)^0.15 * QMax^0.16 * SBody^1.05)^A(81) + A(16)
    WSecSt = A(17) * SBody + A(18)
    WThrSt = A(19) * (Trj*Nrj + Ttrj*Ntrj) + A(10)*Tre*Nre + A(20) + A(11)



'   Thermal Protection System
    WInsul = A(21) * STPS + A(76)
    WCover = A(22) * STPS + A(77)

    WTPS = WInsul + WCover


'   Main Propulsion

    if Ntrj then
         CMT = DM
         CHT = DH
         GOSUB Atmos
         X = 1# + 0.2# * CMT^2
         PTO = PAlt * X^3 * sqr(X)

         if CMT <= 1 then
              PR = 1#
         elseif 1 < CMT and CMT <= 5 then
              PR = 1# - 0.075# * (CMT-1#)^1.35
         else
              PR = 800# / (CMT^4 + 935#)
         end if
         PT2 = PR * PTO
```

```
          WA = WARef * Actr
          WEngMt = A(102) * Ttrj + A(103)
          Wtrj = (A(32) * EXP(A(33) * WA) * ((PT2 -_
               PHigh)/(PLow - PHigh)) + A(34) *_
               EXP(A(35) * WA) * ((PT2 - PLow)/(PHigh - PLow)))_
               * Ntrj + WEngMt*Ntrj
     end if

     if Nrj then
          WEngMt = A(102) * Trj + A(103)
          Wrj = (A(82) * Trj + A(83)) * Nrj + WEngMt*Nrj
     end if

     if Nre then
          WEngMt = A(12) * Tre + A(13)
          Wre = (A(28) * Tre + A(29)   Tre * ARatio^A(30)_
               + A(31) + WEngMt) * Nre
     end if

     if Ntrj OR Nrj then
          WIDuct = A(53) * ((LInlet*NIntake)^0.5 * (AICapt / NIntake)_
        ^0.3334 * PT2^0.6667 * GFct * FMach) ^ A(54)_
               + A(105)
          WVRamp = A(106) * (LRamp * NIntake * (AICapt/NIntake)^0.5_
               * TFct)^A(107) + A(108)

      WSpike = A(109) * NIntake
          WInlet = WIDuct + WVRamp + WSpike
     else
          WInlet = 0#
     end if

     if ICry then
          WInsFT = A(40) * SFuTk + A(41)
          WInsOT = A(42) * SOxTk + A(43)
          WFuSys = A(44) * TTOT + A(45) * LBody + A(46)
          WPrSys = A(50) * VFuTk + A(51) * VOxTk + A(52)
          WOxSys = A(47) * TTOT + A(48) * LBody + A(49)
          if Integral then
              WInFuT = A(130) * VFuTk + A(131)
              WInOxT = A(132) * VOxTk + A(133)
          else
              WFuCnt = A(38) * VFuTk + A(37)
              WOxCnt = A(38) * VOxTk + A(39)
          end if

    elseif not ICry then
          WFuCnt = A(36) * (Gal/NTanks)^0.6 * NTanks + A(37)
          WPrSys = 0.0009 * TTOT * NTanks
          WBPump = TTOT/1000# * (1.75 + 0.266 * NEng)
          WDistl = NEng * A(104) * (TTOT/NEng)^0.5
```

```
        WDist2 = 0.255 * Gal^0.7 * NTanks^0.25
        WFCont = 0.169 * NTanks * Gal^0.5
        WReFul = NTanks * (3.0 + 0.45 * Gal^0.333)
        WDrain = 0.159 * Gal^0.65
        WSeal = 0.045 * NTanks * (Gal/NTanks)^0.75
        WFuSys = WBPump + WDist1 + WDist2 + WFCont + WReFul + WDrain + WSeal
   end if

   WPropU = Wtrj + Wrj + Wre + WEngMt + WFuCnt + WOxCnt + WInsFT _
        + WInsOT + WFuSys + WOxSys + WPrSys + WInlet

   WBody = WBasic + WSecSt + WThrSt + WInFuT + WInOxT

'  Crew and Crew Life Support
   WCrew = A(72) * NCrew + A(73)


'  Main Propellants
   WFuelM = WPMain / (1# + f)
   WOxidM = WFuelM * f
   WPropM = WFuelM + WOxidM

'  Reserve Propellant
   WFResv = A(84) * WFuelM + A(85)
   WOResv = A(86) * WOxidM + A(87)
   WResP = WFResv + WOResv

'  Inflight Losses
   WFLoss = A(116) * WFuelM
   WOLoss = A(116) * WOxidM
   WPLoss = WFLoss + WOLoss

'  Residual Propellants
   WFTrap = A(92) * WFuelM + A(93)
   WOTrap = A(94) * WOxidM + A(95)

   WResid = WFTrap + WOTrap




do 'Iterate on Gross Takeoff Weight

'  Aerodynamic Surfaces
   if IShape then
        WWing = A(1) * (WTO * XLF * bSt * SWing / TRoot)^A(78) * 1E-6 +_
               A(2) * SWing + A(3) + A(117) * (WLand * XLF * bSt * SWing _
              / TRoot * 1E-9)^A(118)
        WHorz = A(6) * ((WTO / SWing)^0.6 * SHorz^1.2 * QMax^0.8)^A(90)
        WHorz = WHorz + A(7)
        WHorz = WHorz + A(119) * ((WLand / SWing)^0.6 * SHorz^1.2 * QMax^0.8)^A(120)
```

```
    end if

    WSurf = WWing + WVert + WHorz + WFair

  ' Orientation Controls and Seperation
    WGimbal = A(55) * TDel^A(110) + A(56)
    WACS = A(57) * WTO^A(58) + A(59) + A(124) * WEntry^A(125)
    WACSFu = A(96) * WEntry + A(97)
    WACSOx = WACSFu * fACS
    WACSTk = A(64) * (WACSFu + WACSOx) + A(65)
    WAero = A(60) * (WTO^0.667 * (LBody + b)^0.25)^A(111)
    WAero = WAero + A(61)
    WAero = WAero + A(122) * (WEntry^0.667 * (LBody + b)^0.25)^A(123)
    WSep = A(62) * WTO + A(63)
    WACSP = WACSFu + WACSOx
    WACSRe = A(115) * WACSP

    WOrnt = WGimbal + WACS + WACSTk + WAero + WSep


  ' Power Supply, Conversion, and Distribution
    WElect = A(66) * (sqr(WTO) * LBody^0.25)^A(112) + A(67)
    WElect = WElect + A(126) * (sqr(WEntry) * LBody^0.25)^A(127)
    WHyPnu = A(68) * ((SWing + SHorz + SVert) * QMax / 1000#)^0.334
    WHyPnu = WHyPnu + (sqr(LBody + bSt) * TyTail) ^ A(113) + A(69)
    WHyPnu = WHyPnu + A(128) * WTO + A(129) * WEntry

    WPwrSy = WElect + WHyPnu

  ' Avionics
    WAvonc = A(70) * WTO^A(114) + A(71)

  ' Aircraft Crew Systems
    WCProv = A(74) * WTO + A(80) * NCrew + A(75)

  ' Dry Weight
    WDry = WSurf + WBody + WTPS + WGear + WPropU + WOrnt + WPwrSy_
         + WAvonc + WCProv

  ' Design Reserve (Contigency)
    WCont = A(98) * WDry + A(99)

  ' Empty Weight
    WEmpty = WDry + WCont

  ' Launch and Recovery Gear
    WLaunch = A(23) * WTO + A(24)
    WLG = A(25) * WTO^A(101) + A(26) * WLand^A(121) + A(27)

    WGear = WLaunch + WLG
```

```
'    Landing Weight
      WLand = WEmpty + WPayLd + WCrew + WResid + WACSRe

'    Entry Weight
      WEntry = WLand + WACSP + WResP

'   Takeoff Gross Weight
      WTOi = WTO
      WTO = WEntry + WPMain + WPLoss

'   Iterate with WTO
loop until abs(WTO - WTOi)/WTOi < .0001

return
'--------------------------------------------------------------------------
'                        Screen and Initialization
'                           Routine for WAATSb

GetData:
     CLS
     Title$ = "Weights Analysis"
     sel$(1) = ""
     do
         errf = 0
         gosub MenuDisplay
         locate 11,1

         input "               Input datafile name (*.DAT)   :",InFileN$
         dot% = INSTR(InFileN$,".")
         if dot% then
            FileRoot$ = LEFT$(InFileN$,dot%-1)
         else
            FileRoot$ = InFileN$
            InFileN$ = InFileN$ + ".DAT"
         end if
         on error goto errchk
         open InFileN$ for input AS #1
     loop until not errf

     do
        OutFileN$ = ""
        locate 13,1
        print"               Output datafile name (";FileRoot$;".OUT) :";
        Input, OutFileN$

        if OutFileN$ = "" then OutFileN$ = FileRoot$ + ".OUT"

        errf = 0
        on error goto errchk
     open OutFileN$ for output as #2
     loop until not errf
```

```
      line input#1, a$
      line input#1, a$
          FDate$ = mid$(a$,7)
      line input#1, a$
         FTime$ = mid$(a$,7)
      line input#1, a$
         if len(a$) <=15 then
           VName$ = "No Name"
         else
           VName$ = mid$(a$,16)
         end if
      line input#1, a$
         if len(a$) <=14 then
           VDisc$ = "None"
         else
           VDisc$ = mid$(a$,15)
         end if
      line input#1, a$
      line input#1, a$
      input #1, Actr, a$
      input #1, AICapt, a$
      input #1, ARatio, a$
      input #1, NCrew, a$
      input #1, DH, a$
      input #1, DM, a$
      input #1, LBody, a$
      input #1, LInlet, a$
      input #1, LRamp, a$
      input #1, FMach, a$
      input #1, f, a$
      input #1, GFct, a$
      input #1, b, a$
      input #1, h, a$
      input #1, ICry, a$
      input #1, Integral, a$
      input #1, IShape, a$
      input #1, Nre, a$
      input #1, Nrj, a$
      input #1, Ntrj, a$
      input #1, fACS, a$
      input #1, PCham, a$
      input #1, PHigh, a$
      input #1, PLow, a$
      input #1, QMax, a$
      input #1, SBody, a$
      input #1, SFair, a$
      input #1, SFuTk, a$
      input #1, SHorz, a$
      input #1, SOxTk, a$
      input #1, STPS, a$
```

```
    input #1, bSt, a$
    input #1, SVert, a$
    input #1, SWing, a$
    input #1, NTanks, a$
    input #1, Tre, a$
    input #1, Trj, a$
    input #1, Ttrj, a$
    input #1, TRoot, a$
    input #1, TyTail, a$
    input #1, VFuTk, a$
    input #1, VOxTk, a$
    input #1, WLand, a$
    input #1, WPayLd, a$
    input #1, WPMain, a$
    input #1, WTO, a$
    input #1, NIntake, a$
    input #1, XLF, a$
    input #1, WARef, a$

    for I = 1 TO 134
        input #1, A(I), a$
    next I

    close #1

    g = 32.174
    RE = 20920000
    AK = 0.3048#
    PSL = 760#
    TSL = 288.15#
    CHt1 = -1#
    V1 = -1#
    CMt1 = -1#
    C1 = 28.9664# * 144.0# / (1545.31# * g)

    for I = 1 TO 8
        read HB(I), ALM(I), PB(I), TMB(I), JSWA(I)
    next I
    restore

       'HB        ALM        PB          TMB        JSWA
data      0,   -0.0065,   760,         288.15,      1
data  11000,    0,        169.79,      216.65,      1
data  20000,    0.001,     41.0649,    216.65,      1
data  32000,    0.0028,     6.51064,   228.65,      1
data  47000,    0,          0.8318598, 270.65,      0
data  52000,   -0.002,      0.44254,   270.65,      1
data  61000,   -0.004,      0.136585,  252.65,      1
data  79000,    0,          0.0077834, 180.65,      0

return
```

```
'---------------------------------------------------------------------------
'                    1962 Atmosphere Routine
Atmos:
'        Altitude must be less then 299500 ft.

     if GH < 11000 then
        IH = 1
     elseif 11000 < GH and GH < 20000 then
        IH = 2
     elseif 20000 < GH and GH < 32000 then
        IH = 3
     elseif 32000 < GH  and GH < 47000 then
        IH = 4
     elseif 47000 < GH and GH < 52000 then
        IH = 5
     elseif 52000 < GH and GH < 61000 then
        IH = 6
     elseif 61000 < GH and GH < 79000 then
        IH = 7
     else
        IH = 8
     end if

     TMPB = TMB(IH) + ALM(IH) * (GH - HB(IH))
     if JSWA(IH) then
          EX = 0.0341631195# * (GH - HB(IH))
          P = PB(IH) * (TMB(IH) / TMPB)^EX
     else
          EX = (-0.0341631195# * (GH - HB(IH))) / TMB(IH)
          P = PB(IH) * EXP(EX)
     end if

     Delta = P / PSL
     Theta = TMPB / TSL
     RTheta = sqr(Theta)

          TMPA = ALM(IH) * ((re^2 * AK) / (re + HK)^2)
     DTDH = TMPA / 2# / TMPB

     if IMV = 1 then CMT = V / (1116.89# * RTheta) _
          else V = 1116.89# * RTheta * CMT

     QO = 1481# * Delta * CMT^2
     PAlt = P * 0.01933385#
     TAlt = TMPB * 1.8#
     Rho = C1 * PAlt / TAlt
     AMU = 1.456E-6 * TMPB * sqr(TMPB + 110.4#) * 7.2330137# / g
     RENO = Rho * V / AMU

return
'---------------------------------------------------------------------------
```

```
'                        Output Display Routine
'                            for WAATSb
Display:

print #2,
print #2,space$(31);"Vehicle Summary"
print #2,
print #2,space$(27);"Vehicle Name: ";VName$
print #2,"Vehicle Discription: ";VDisc$
print #2,
print #2, using"Data File = \             \              Output File = \";ucase$(InFileN$);ucase$(OutFileN$)
print #2, using"     Date = \         \                   Date = \          \";FDate$;Date$
print #2, using"     Time = \         \                   Time = \          \";FTime$;Time$
print #2,
print #2, using"Length = ######,.   Height = ######,.   Span  = ######,.   TRoot =   ##.#";LBody,h,b,TRoot
print #2, using" SBody = ######,.    SWing = ######,.   SVert = ######,.   SHorz = #####,.";SBody,SWing,SVert,SHorz
print #2,
print #2, using" Number of: RamJets = ##         Rockets = ##         TurbRamJets = ##";Nrj,Nre,Ntrj
print #2, using" Thrust of:      = #######,.           = #######,.             = #######,.";Trj,Tre,Ttrj
print #2,
print #2, using"Fuel = ########,.            Fuel Density = ###.###          LOX = ########,.";WFuelM,A(80),WOxidM
print #2,
print #2, using"Weights:  GTOW = ########,.       Payload = ########,.    Dry = ########,. ";WTO,WPayLd,WDry
print #2, using"        Landing = ########,.        Entry = ########,. ";WLand,WEntry
print #2,
print #2, using"Ratios:  GTOW/S = ####.##     AR = ##.###     T/GTOW = ##.##     T/S = ###.##";WTO/SWing,b^2/SWing,TTOT/WTO,TTOT/SWing
print #2,
print #2,      "                             Weight Statement"
print #2,"------------------------------------------------------------------------------"
print #2, using"       Group 1:   Aero surfaces             ########,.";WSurf
print #2, using"                     Wing                   ########,.";WWing
print #2, using"                     Vertical               ########,.";WVert
print #2, using"                     Horizontal             ########,.";WHorz
print #2, using"                     Fairing                ########,.";WFair
print #2,
print #2, using"       Group 2:   Body structure                         ########,.";WBody
print #2, using"                     Basic body             ########,.";WBasic
print #2, using"                     Secondary              ########,.";WSecSt
print #2, using"                     Thrust                 ########,.";WThrSt
print #2, using"                     Integral fuel tanks    ########,.";WInFuT
```

```
795 print #2, using"                    Integral Ox tanks        ########,.";WInOxT
796 print #2,
797 print #2, using"   Group 3:   Thermal Protection System                    ########,.";
797 WTPS
798 print #2, using"                    Vehicle insulation       ########,.";WInsul
799 print #2, using"                    Cover panels             ########,.";WCover
800 print #2,
801 print #2, using"   Group 4:   Launch and Recovery  Gear                    ########,.";
801 WGear
802 print #2, using"                    Launch gear              ########,.";WLaunch
803 print #2, using"                    Landing gear             ########,.";WLG
804 print #2,
805 print #2, using"   Group 5:   Propulsion                                   ########,.";
805 WPropU
806 print #2, using"                    Rocket engines           ########,.";Wre
807 print #2, using"                    Ramjets                  ########,.";Wrj
808 print #2, using"                    Turboramjet              ########,.";Wtrj
809 print #2, using"                    Nonstructural fuel tank  ########,.";WFuCnt
810
811 print #2, using"                    Nonstructural Ox tank    ########,.";WOxCnt
812 print #2, using"                    Fuel tank insulation     ########,.";WInsFT
813 print #2, using"                    Ox tank insulation       ########,.";WInsOT
814 print #2, using"                    Fuel system              ########,.";WFuSys
815 print #2, using"                    Oxidizer system          ########,.";WOxSys
816 print #2, using"                    Pressurization system    ########,.";WPrSys
817 print #2, using"                    Inlets                   ########,.";WInlet
818 print #2,CHR$(12)
819 print #2,
820 print #2,      "   Weight statement for:";VName$;"                    Page 2"
821 print #2,
822 print #2, using"   Group 6:   Orientation Control System                   ########,.";
822 WOrnt
823 print #2, using"                    Engine gimbal system     ########,.";WGimbal
824 print #2, using"                    Attitude control system  ########,.";WACS
825 print #2, using"                    Aerodynamic controls     ########,.";WAero
826 print #2, using"                    Seperation system        ########,.";WSep
827 print #2, using"                    ACS tankage              ########,.";WACSTk
828 print #2,
829 print #2, using"   Group 8:   Power supply                                 ########,.";
829 WPwrSy
830 print #2, using"                    Electrical System        ########,.";WElect
831 print #2, using"                    Hydraulic/Pneumatic Sys  ########,.";WHyPnu
832 print #2,
833 print #2, using"   Group 10:  Avionics                                     ########,.";
833 WAvonc
834 print #2,
835 print #2, using"   Group 14:  Crew Provisions                              ########,.";
835 WCProv
836 print #2,      "                                                     -------------
837 print #2, using"                             Vehicle Dry Weight     ########,.";
837 WDry
```

```
print #2,
print #2, using"        Group 17:  Crew                                            ########,.";WCrew
print #2,
print #2, using"        Group 18:  Payload                                         ########,.";WPayLd
print #2,
print #2, using"        Group 21:  Residual Propellant                             ########,.";WResid
print #2, using"                       Trapped fuel          ########,.";WFTrap
print #2, using"                       Trapped Oxidizer      ########,.";WOTrap
print #2,      "                                                     -------------"
print #2, using"                                          Landing Weight      ########,.";WLand
print #2,
print #2, using"        Group 22:  Reserve Propellants                             ########,.";WResP
print #2, using"                       Fuel                  ########,.";WFResv
print #2, using"                       Oxidizer              ########,.";WOResv
print #2, using"                       ACS fuel              ########,.";WACSFu
print #2, using"                       ACS oxidizer          ########,.";WACSOx
print #2,      "                                                     -------------"
print #2, using"                                          Entry Weight        ########,.";WEntry
print #2,
print #2, using"        Group 23:  Inflight Losses                                 ########,.";WPLoss
print #2, using"                       Fuel                  ########,.";WFLoss
print #2, using"                       Oxidizer              ########,.";WOLoss
print #2,
print #2, using"        Group 25:  Main Propellants                                ########,.";WPropM
print #2, using"                       Fuel                  ########,.";WFuelM
print #2, using"                       Oxidizer              ########,.";WOxidM
print #2,      "                                                        -------------"
print #2, using"                                          Gross Weight     #########,.";WTO

close #2
return
'----------------------------------------------------------------------
'                        File Viewer Routine

FileViewer:
do
    sel$(1) = "F1.  File Viewer"
    sel$(2) = "F2.  File Converter"
    sel$(3) = "F3.  DOS Shell"
    sel$(4) = "F4.  Exit to Main Menu"
    Title$ = "File Viewer"
    gosub MenuDisplay
    on menu gosub SOut,Convert,DOSShell
```

```
loop until menu = 4
menu = 1
return
'---------------------------------------------------------------
'                        Screen Output Routine
sOut:
    locate 25,25
    input "Enter data filename: ";InFileN$
    if len(InFileN$) <> 0 then
        cls
        SHELL "PD1 "+InFileN$
    end if
return
'---------------------------------------------------------------
'                DOS Shell Routine
DOSShell:
    cls
    print "Type EXIT and hit <RETURN> to return to PDWAP."
    SHELL
return
'---------------------------------------------------------------
'           Converts Ver 2.0 data files to Ver 2.1 format
Convert:

    sel$(1) = ""
    Title$ = "File Converter"
    gosub MenuDisplay
    locate 11,5
    print "This routine will convert ver 2.0 data files to the ver 2.1 format."
    do
        errf = 0
    locate 15,1
    input "                 ver. 2.0 data filename                = ",InFileN$
    if InFileN$ = "" then return
    on error goto errchk
    open InFileN$ for input as #1
    loop until not errf
    do
        locate 16,1
        print "                 ver. 2.1 data filename (";InFileN$;
        input ") = ",OutFileN$
        if OutFileN$ = "" then OutFileN$ = "tempfile.dat"
        on error goto errchk
        open OutFileN$ for output as #2
    loop until not errf
    print
    input "Vehicle Name = ";VName$
    if VName$ = "" then VName$ = "No Name"
    print "On the next line you may type a Vehicle description up to 65 characters."
    input ,VDisc$
    VDisc$ = left$(VDisc$,65)
```

```
    If VDisc$ = "" then VDisc$ = "None"

    Line input# 1, a$

    if OutFileN$ = "tempfile.dat" then
        print #2, "File Name = ";InFileN$
    else
        print #2, "File Name = ";OutFileN$
    end if
    print #2, "Time = ";Time$
    print #2, "Date = ";Date$
    print #2, "Vehicle Name = ";VName$
    print #2, "Discription = ";VDisc$

    do
        Line input# 1,a$
        print #2, a$
    loop until eof(1)
    close

    if OutFileN$ = "tempfile.dat" then
        kill InFileN$
        name OutFileN$ as InFileN$
    end if


return

'------------------------------------------------------------------
MenuDisplay:
'          Provides Initial Screen For All Routines
'          Also Menu Routine When Menus are needed

CLS

locate 2,33,0
print "PDWAP ver 2.1"
locate 4,16
print "Preliminary Design and Weights Analysis Program"
ltitle = 39-len(Title$)/2
locate 6,ltitle
print Title$

if sel$(1) <> "" then

    for I = 1 to 4
        locate 10+2*I,27
        print sel$(I)
    next I

    locate 20,24
```

```
        print "Select number or use arrows"
        locate 21,30
        print "Then hit <ENTER>"

        menu = 2
        gosub f1

        key (1) on
        key (2) on
        key (3) on
        key (4) on
        key (11) on
        key (14) on

        on key(1) gosub f1
        on key(2) gosub f2
        on key(3) gosub f3
        on key(4) gosub f4
        on key(11) gosub UpArrow
        on key(14) gosub DownArrow

        a$ = " "
        while asc(a$)<>13
            a$=""
            while len(a$) = 0
                    a$=inkey$
            wend
            if a$ => "1" and a$ <= "4" then
                    on val(a$) gosub f1,f2,f3,f4
            elseif asc(a$)<>13 then
                    beep
            end if
        wend
end if

key (1) off
key (2) off
key (3) off
key (4) off
key (11) off
key (14) off

return
'----------------------------------------------------------------------
f1:
        old = menu
        menu = 1
        gosub HighLite
        return
'----------------------------------------------------------------------
f2:
```

```
    old = menu
    menu = 2
    gosub HighLite
    return
f3:
'===============================================================
    old = menu
    menu = 3
    gosub HighLite
    return
f4:
'===============================================================
    old = menu
    menu = 4
    gosub HighLite
    return
UpArrow:
'===============================================================
'       Reads Up Arrow and Adjusts Choice
    if menu = 1 then
        old = menu
        menu = 4
    else
        old = menu
        menu = menu - 1
    end if
    gosub HighLite
    return
DownArrow:
'===============================================================
'       Reads Down Arrow and Adjusts Choice
    if menu = 4 then
        old = menu
        menu = 1
    else
        old = menu
        menu = menu + 1
    end if
    gosub HighLite
    return
highlite:
'===============================================================
'               Highlites Selected Choice
    if old = menu then return
    locate 10+2*menu,24
    print "<<<";sel$(menu);">>>"
    locate 10+2*old,24
    print "   ";sel$(old);"   "
return

errchk:
```

```
'------------------------------------------------------------
'                 Error Handling Routine
    errf = -1
    beep
'Disk Error Checking
        if err = 53 then
                a$ = "File Not Found"
        elseif err = 55 then
                a$ = "File Already Open"
        elseif err = 57 then
                a$ = "Device I/O Error"
        elseif err = 61 then
                a$ = "Disk Full"
        elseif err = 62 then
                a$ = "Input Past End of File"
        elseif err = 64 then
                a$ = "Bad File Name"
        elseif err = 67 then
                a$ = "Too Many Files"
        elseif err = 70 then
                a$ = "Disk Write Protected"
        elseif err = 71 then
                a$ = "Disk Not Ready"
        elseif err = 72 then
                a$ = "Disk Media Error"
        elseif err = 75 then
                a$ = "Path/File Access Error"
        elseif err = 76 then
                a$ = "Path Not Found"
'Printer Error Checking
        elseif err = 24 then
                a$ = "Device Timeout"
        elseif err = 25 then
                a$ = "Device Fault"
        elseif err = 27 then
                a$ = "Out of Paper"
        else
                a$ = "Unknown Error/DOS Code ("+str$(err)+")"
        end if
    a$ = a$ + ": Hit Any Key to Continue"
    lena = 39 - len(a$)/2
    locate 25,lena
    print a$;
    a$=""
    while len(a$) = 0
        a$=inkey$
    wend
resume next
```

A	2	362	362	363	363	363	367	368	368	368	369
	369	370	370	370	370	375	375	376	376	399	399
	400	400	401	402	407	407	408	408	412	412	413
	413	413	414	418	419	420	421	422	422	424	431
	431	432	432	433	433	433	434	434	434	435	435
	435	437	437	438	438	440	440	441	441	445	445
	448	463	463	472	472	473	473	477	478	482	482
	483	483	494	494	495	495	495	496	497	497	498
	499	499	505	505	505	506	506	506	506	506	507
	507	509	509	510	510	511	512	512	513	513	515
	521	521	521	522	522	523	524	524	525	525	530
	530	530	533	533	533	540	540	546	546	547	547
	547	547	547	673	775						
AICapt	181	200	418	421	623						
AK	680	740									
ALM	2	689	727	740							
AMU	750	751									
ARatio	413	624									
Actr	398	622									
Atmos	386	706									
C1	686	749									
CHT	385										
CHt1	683										
CMT	384	387	390	392	392	393	395	743	744	746	
CMt1	685										
CalcWts	344	352									
Convert	880	903									
DH	385	626									
DM	384	627									
DOSShell	880	896									
DTDH	741										
Delta	736	746									
Display	345	757									
DownArrow	1002	1062									
E	494	496	750								
EX	729	730	732	733							
FDate$	605	766									
FMach	419	631									
FTime$	607	767									
FileRoot$	581	583	593	596							
FileViewer	46	872									
GFct	419	633									
GH	709	711	711	713	713	715	715	717	717	719	719
	721	721	727	729	732						
Gal	357	445	449	450	451	452	453				
GetData	343	569									
HB	2	689	727	729	732						
HK	740										
HighLite	1030	1036	1042	1048	1060	1072					
I	296	298	299	300	301	302	303	304	305	306	307

	308	309	310	311	312	313	314	315	316	317	318
	319	320	321	322	323	324	326	672	673	674	688
	689	689	689	689	689	690	977	978	979	980	
ICry	430	444	636								
IH	710	712	714	716	718	720	722	724	727	727	727
	728	729	730	730	732	732	733	740			
IMV	743										
IShape	361	493	638								
InFileN$	74	75	75	78	578	579	581	583	584	584	587
	765	888	889	891	913	914	916	920	937	953	954
Integral	436	637									
JSWA	2	689	728								
LAMle	58	177									
LBody	163	164	210	368	433	435	510	512	521	522	524
	628	769									
LInlet	212	418	629								
LRamp	214	421	630								
MenuDisplay	45	72	575	879	907	961					
NCrew	101	204	463	533	625						
NEng	355	447	448	448							
NIntake	167	181	290	418	418	421	421	424	668		
NTanks	266	445	445	446	449	450	451	453	453	656	
Nre	104	105	180	232	355	356	370	411	414	639	772
Nrj	111	112	167	180	234	355	356	370	406	408	408
	417	640	772								
Ntrj	118	119	167	180	236	355	356	370	383	403	403
	417	641	772								
OutFileN$	591	594	596	596	600	765	921	922	922	924	936
	939	952	954								
OutName$	84	85	85	85	86	184					
P	730	733	736	747							
PAlt	388	747	749								
PB	2	689	730	733							
PCham	643										
PD	46	52									
PD1	65	70									
PHigh	401	401	402	644							
PI	55	56	57	58	147	157	159	161	169	169	172
	174										
PLow	401	402	402	645							
PR	391	393	395	397							
PSL	681	736									
PT2	397	400	402	419							
PTO	388	397									
QMax	368	497	499	523	646						
QO	746										
RE	679										
RENO	751										
RTheta	738	743	744								
Rho	749	751									
RhoF	95	95	144	311							

RhoF$	94	95	95								
RhoH2	59	95									
RhoO2	60	145									
SBody	169	248	367	368	369	647	770				
SFair	362	648									
SFuTk	171	252	431	649							
SFuTk1	159	171									
SFuTk2	161	171									
SHorz	497	499	523	650	770						
SIGf	56	147	147	149	150						
SIGr	57	147	151								
SOut	880	886									
SOxTk	172	256	432	651							
STPS	174	258	375	376	652						
SVert	176	262	363	523	654	770					
SWing	173	174	176	264	494	495	495	497	499	523	655
	770	780	780	780							
TAlt	748	749									
TDel	505										
TFct	422										
TMB	2	689	727	730	732						
TMPA	740	741									
TMPB	727	730	737	741	748	750	750				
TOA$	124	124	126	298	299	300	301	306	307	308	309
	310	312	313	314	315	316	317	318	319	320	321
	322	323	324								
TRoot	177	274	494	496	660	769					
TSL	682	737									
TTOT	180	302	303	304	305	356	433	435	446	447	448
	780	780									
Theta	737	738									
Title$	44	61	571	878	906	971	973				
Tre	107	180	268	356	370	412	413	413	657	773	
Trj	114	180	270	356	370	407	408	658	773		
Ttrj	121	180	272	356	370	399	659	773			
TyTail	524	661									
UpArrow	1001	1050									
V	743	744	751								
V1	684										
VDisc$	135	137	137	138	138	192	616	618	763	930	931
	931	932	932	944							
VFuTk	144	147	278	357	434	437	440	662			
VName$	133	139	139	190	610	612	762	820	927	928	928
	943										
VOxTk	145	157	280	434	438	441	663				
WA	398	400	402								
WAATSb	46	341									
WACS	506	517	824								
WACSFu	507	508	509	514	852						
WACSOx	508	509	514	853							
WACSP	514	515	555								

WACSRe	515	552								
WACSTk	509	517	827							
WARef	398	670								
WAero	510	511	511	512	512	517	825			
WAvonc	530	537	833							
WBPump	447	454								
WBasic	367	368	368	460	791					
WBody	460	536	790							
WCProv	533	537	835							
WCont	540	543								
WCover	376	378	799							
WCrew	463	552	839							
WDistl	448	454								
WDist2	449	454								
WDrain	452	454								
WDry	536	540	543	777	837					
WElect	521	522	522	527	830					
WEmpty	543	552								
WEngMt	399	403	407	408	412	414	457			
WEntry	506	507	512	522	525	555	559	778	855	
WFCont	450	454								
WFLoss	477	479	858							
WFResv	472	474	850							
WFTrap	482	485	844							
WFair	362	502	788							
WFu	91	142	144	178						
WFuCnt	440	445	457	809						
WFuSys	433	454	458	814						
WFuelM	467	468	469	472	477	482	775	862		
WGear	536	549	801							
WGimbal	505	517	823							
WHorz	497	498	498	499	499	502	787			
WHyPnu	523	524	524	525	525	527	831			
WIDuct	418	425								
WInFuT	437	460	794							
WInOxT	438	460	795							
WInlet	425	427	458	817						
WInsFT	431	457	812							
WInsOT	432	458	813							
WInsul	375	378	798							
WLG	547	549	803							
WLand	179	282	495	499	547	552	555	664	778	847
WLaunch	546	549	802							
WO2	142	145	178							
WOLoss	478	479	859							
WOResv	473	474	851							
WOTrap	483	485	845							
WOrnt	517	536	822							
WOxCnt	441	457	811							
WOxSys	435	458	815							
WOxidM	468	469	473	478	483	775	863			

WPLoss	479	559	857								
WPMain	178	179	286	467	559	666					
WPayLd	130	179	284	552	665	777	841				
WPrSys	434	446	458	816							
WPropM	469	861									
WPropU	457	536	805								
WPwrSy	527	536	829								
WReFul	451	454									
WResP	474	555	849								
WResid	485	552	843								
WSeal	453	454									
WSecSt	369	460	792								
WSep	513	517	826								
WSpike	424	425									
WSurf	502	536	784								
WTO	141	179	288	494	497	506	510	513	521	525	530
	533	546	547	558	559	562	667	777	780	780	865
WTOi	558	562	562								
WTPS	378	536	797								
WThrSt	370	460	793								
WVRamp	421	425									
WVert	363	502	786								
WWing	494	502	785								
Wre	413	457	806								
Wrj	408	457	807								
Wtrj	400	457	808								
X	387	388	388								
XLF	368	494	495	669							
Xc	148	159	160								
Xf	147	148	149	150	159	163	169	169	174		
Xm	157	163	169	172	174						
Xr	151	161	161	163	169	169					
Xw	164	173	175								
a	197	198	199	201	202	203	205	206	207	208	209
	211	213	215	216	217	219	220	221	223	225	226
	227	228	229	230	231	233	235	237	238	239	240
	241	242	243	244	245	246	247	249	250	251	253
	254	255	257	259	261	263	265	267	269	271	273
	275	276	277	279	281	283	285	287	289	291	292
	293	294	297	298	299	300	301	302	303	304	305
	306	307	308	309	310	311	312	313	314	315	316
	317	318	319	320	321	322	323	324	325		
a$	29	30	31	89	90	93	97	100	103	106	110
	113	117	120	125	129	132	183	185	187	189	191
	193	194	195	196	197	198	199	200	201	202	203
	204	205	206	207	208	209	210	211	212	213	214
	215	216	217	218	219	220	221	222	223	224	225
	226	227	228	229	230	231	232	233	234	235	236
	237	238	239	240	241	242	243	244	245	246	247
	248	249	250	251	252	253	254	255	256	257	258
	259	260	261	262	263	264	265	266	267	268	269

	270	271	272	273	274	275	276	277	278	279	280
	281	282	283	284	285	286	287	288	289	290	291
	292	293	294	297	325	603	604	605	606	607	608
	609	612	614	615	618	620	621	622	623	624	625
	626	627	628	629	630	631	632	633	634	635	636
	637	638	639	640	641	642	643	644	645	646	647
	648	649	650	651	652	653	654	655	656	657	658
	659	660	661	662	663	664	665	666	667	668	669
	670	673	934	947	948	1004	1005	1006	1007	1008	1010
	1010	1011	1012	1092	1094	1096	1098	1100	1102	1104	1106
	1108	1110	1112	1114	1117	1119	1121	1123	1125	1125	1126
	1128	1129	1130	1131							
again$	66	67	347	348	348						
b	165	173	175	177	222	510	512	634	769	780	
bSt	175	260	494	495	524	653					
dot%	579	580	581								
errchk	77	586	599	915	923	1084					
errf	76	79	83	87	574	588	598	601	911	917	925
	1088										
f	98	142	218	467	468	632					
f1	988	997	1011	1027							
f2	998	1011	1033								
f3	999	1011	1038								
f4	1000	1011	1044								
fACS	508	642									
g	678	686	750								
h	166	224	368	635	769						
highlite	1074										
input#1	183	185	187	189	191	193	195	603	604	606	608
	614	620	621								
lena	1126	1127									
ltitle	971	972									
menu	46	47	880	881	882	987	1028	1029	1034	1035	1040
	1041	1046	1047	1053	1054	1055	1057	1058	1058	1065	1066
	1067	1069	1070	1070	1077	1078	1079				
old	1028	1034	1040	1046	1054	1057	1066	1069	1077	1080	1081
rOT	153										
rb	149	151	153	155	157	157	159	159	160	160	161
	161	161	161	165	166	169	169	169	174	174	174
re	740	740									
rf	150	159	159	160	160						
rt	155	172	172								
sel$	2	40	41	42	43	62	572	874	875	876	877
	905	975	979	1079	1081						
xf	160										
xr	174										

P

*U.S.Government Printing Office: 1990-748-056/24363