A Microcomputer Simulation of an Otto Aeroengine Powerplant System

Emmanuel Constantine Manoussakis

A Major Technical Report

in

the Faculty

of

Engineering and Computer Science

Presented in partial fulfillment of the requirements
for the degree of Master of Computer Science at
Concordia University
Montréal, Québec, Canada.

September 1983

ABSTRACT

A microcomputer simulation of an Otto aeroengine powerplant system.

Emmanuel C. Manoussakis

Hazardous situations, accidents, costly inflight training, complicated navigation and the inclination towards better quality of training have set the trend of novice pilot instruction in flight simulators. One of the main components of such a flight simulator is the aircraft powerplant. It embodies the aeroengine(s), the propeller(s), the control system and its simulation is the subject of this study. Linked with the Light Twin Aircraft Simulator development project at Concordia University with the cooperation of CAE Electronics Ltd. - this report aims at presenting a realistic simulation model of the entire powerplant of a light twin airplane developed in the course of the project.

Using economical and versatile microprocessor technology and

industry standard hardware, the simulation system described herein was put together. The heart of it, the software, is a detailed mathematical image of the system model with function generation covering particular nonlinearities. The study is organized as follows: system model definition; derivation and evaluation; simulation software and hardware design and testing; implementation and performance measurements.

A K N O W L E D G M E N T S

It is my privilege to thank the Staff of our Faculty and my colleagues. First and foremost my Graduate programme supervisor, Professor Radhakrishnan. His guidance and encouragement were fundamental in the development of this study in so many ways; I feel indebted to him in the true sense of the word. Professors Svoboda and McKinnon directly supervising the Light Twin Aircraft Simulator project gave me the opportunity to work on that component of the prototype; their help and directions in going about the subject were valuable and greatly appreciated. The help of Wolfgang Blach and Alex Marotidis about model derivation and structuring was very important in this study. The staff of the Industrial Control Lab were most helpful as well, especially in the terms of printing. Last but not least I wish to mention my brother George who is with the engineering staff of Hellenic Aerospace in Greece, for assistance in obtaining detailed technical information and literature as well as encouragement 'over the Atlantic'. I wish to thank everybody very much. It was a real pleasure for me to put together this piece of work with their help. The administration of the Faculty gave me this opportunity by accepting me in this graduate programme; certainly something that I am greatly indebted for.

στούς γονεῖς μου

TABLE OF CONTENTS.

LIST OF FIGURES.

VOLUME I

INTRODUCTION

Nikolaus Otto invented the four stroke internal combustion engine cycle in the dusk of the nineteenth century. Born in the German city of Holzhausen in 1832, this engineer founded the principles and set the pace for what was later to become the dominant source of autonomous kinetic energy generation . Two other compatriots of his, Rudolph Diesel and Felix Wankel varied the theme later to the diesel and rotary internal combustion engines but, nevertheless his invention has been in full application ever since. Trains, ships, automobiles and airplanes alike, use it to convert the thermal energy of hydrocarbon fuels into rotational kinetic energy. This form of energy accelerates the wheels of a car, drives the rotor of a generator and in this case powers the propellers of airplanes such as all those used before the jet turbine came into being and most of the lighter aircraft in use, today.

Although the mechanical principles are the same, the Otto aeroengines possess different characteristics, structural and operational in comparison to the automotive engines, chiefly in order

to comply to specific requirements for dependable inflight operation. They have been in use since the time of the first World War in commercial, military and recreational aviation in four cylinder configurations or twentyfour; on DC-6 cargo aircraft or a private Cessna 172. They are still in unlimited use in the light twin and single engine aircraft, having given their place in military and commercial applications to the Olympus; the CFM56, the RB211, the JW2000 or other jet engines of the gas turbine technology.

These machines form a integral part of the craft they power. It is a logical deduction then, that they - or their models - would be primary components of any flight simulator for light aircraft; indeed, a faithful representation of Otto Aeroengines is of high importance in such flight trainers. The engine system is certainly the only source of power in a flying airplane. Therefore in the case of synthetic flight training the student pilot must be presented with a highly accurate and broadended engine behaviour.

As stated, the scope of this study is to develop a realistic and compact simulation model of the total engine system of a light twin aircraft such as the Piper Seminole or the Beech Duchess or a similar airplane. Two identical Otto Aeroengines, two controlled propulsion systems and double control, fuel and electrical systems comprise the powerplant of such planes. A model developed for flight simulation purposes, may be useful in other aeromodelling studies or applications.

Introduction

The present electronic technology, mandatory manufacturing economization and the fashionable compactness of industrial goods suggest - if not dictate - the use of microprocessor based computing equipment. Certain limitations may arise in such arrangements but they would tend to be small in comparison to the overall advantages. The real time system used is assembled around a single board computer. It is based on the iAPX86 system and the corresponding family of peripheral devices.

Simulation of dynamic systems by means of digital computation may pose problems of 'stepping response' or disordered event sequencing; these are mostly attributable to slow processing. Use of a relatively high performance, iAPX86 system based computer reduces these problems due to its higher throughput rate and the efficient software tools that are designed for it, the PASCAL 86 and FORTRAN 86 compilers for example. In the case that this simulation is used as part of a flight simulator it would be attached to a critical band execution list. Thus, the code is designed to be as compact and small as possible up to the limit where the desired performance level shows signs of degradation; the criteria of performance would have to be established in respect to the ratio of the computing power available and the behaviour to be reproduced.

The general scheme of the simulation is briefly as follows; Input from the control system of the powerplant is entered to the computer after the appropriate transformations where it is processed by the model software. The output that is produced is checked against

performance specification data, converted to the proper format and transmitted to the output indicators; at the same time certain predefined functions take place according to specified requests from the user, such as system malfunctions, statistical information etc.

chapter 1

SYSTEMS DESCRIPTION

In this chapter the computing system and the powerplant systems are described in general; since the computing system is used as the vehicle for simulation, the operation of the system components is presented in a functionally descriptive form. Regarding the powerplant, the emphasis is placed on the major components of the aeroengines, whose operation is of importance to the overall level of fidelity of the simulation model; therefore, their operation is described in detail from the viewpoint of their influence on the total model.

1.1 THE COMPUTER SYSTEM.

The computer system is based on the iAPX86 system in the form of the iSBC 86/12A component board which is the central element of the system since it contains the system processor, memory, and the necessary I/O channels. The analogue to digital and digital to analogue conversion modules are added along with the necessary instrument drivers and power sources.

1.1.1 The iSBC 8612A computer in Real Time.

The iSBC 86/12A single board computer is designed for a wide range of industrial applications and efficient real time operation. It provides economical, self contained data processing power based on the 8086 microprocessor. There is a 32 kilobyte dual port dynamic RAM bank onboard and a system monitor in PROM. The system used includes the 8087 numeric data processor for floating point operations limiting the system clock frequency to 5 MHz. A twentyfour line parallel port and a serial port are available in the system with timer / counters and nine levels of vectored interrupt control (Fig. 1.1). It is MULTIBUS compatible and memory expandable. Word size may vary to 8, 16, 24 and 32 bits in instruction formats and 8, 16 bits of data words; the basic instruction cycle of this microcomputer is of the order of 1.2 μsec (400 nsec with the instruction in the stream queue). The manufacturer's specifications are listed briefly in Appendix D.

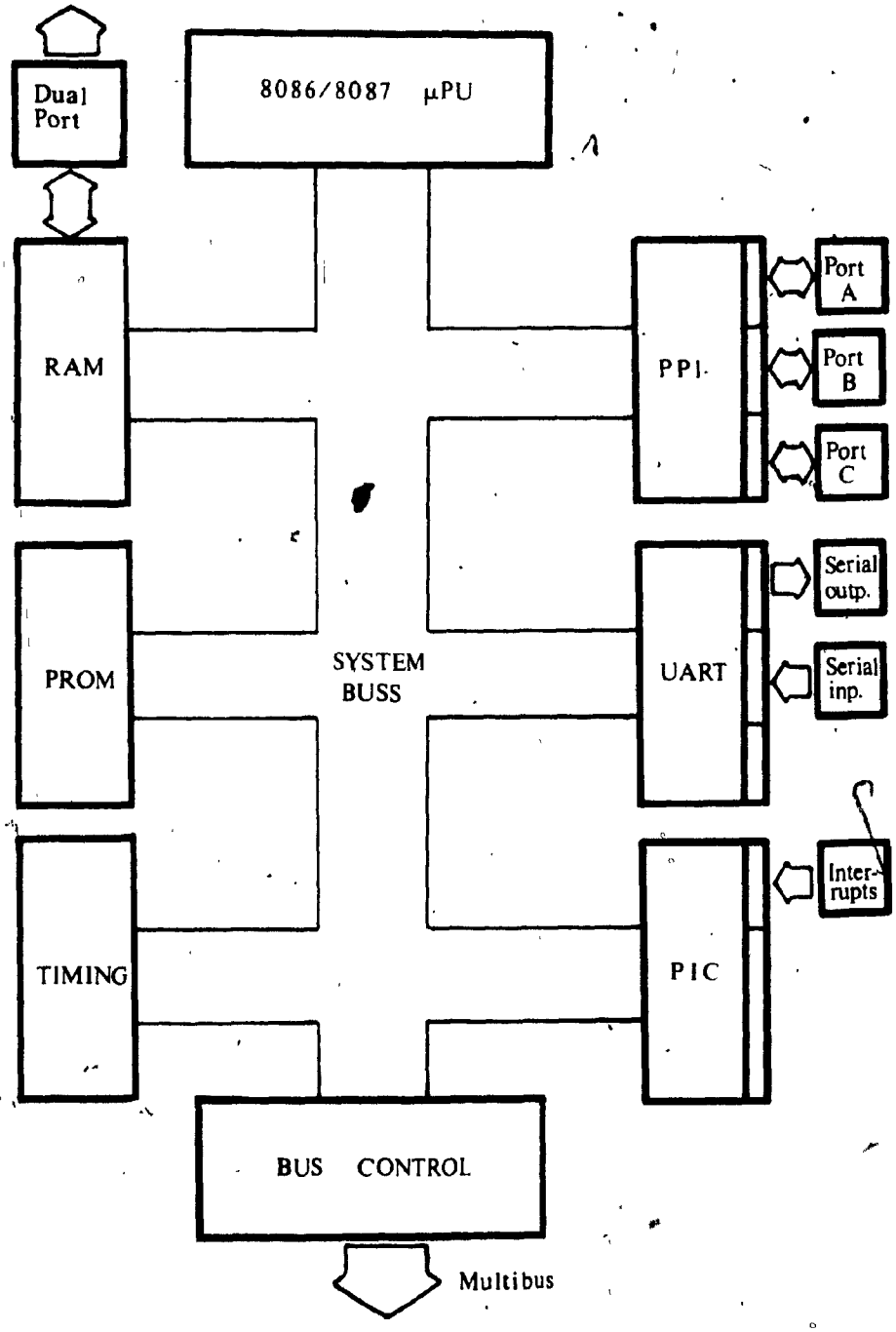


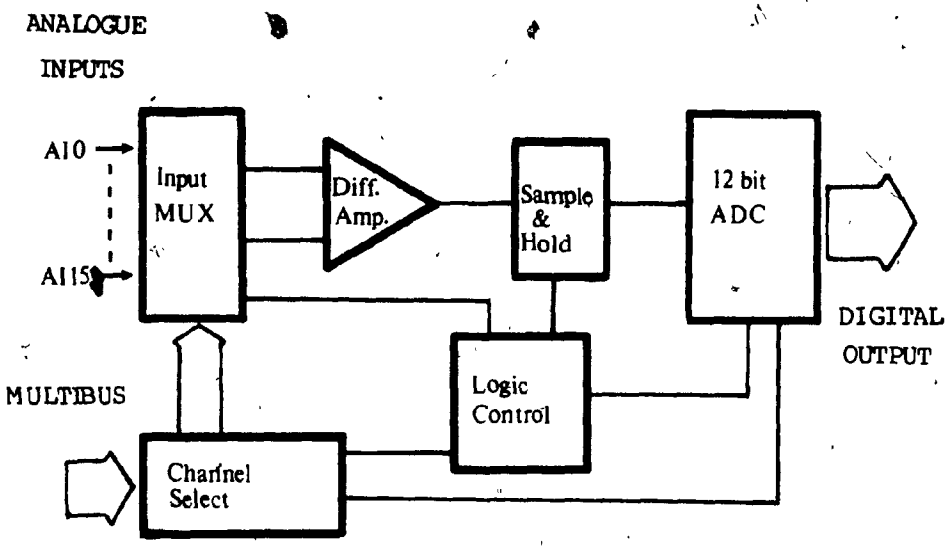
Fig. 1.1 The iSBC 86/12A I/O and memory block logic.

Since the simulation in discussion requires Real Time operation it would be desirable to use a real time oriented operating system; however, the cost of such a utility, e.g. RMX86, may be prohibitive. Alternatively, a simpler real time executive can be used that is based on scheduled priority and preemption of fixed nature operating at the module level. Preset parameters can be altered via the supervisory monitor if needed and scheduling is then reconfigured.

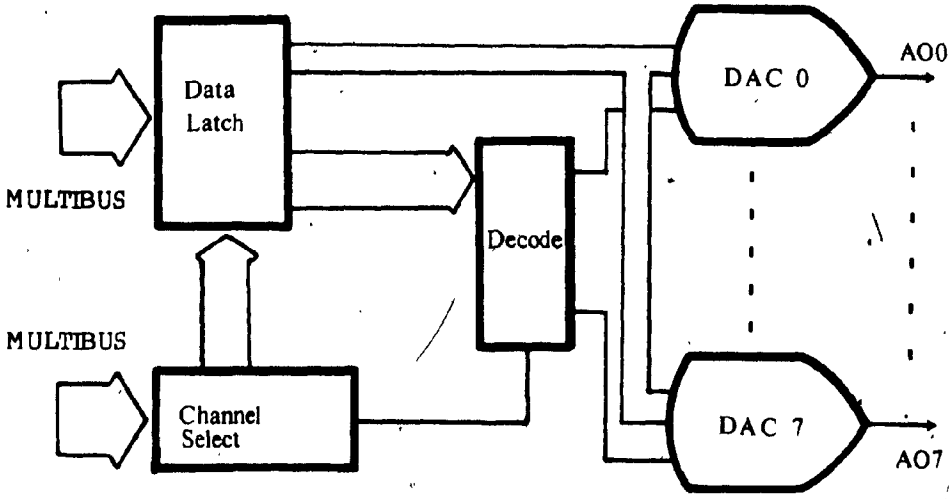
1.1.2 The Input / Output Interface.

Since the I/O controls and instruments are of analogue nature, the relevant conversions have to be effected before and after processing of data by the model software. Input from the throttle quadrant levers and the console controls is fed into a relatively low speed multiplexed (multichannel) analogue to digital converter; the analogue signal is polled at a 50 msec rate and a digital signal of the corresponding input is sent to the μPU in parallel form.

Conversely, digital output is distributed to the digital to analogue converters and latched; the converted information is then transmitted to the various powerplant monitoring instruments via the appropriate driver circuitry. Certain security mechanisms are implemented by means of interlocking, both in hardware and software in order that accidental error and hardware damage may be prevented. Figure 1.2 shows the conversion components of the I/O Interface.



16 CHANNEL ANALOGUE TO DIGITAL CONVERTER (MULTIPLEXED)



8 CHANNEL LATCHED ISOLATED DIGITAL TO ANALOGUE CONVERTER

Fig. 1.2 The A/D and D/A conversion logic.

Discrete inputs and outputs such as switches, luminous indicators and circuit breakers are buffered properly and grouped in byte or nibble format for faster manipulation. Largely, the peripheral system units perform conversion functions of relatively low accuracy and speed which in turn simplifies circuit wiring and does not account for higher component cost. All of the I/O interface is memory mapped on the system bus (Multibus); External communication takes place through the serial and/or parallel output ports of the microcomputer (Chapter four). Specification data for the analogue and discrete conversion and buffer components are listed in Appendices E, F and G.

1.1.3 The Real Time Computer System in block form.

The following block diagram (Fig. 1.3) shows a general view of the computer system; it has most of the characteristics of process control computer systems with the exception of safety related redundancy. The microcomputer in the form of the iSBC 8612/A component board, is configured to collect converted data from the analogue and discrete transducers and transmit information to the analogue and discrete output buffers of the I/O interface in order to drive the output devices viz. the motors of the indicators, the galvanometres etc. The model software with the Real Time executive monitor is compiled and linked externally in a development system or a host computer in the case of a flight simulator and then loaded in the memory of the microcomputer; in the case of a stand alone system, a disk subsystem may be attached on the Multibus via the necessary hardware and software components.

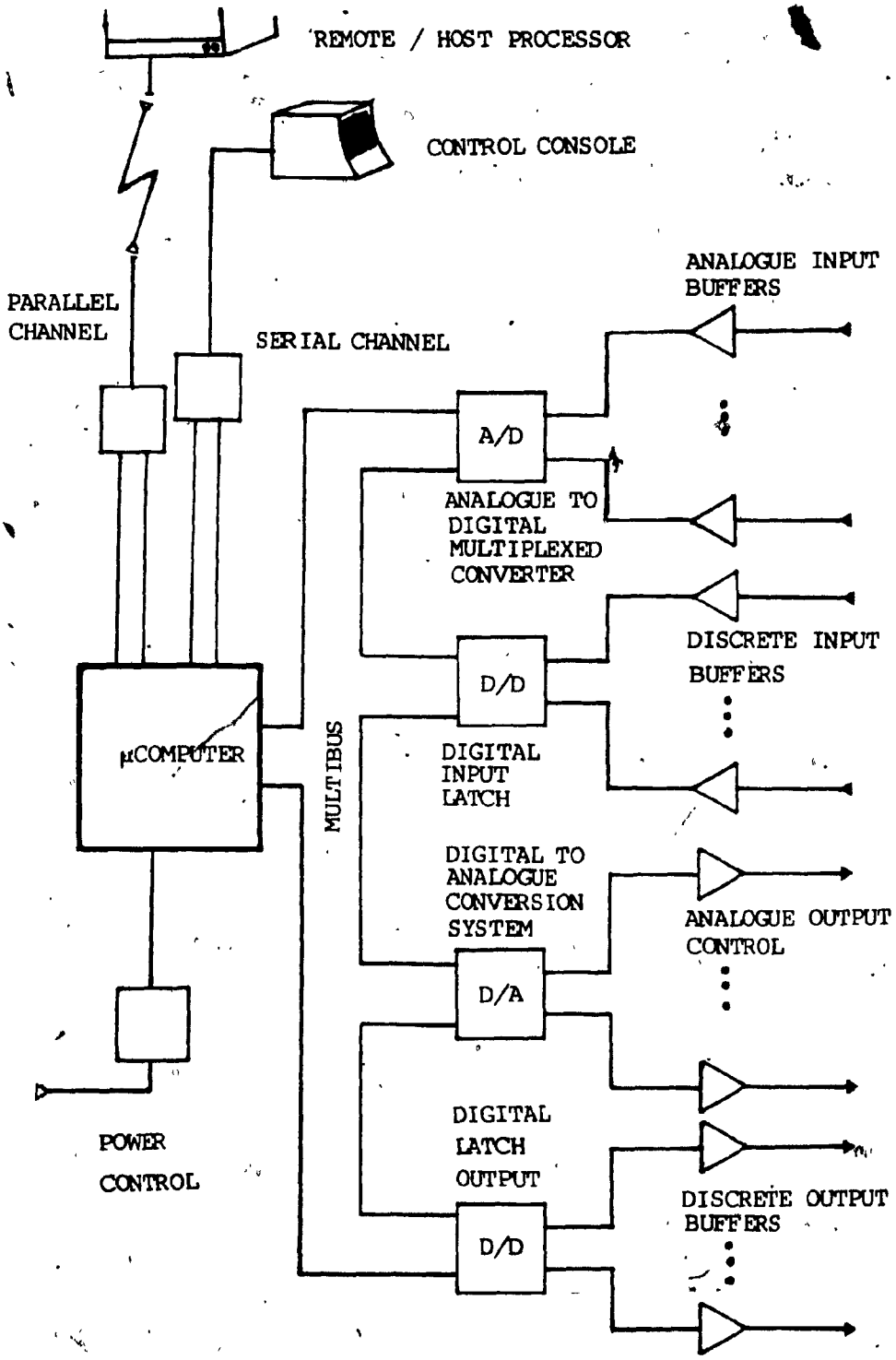


Fig. 1.3 The Real Time computer system block diagram.

The Real Time system monitor configures the I/O interface before run time assigning memory addresses to all input and output components as specified by a system I/O assignment file. Input data from the analogue signal sources is obtained in parallel format by selecting the A/D converter and placing a relative address on the Multibus which will select an analogue input channel of the converter. After a conversion time interval the data is latched at the output of the converter until a new channel address is selected. Since the data path of the Multibus is 16 bits wide and only 8 bits of analogue input data suffice, discrete data are read at the same time by the remaining 8 bits of the data bus; this scheme improves turnaround time and is very efficient when the iteration rate is lowered. Data from the discrete sources is buffered and latched to avoid bouncing and other undesirable conditions. Selection is made as with the A/D conversion system. The analogue and discrete data acquisition components form the input part of the I/O Interface. After aquisition these 16 bit data values are decomposed by the Data Control software and used by the Model software as input parameters after the necessary unit conversions and/or value biasing and scaling.

At the output of the microcomputer the stream is reversed. The output part of the I/O interface consists of the digital to analogue conversion components and the digital to discrete buffers and latches. A low cost single chip digital to analogue converter is used for each analogue output device in groups of 8 per circuit board with an optically isolated latch system; 8 bit values per output channel are stored in the latch circuitry. This allows for use of the remaining

lines of the data bus (Multibus) for information transfers to either discrete outputs or other D/A circuit boards in a manner similar to input channelling. Discrete outputs are latched and buffered as required by separate circuitry. These circuits are analysed in Chapter five. External to the I/O interface are the serial and parallel ports of the microcomputer. These channels may be configured for serial or parallel communication to a host computer, a terminal, a printer or a combination as required.

1.2 THE POWERPLANT SYSTEM.

The powerplant simulated is not that of a particular aeroengine model but rather a generic version using as guide the series 76 AVCO LYCOMING O-360 Otto aeroengine; this engine was chosen since it is used on many aircraft of the light twin category and can be considered as a representative member of its class. Performance data were examined and entered in the function generator with modifications considered necessary to maintain typical engine behaviour.

1.2.1 General system description.

The powerplant of a light twin aircraft is composed in general of two counter rotating, four stroke, four cylinder internal combustion aeroengines, mounted on the leading edge of each wing. A propeller is attached directly on the crankshaft of each engine - two blade in this case. These are full feathering, constant speed air dome propellers tracking in opposing directions; pitch is controlled by oil pressure and counterweight balancing. The fuel system of the powerplant consists of two wing tanks, the fuel pumps and fuel selector valves. Electricity is provided by a battery and two alternators, one on each engine. A typical light twin panel is shown in Fig. 1.4.

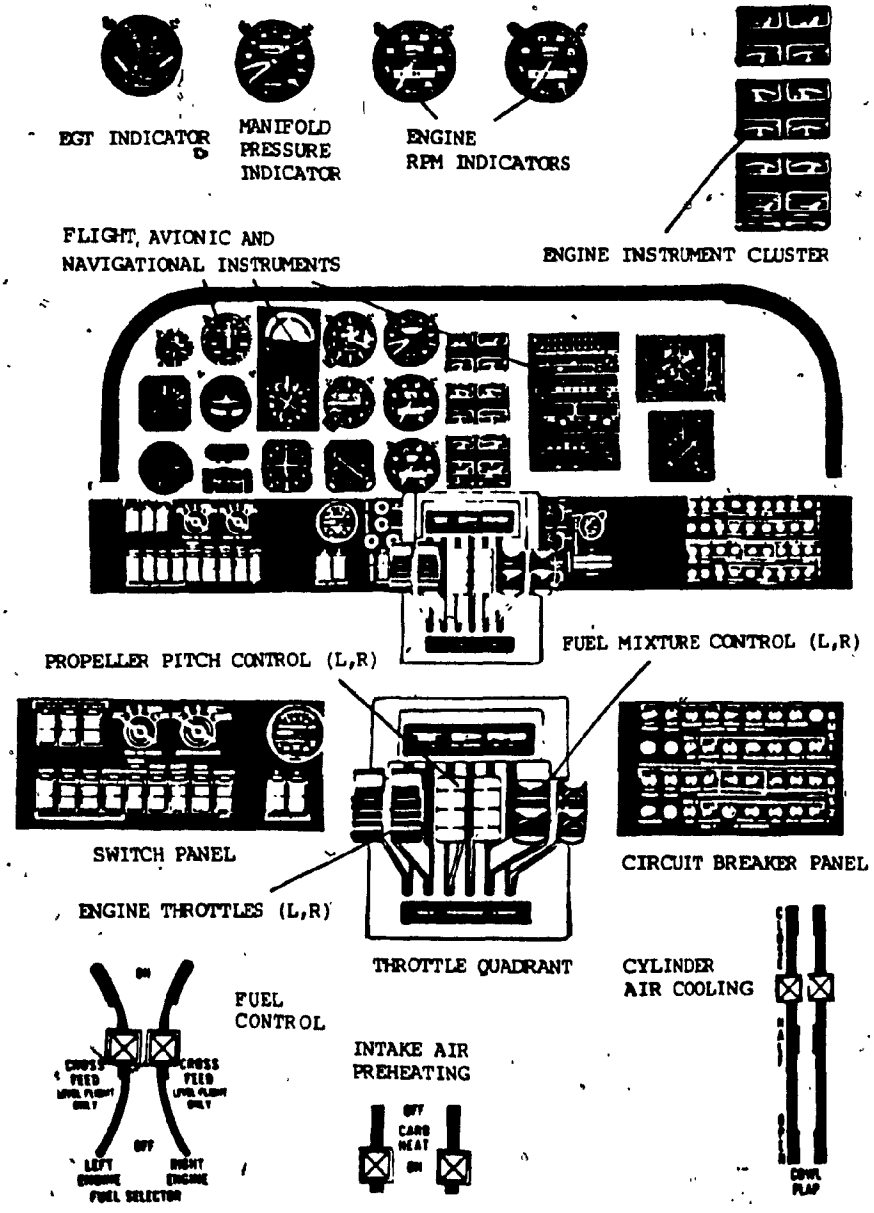


Fig. 1.4 Light twin aircraft instrument panel and controls.

1.2.2 System control input.

Control inputs to the system are installed in the instrument panel and pedestal in the aircraft cabin (Fig. 1.4) and they can be described as follows;

i) **The throttle levers** control the amount of air and fuel that enters the manifold of each engine and therefore their power output.

ii) **The pitch levers** control the oil pressure that reaches the propeller domes in order that the blade pitch angle be varied to the required settings.

iii) **The fuel / air mixture levers.** Since air density changes with altitude, the fuel / air ratio has to be varied with it for different flight conditions; these levers affect that ratio.

iv) **The carburator heat** and **cowl flap controls** alter the air temperature in the carburator and the engine cooling.

v) **The panel switches;** viz. the ignition/start switches, the auxiliary fuel pump switches, the battery and alternator field switches, etc.

vi) **The fuel selector valves** control fuel flow to the engines. Each valve is a three position control that may switch fuel from its tank, off, on to its engine or, on to both engines making fuel

available from both tanks to either engine or both.

vii) **The system circuit breakers** are used to prevent electrical overload conditions in the system; often, they are also used as switches.

1.2.3 System status output.

The powerplant status in light twin aircraft is monitored usually via the following gauges/indicators:

i) **The engine tachometres** show each engine's number of revolutions per minute and the accumulated 'hours' of engine operation after calibration.

ii) **The manifold pressure indicator** is usually a dual indicator where the absolute manifold air pressure of each engine is displayed.

iii) **The engine instrument cluster** is a collection of galvanometres where the following parameters are displayed:

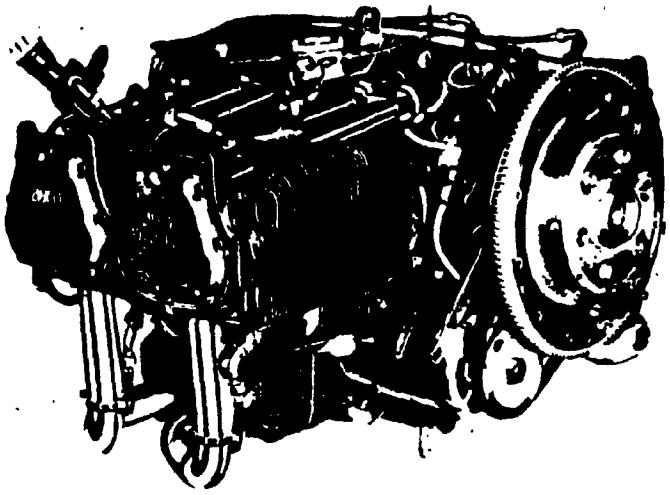
1. Left and right tank fuel quantity.
2. Left and right pump fuel pressure.
3. Left and right engine oil pressure.
4. Left and right engine oil temperature.
5. Left and right engine cylinder head temperature.
6. Left and right alternator load condition.
7. Left and right alternator under/over voltage condition.

iv) **The exhaust gas temperature indicator** is a dual indicator where the temperature of the exhaust gases is shown for fuel control purposes.

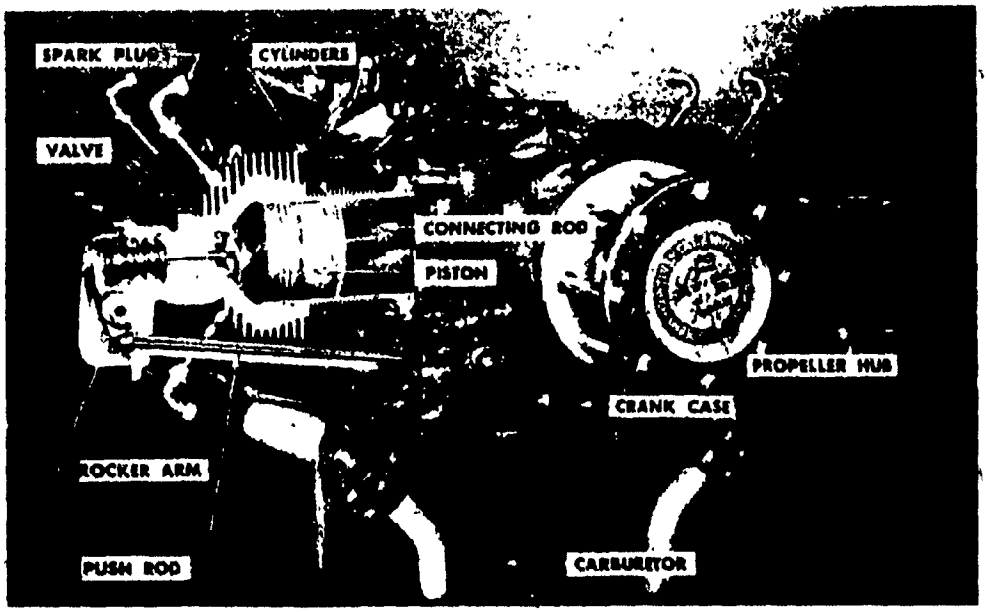
1.2.4 The aeroengine.

The aeroengine in discussion is rated at 180 Hp at 2700 RPM. It is air cooled and can be operated safely in pressure altitudes from sea level to approximately 20000 feet in most weather conditions. Mechanically it is a four cylinder, direct drive, horizontally opposed arrangement. It starts via an electric motor-starter. Ignition is provided by a double high voltage circuit controlled from the ignition/start switch on the aircraft panel. Electrical power is generated by a low voltage, high current alternator and rectified into DC power for use in the various electrical systems. The engine is lubricated constantly by pressurised oil circulated through a heat exchanger when overheated.

Fuel reaches the fuel control unit through a series of pumps and valves, where it is metered and atomised in the carburator; the mixture then is induced into the cylinders and ignited at the specified time intervals of the Otto cycle.



THE O-360 AVCO LYCOMING AEROENGINE



AEROENGINE NOMENCLATURE

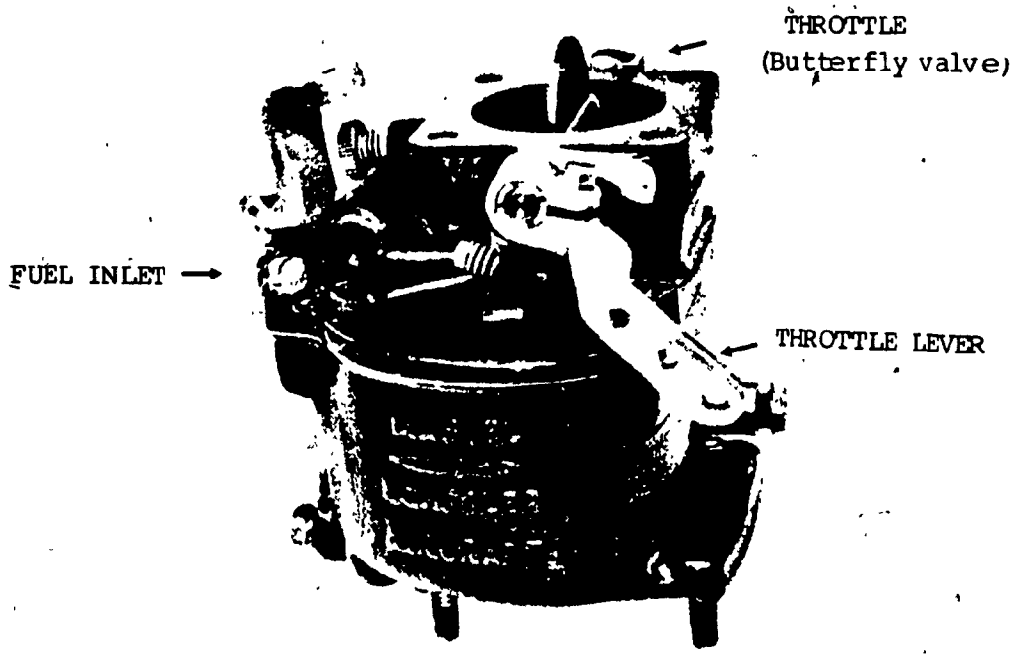
Fig. 1.5 The O-360 AVCO-LYCOMING Aeroengine.

A diagram of the engine is illustrated in Fig. 1.5 and the manufacturer's data are presented in Appendix C. This type of engine does not use a supercharger since it is designed primarily for low altitude operation. Significant to the simulation model are the following engine subsystems whose operation is analysed in Chapter two for model derivation purposes:

i) **The carburator / fuel control unit.** This subsystem is the most vital in the operation of the engine; here air suction is adjusted through the throttle lever and fuel is metered proportionately into the air stream according to the setting of the mixture lever, amounting to power output and efficiency regulation. An automatic air choke mechanism is assumed but not included in this model. (Fig. 1.6).

ii) **The engine manifold.** This component distributes the fuel mixture to the cylinders of the engine after regulation through the throttle control (butterfly valve); the air pressure in it is an indication of engine status and is displayed to the operator.

iii) **The cylinder / piston assembly.** The energy conversion is effected in this part of the engine generating rotational kinetic energy and heat.



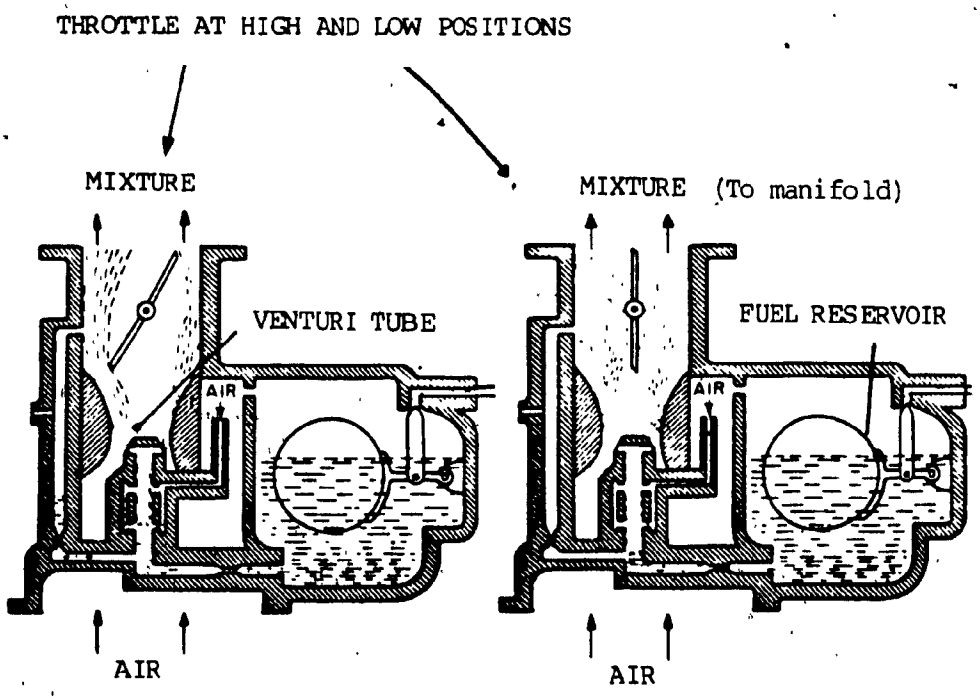


Fig. 1.6 The Carburator / Fuel control unit.

iv) **The crankshaft assembly.** Although this is a passive component, its mechanical compliance, inertia and viscous friction affect the engine's performance.

v) **The ignition system.** This is a double system that provides a pulsed high voltage which is distributed to the igniters (sparkplugs) on each cylinder at specified times.

vi) **The oil system.** As with most internal combustion engines, the aeroengine in discussion embodies a lubrication system that affects its performance significantly; (Fig. 1.7). Oil is injected onto the moving parts that are subjected to mechanical stresses in order that frictional losses and heat are reduced; in doing so the oil gains heat. After collection in the engine's sump it is filtered and cooled through a heat exchanger and it is then pressurised and recirculated. Pressurised oil is used for propeller control as well.

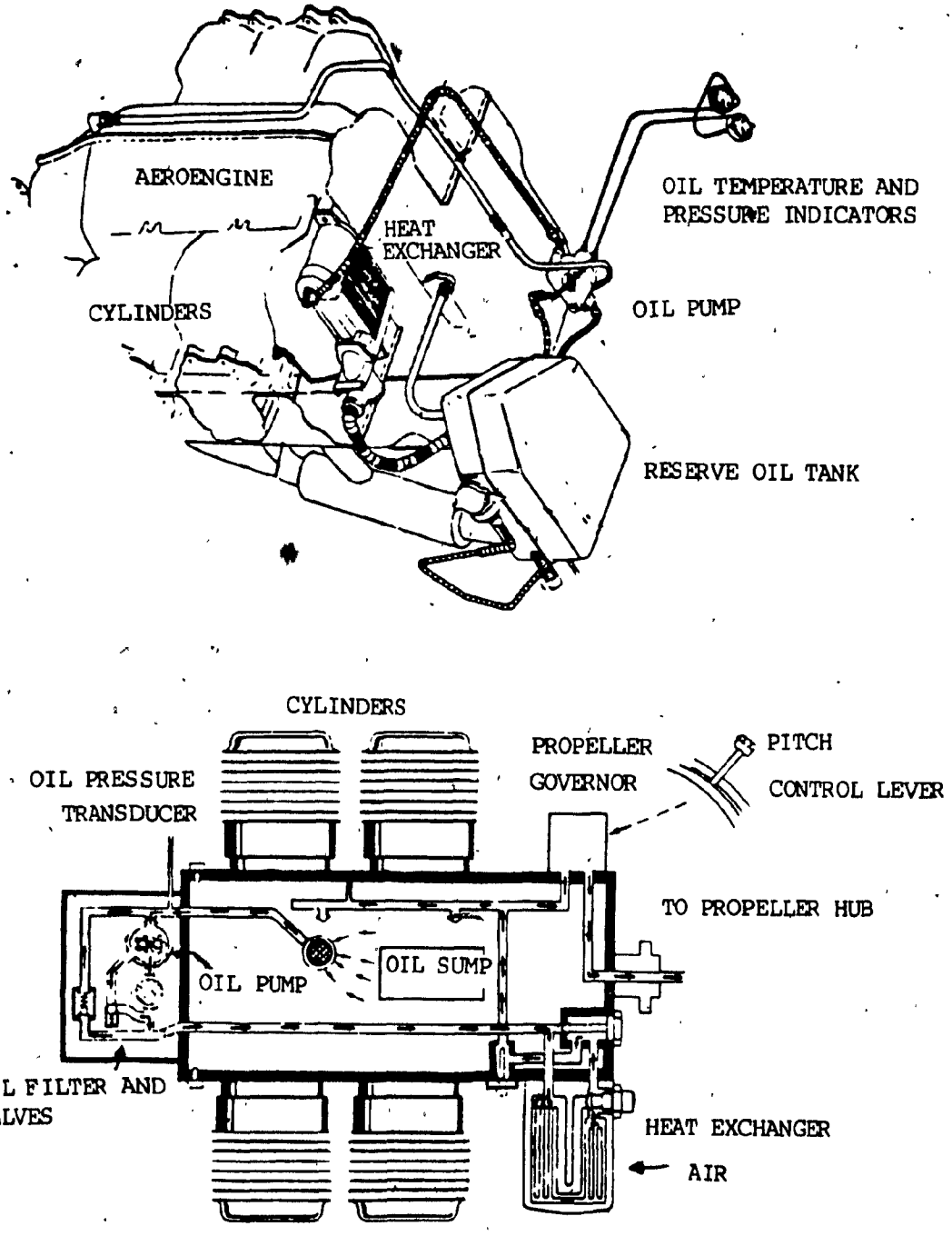


Fig. 1.7 The Lubrication system.

1.2.5 The electrical system.

The electrical system of a light twin aircraft includes two alternator and one battery power busses. Electrical power is a low voltage (24-32 V), DC current which is used to operate the avionic equipment and the other onboard electrical devices. The power output of both alternators is regulated in order to maintain a stable nominal line voltage (28V) on each power bus. (Fig. 1.8)

1.2.6 The fuel system.

The fuel system supplies the demanded fuel quantities to the aeroengines. It consists of the two wing tanks, the fuel lines, the selector valves, the auxiliary (electric) and the main (mechanical) fuel pumps and the fuel priming mechanisms for engine start up. Such a fuel system is depicted in Fig. 1.9.

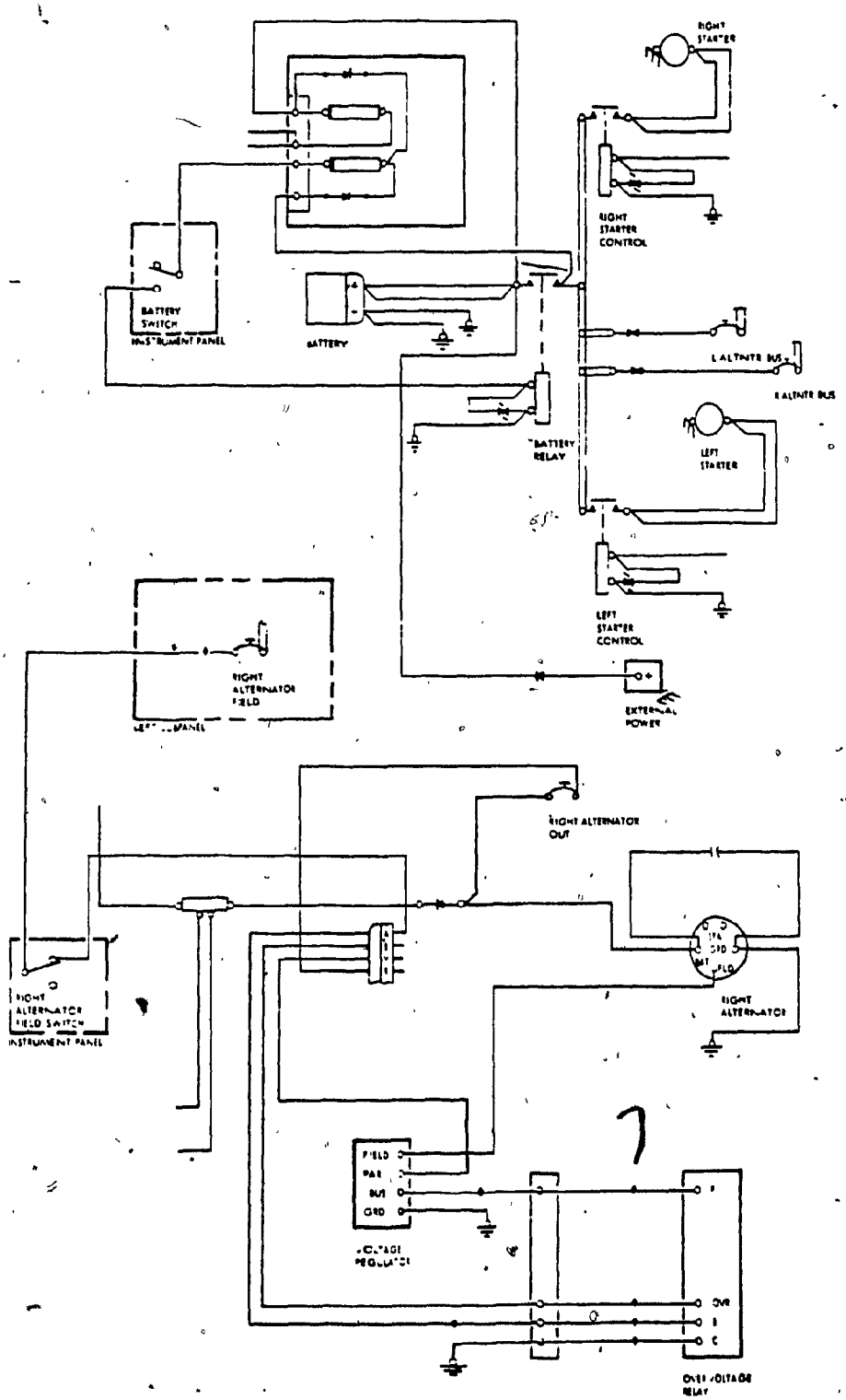


Fig. 1.8 The Electrical power system.

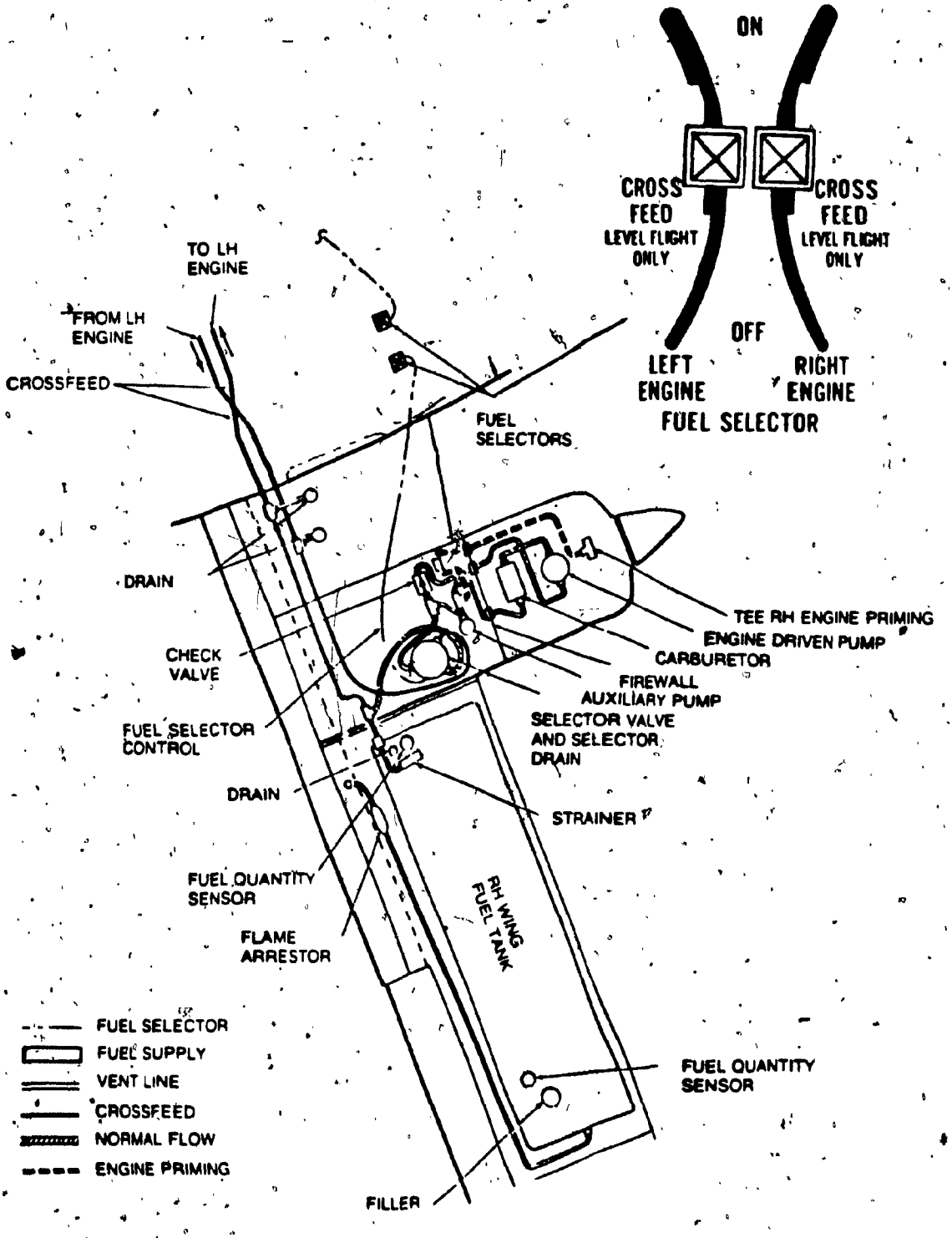
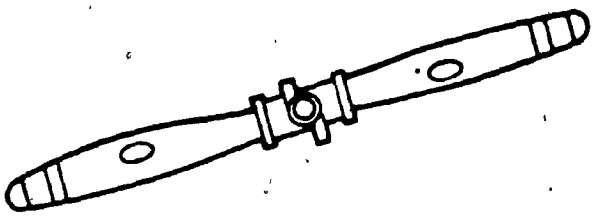


Fig. 1.9 The Fuel system.

1.2.7 The propulsion system.

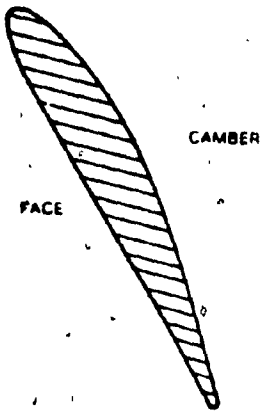
The propulsion system can be thought of as part of the aeroengine assembly since the propellers are either mounted directly on the crankshaft or are connected to it via a reduction gear train. In this case no geared transmission is used. Each propeller is a two blade, constant speed, full feathering unit. They have variable pitch which is at the low setting 12 degrees, at high 17 to 20 degrees and feathered 81 degrees. Synchronising propeller governors regulate engine speed by varying the propellers' angle of attack through the airstream to match propeller demand torque to engine brake torque in response to varying flight conditions. This type of propeller governor allows for exact matching of engine RPM at all altitudes including climb, cruise and descent conditions resulting in turn, in lower noise levels and vibrations. Feathering is accomplished via a pressure storage accumulator at the pilot's discretion. If the propeller is unfeathered in flight it windmills and may easily restart the engine.

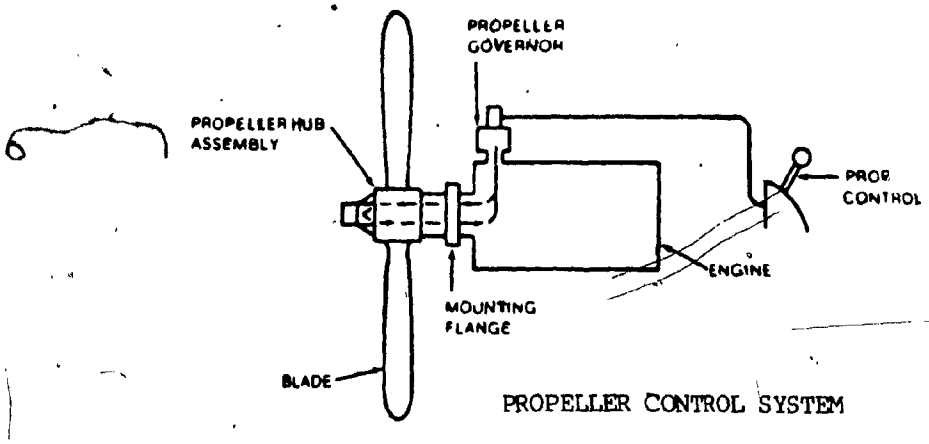
All of the above functions of the propulsion system are of importance in the simulation since they affect significantly the conversion of the rotational kinetic energy into thrust. Their effects are analysed in Chapter two. The following illustration shows the components involved. (Fig. 1.10)



LEADING EDGE

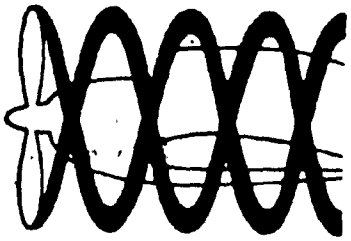
TWO BLADE PROPELLER



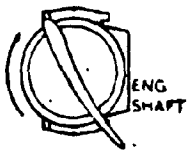


PROPELLER CONTROL SYSTEM

PROPELLER CROSS SECTION

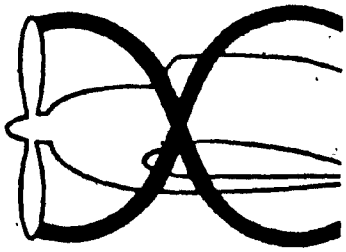


a. LOW PITCH



DIRECTION OF FLIGHT

PROPELLER OPERATION AND AIRSTREAM



b. HIGH PITCH

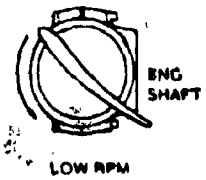


Fig. 1.10 The Propulsion system.

chapter 2

MODEL DERIVATION

The mathematical model of the Powerplant system is derived step by step in this chapter in the form of differential equations. Most of these equations describe the partial functions of the aeroengine which is regarded as a dynamic system responding to control and disturbance inputs. After defining the system model's primary functions in concise algebraic terms, the simulation software model is derived in the next Chapter in modular form. Function generation is used for compliance to specific performance data where necessary. Since the engine system is duplicated, the model derivation is in the following order: the model for one of the aeroegnines is defined along with the model of one propulsion system. Ancillary and control system model derivation follows, embodying all the logical functions and controls of the powerplant system.

It must be noted that the two engines and propellers are identical except that the engines crank in opposing rotations and the propellers track in opposite directions; this is established so that the moments of the system are balanced and stability is preserved. The mechanical functions of the system are not analysed in this study. Only the systems of importance for simulation purposes are examined in

detail. Système International (SI) units are used throughout the model derivation and simulation. An analogue equivalent model is presented in Appendix B for comparison purposes; that model was used for a partial evaluation of the simulation.

The mathematical model derivation is presented along with the basic laws and theorems that govern the simulated systems. Chiefly, most critical functions are carried out in the carburator of the aeroengine, hence this system model is very important. The powertrain model is critical in performance as well, since power generation is represented by it. Together they are the principal components of the power generation model. The power dissipation model interacts directly with the latter and therefore it affects the system significantly. The control system of the powerplant which is a combination of levers, switches and indicators forms the final major system that is simulated. Mathematical derivation of secondary systems of logical nature is established in Boolean form. Figure 2.1a shows the general simulation model which is based on the equations of this Chapter.

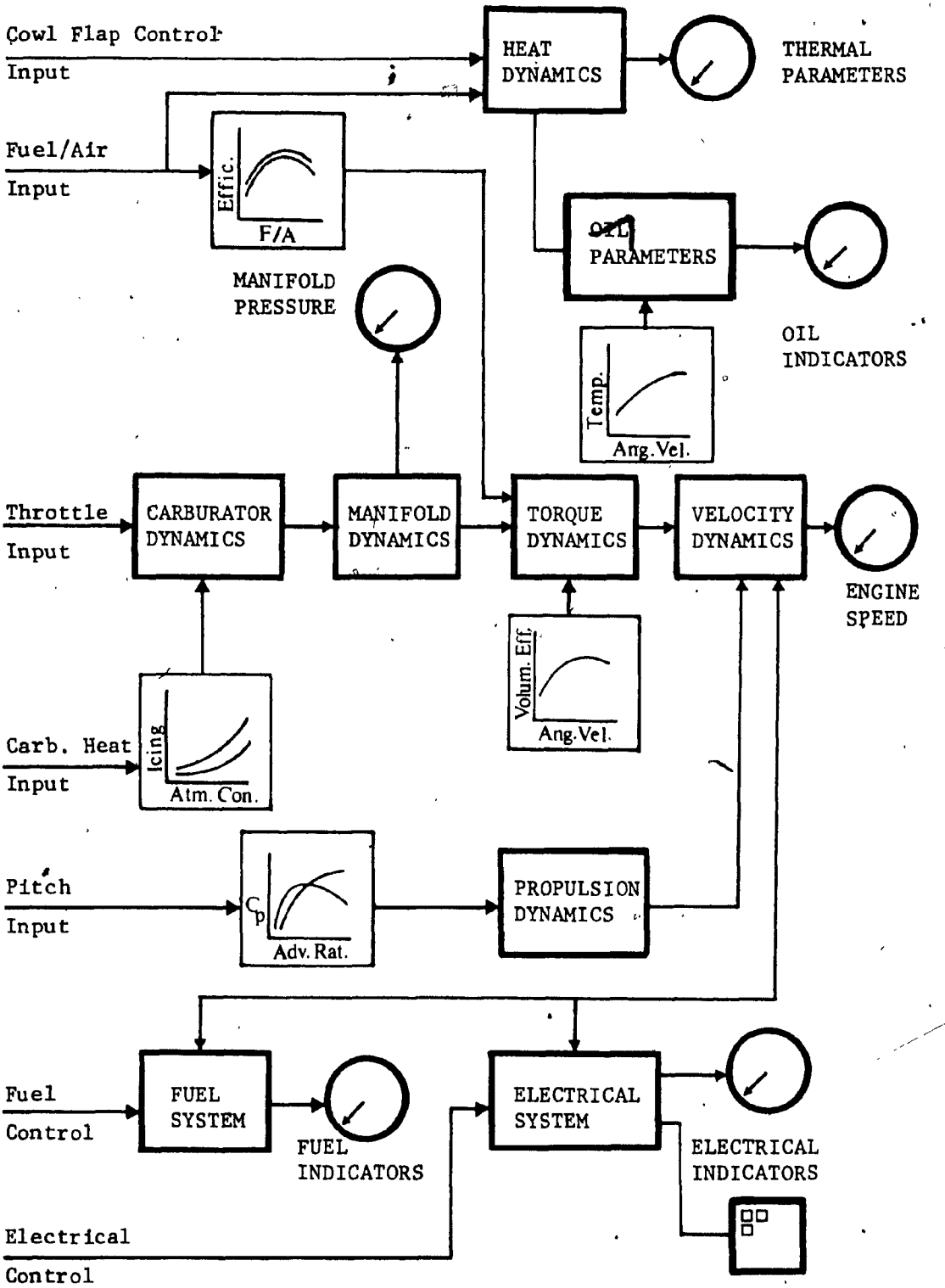


Fig. 2.1a The general simulated system.

2.1 THE AEROENGINE MODEL.

The powerplant system is a highly non linear first order system. Its transfer function is shown in Fig. 2.1b. Control inputs to this system are primarily the throttle and fuel mixture levers; the propeller demand torque is treated as a disturbance input. Output is in the form of angular velocity of the crankshaft and heat dissipated into the environment. The alternator loading torque is considered as another disturbance input. The aeroengine - or power generation - model is composed of the carburator/fuel control model, the manifold model and the cylinder/crankshaft model. These partial models are grouped together with the propulsion and the ancillary system models to form the complete Powerplant system model used in the derivation of the software modules in Chapter three.

2.1.1 The carburator model.

The carburator's principle of operation (Fig. 2.2) as a critical system function is based on Bernoulli's law as applied to the flow of compressible fluids for adiabatic thermodynamic conditions. Briefly, it states that in any moving fluid along each streamline, the sum of the velocity head, the pressure head, and some elevation above a horizontal plane of reference is constant. (Ref. 2.1)

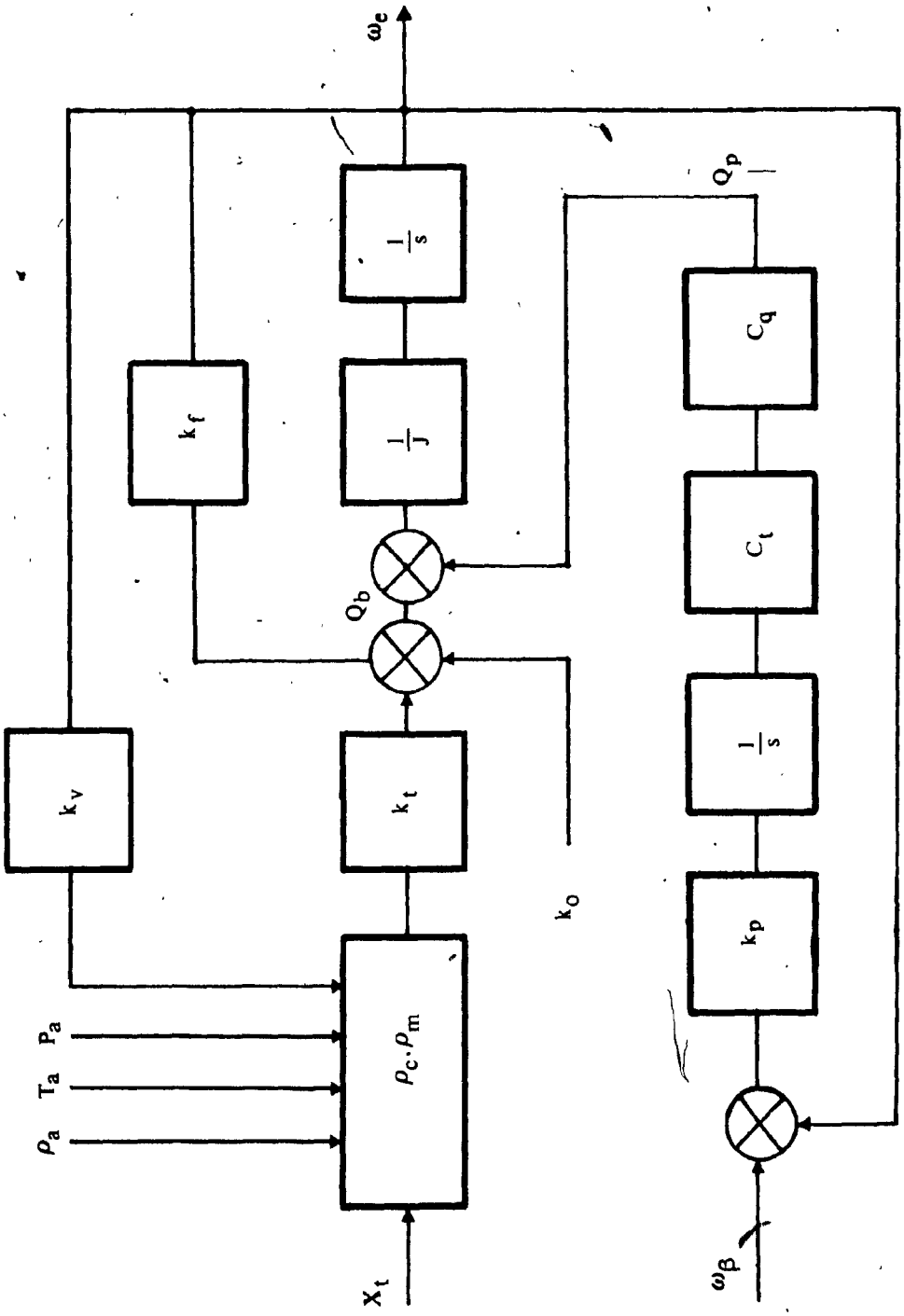


Fig. 2.1b The aeroengine Transfer function.

This law of Physics can be expressed as follows: (Fig. 2.2b).

$$\int_{P_1}^{P_2} \frac{dP}{w} + \int_{V_1}^{V_2} \frac{V\,dV}{g} + \int_{z_1}^{z_2} dz + \int_{1}^{2} dH_l = 0$$

[2.1]

where H refers to the total head lost and w to the specific gravity of the fluid; P refers to the fluid's pressure, V to velocity, z to elevation and g to the gravitational acceleration. The first term cannot be integrated until the relationship between w and p is established. For adiabatic conditions the general gas law can be expressed as:

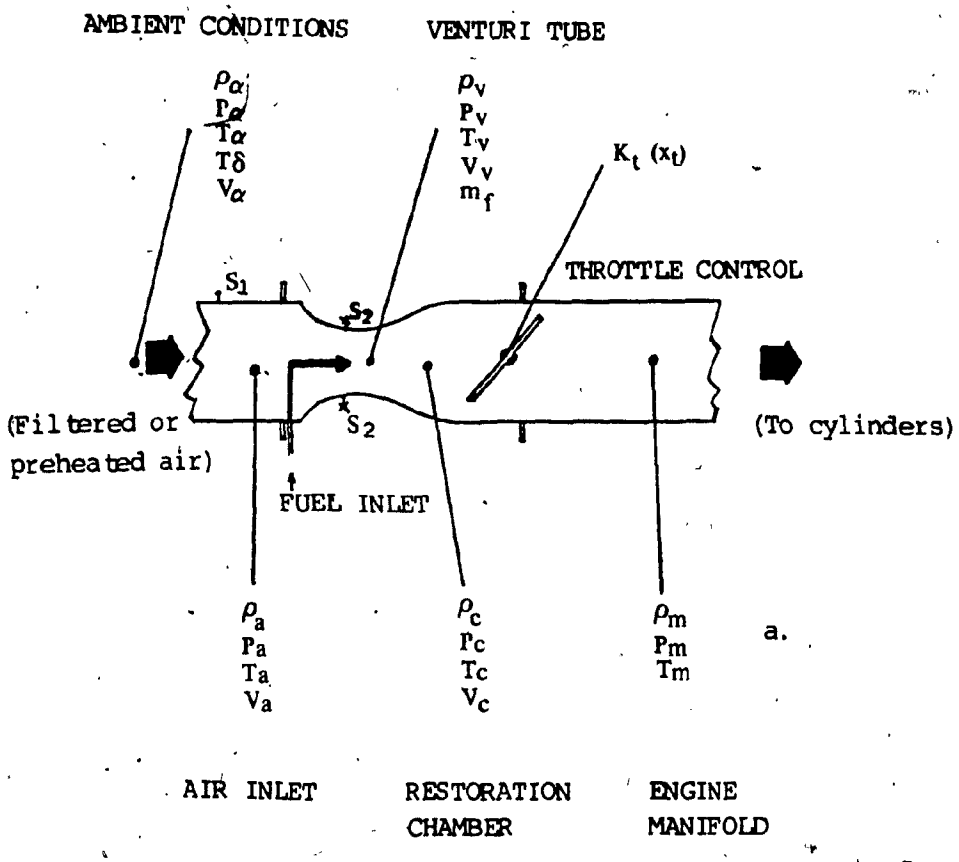
$$w = w_1 \left(\frac{P}{P_1}\right)^{\frac{1}{k}}$$

[2.2]

where k is the adiabatic exponent (c_p / c_v). Integrating the first term separately we have:

$$\int_{p}^{p} \frac{dP}{w\left(\frac{P}{P_1}\right)^{\frac{1}{k}}} = \left(\frac{k}{1-k}\right) \frac{P_1}{w} \left[\left(\frac{P_2}{P_1}\right)^{\frac{k-1}{k}} - 1\right]$$

[2.3]



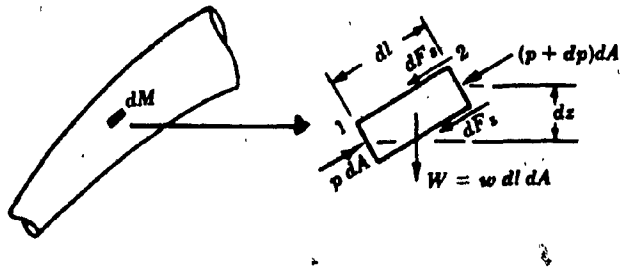


Fig. 2.2 Carburator and Manifold diagram.

then the Bernoulli equation becomes:

$$\left[\frac{k}{k-1}\frac{P_1}{w_1}+\frac{V_1^2}{2g}+z\right]-H_l=\frac{k}{k-1}\frac{P}{w}\left(\frac{P}{w}\right)^{\frac{k-1}{k}}+\frac{V_2^2}{2g}+z_2$$

[2.4]

By the equation of continuity and the general gas law for adiabatic conditions we have an expression where air density is the unknown variable:

$$\rho=\frac{w}{g}$$

$$\rho_1 S_1 V_1=\rho_2 S_2 V_2 \qquad [2.5]$$

The manifold air (mixture) velocity V_2 is a function of the engine's angular velocity and the engine's cylindrical volume as per:

$$V_2=f(k_v,\omega_e)$$

$$V_1=\left(\frac{P_2}{P_1}\right)^{\frac{1}{k}}\left(\frac{S_2}{S_1}\right)V_2 \qquad [2.6]$$

The Bernoulli equation can then be expressed as:

$$\left(\frac{k}{k-1}\right)\frac{P_1}{w_1}+\left(\frac{P_2}{P_1}\right)^{\frac{2}{k}}\left(\frac{S_2}{S_1}\right)^2\frac{V_2^2}{2}+z_1-H_l=\frac{k}{k-1}\,\frac{P_1}{w_1}\left(\frac{P_2}{P_1}\right)^{\frac{k-1}{k}}+\frac{V_2^2}{2g}+z_2$$

[2.7]

The carburator operation is based on these equations. Fuel metering and atomization is governed by the Venturi principle. As such, the carburator function can be expressed in terms of ambient air temperature, air pressure and density based on the Bernoulli equation. First, the air temperature in the carburator inlet before the Venturi tube can be expressed as equal to the ambient air temperature and the effect of the carburator heating when it is engaged. The latter varies with the position of the carburator heat levers.

$$T_a = T_\alpha + k_{ch}(T_e - T_\alpha)$$

[2.8]

the air density in the carburator inlet can be expressed as that of the ambient air varying with the internal temperature changes:

$$\rho_a = \frac{\rho_\alpha T_\alpha}{T_a}$$

[2.9]

The air pressure in the inlet is almost equal to the ambient air pressure, assuming a 'perfect' air filter. The velocity of the air at the same point can be expressed as a function of air density and manifold pressure (suction);

$$V_a = \frac{\rho_m V_m S_1}{\rho_a S_2} \qquad [2.10]$$

In order to derive an expression for the air temperature inside the Venturi tube the density of the air at that point has to be known; applying the equation of continuity it can be expressed as:

$$\rho_v = \frac{\rho_a V_a S_1}{V_v S_2} \qquad [2.11]$$

and by the Bernoulli law it becomes:

$$f(\rho_v) = \frac{P_a}{\rho_a} \frac{k}{k-1} + \left(\frac{\rho_v S_1}{\rho_a S_2}\right)^2 \frac{V_a^2}{2} - \frac{P_a}{\rho_a} \frac{k}{k-1} \left(\frac{\rho_v}{\rho_a}\right)^{k-1} - \frac{V_2^2}{2} \qquad [2.12]$$

since this form is rather complex to reduce, it can be evaluated by using the Newton approximation method (Ref. 2.4) as follows, provided there is sufficient computing capacity;

$$\rho_v = \rho_{v_o} - \frac{f(\rho_{v_o})}{f'(\rho_{v_o})} \qquad [2.13]$$

where $f'(\rho_{v_o})$ represents the first derivative of the function. The air temperature at the Venturi narrowing will then be: (Ref. 2.2)

$$T_v = T_a \left(\frac{\rho_v}{\rho_a}\right)^{k-1} \qquad [2.14]$$

this quantity when combined with the Dew point of the air may determine the presence of a dangerous inflight condition known as carburator icing that may render the engine unusable. (Fig. 2.3b). Ice can be formed from the water vapor in the air, around the Venturi tube and the throttle valve (Fig.2.3a) to the point where the system is congested and inoperable. This effect is caused principally by the fuel spray in the airstream which lowers the air temperature by evaporating. This malfunction is established by the presence of two conditions necessary for ice formation: the temperature of the air (mixture) being around its Dew point and the temperature of the

carburator components that the mixture comes in contact with, being around the freezing point of water (more correctly, water in presence of atomized fuel). When saturation of the water vapor in the induced atmospheric air is reached at the Dew point temperature, condensation will occur. That poses no problem unless the mixture comes in contact with components whose surface temperature is close to the freezing point. If that takes place condensed vapor from the mixture will freeze at these surfaces and remain attached to them while the engine operates. Gradually ice formation will extend to the point of congesting the carburator at the Venturi and Throttle areas. A steady pressure drop at the manifold is indicative of this condition and the engine 'doesn't pick up'. As more ice forms, less mixture is introduced at higher velocities at the Venturi tube maintaining the ice formation rate. To prevent this malfunction, since eventually the affected engine may cease operating, the induced air is preheated by an exhaust gas heat exchanger; this is controlled by the carburator heat controls.(Fig. 2.3a).

To simulate effectively this malfunction the relative humidity of the atmospheric air has to be computed from known values at different altitudes. This is done externally and supplied to the system. Referring to Fig. 2.2, the first quantities to be calculated are the temperature of the mixture at the Venturi tube and before the throttle valve. If these parameters near the freezing point of water then one of the two conditions for ice formation is met since gradually the body of the carburator will come close to that temperature provided there is no significant heat gain. Alternatively, if the ambient air

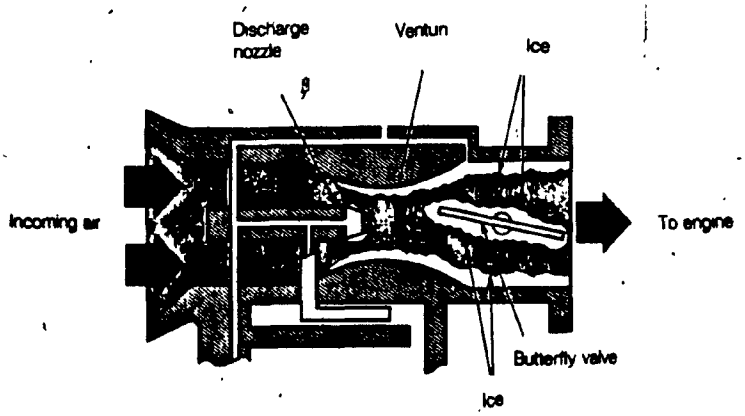
temperature is around or below freezing, if carburator heat is applied then the same condition will appear since ambient air flows around the carburator in most engines; other parameters such as heat conduction from the engine have to be included as well in this calculation. If the carburator chamber is then capable of freezing water, computation of the Dew point as mentioned is necessary; due to the high complexity of this malfunction, function generation is used to produce that parameter by interpolation from a psychrometric chart (Fig. 2.3b). The supplied variable is the relative humidity, ϕ, and the Venturi air temperature. Using a 'steam' table (see Appendix C) the saturation pressure, P_g, of the vapor can be found. The actual vapor pressure, P_v, will then be:

$$P_v = \phi P_g = \frac{\phi R T_g}{V_g} \qquad [2.15]$$

Applying the psychrometric chart and linear interpolation produces a usable approximation of the required Dew point. If it is found close to the temperature of the mixture then an icing flag is raised and a routine to simulate the effects of icing is invoked. That subprogramme consists mostly of a timed alteration of carburator parameters viz. reduction of the chamber diameter, throttle linkage error and short deregulations. Recovery can be effected depending on amount of congestion and applied carburator heat. Under certain conditions carburator heating may actually cause icing by melting ice crystals at low ambient temperatures. The precision of this

derivation is tolerable since it suffices for simulation purposes.

The equations derived up to this point define the air/fuel path through the carburator up to the throttle valve; this control is examined in the next section. Fuel consumption is covered in the fuel system model.



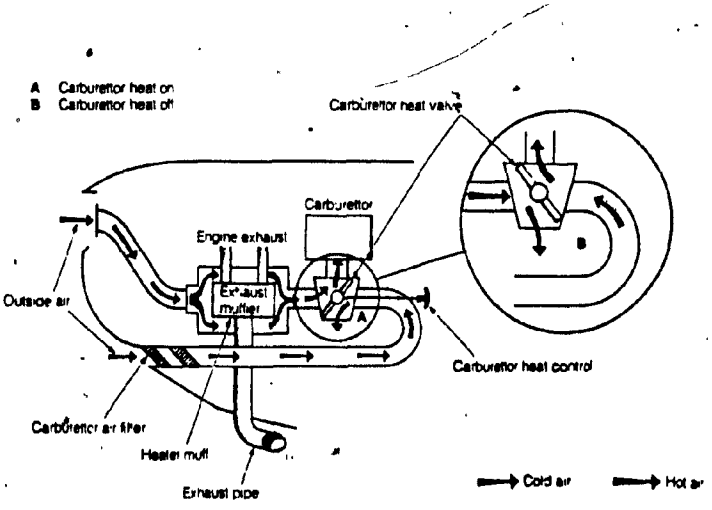


Fig. 2.3a Carburator icing and preheating.

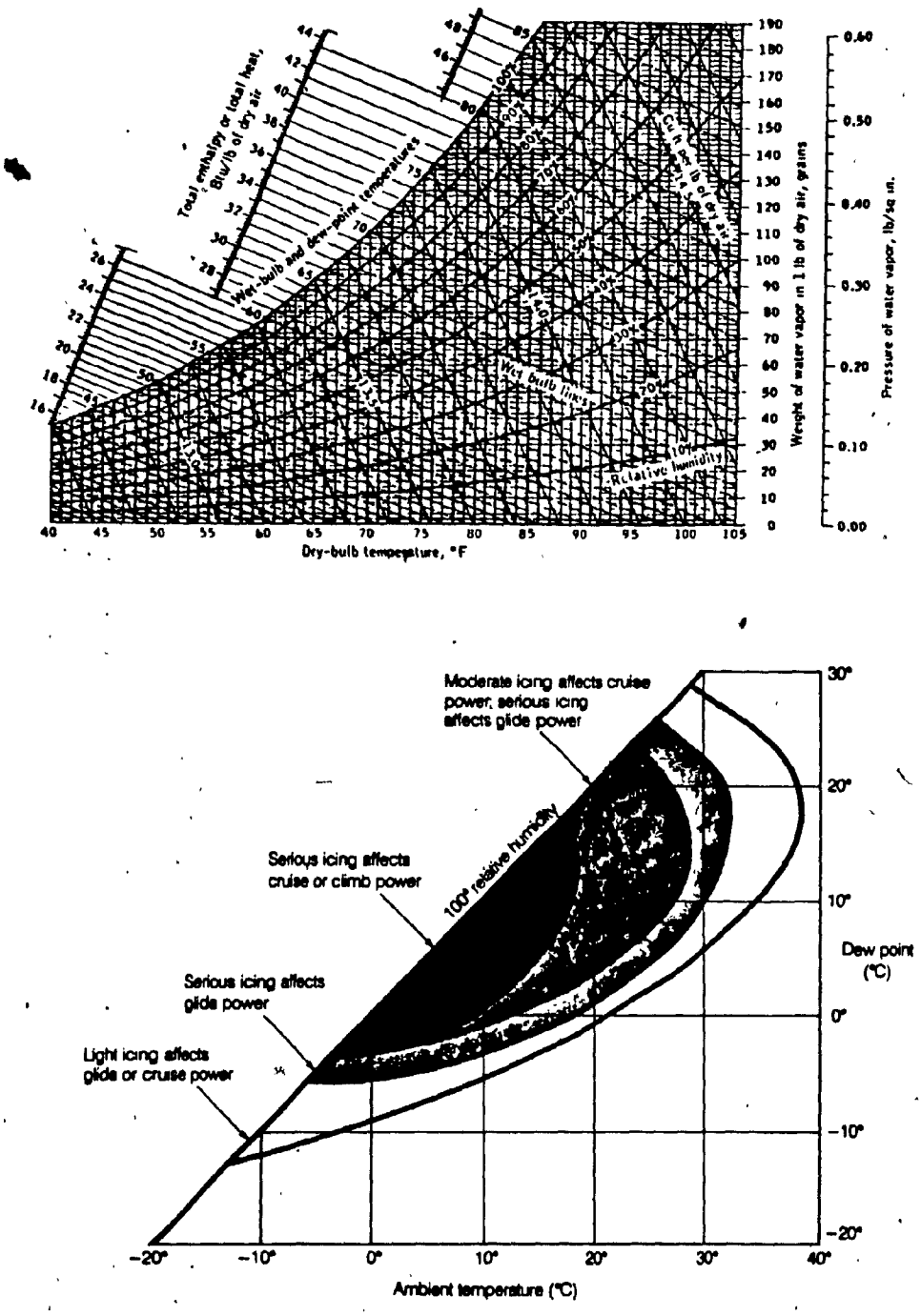


Fig. 2.3b The psychrometric and icing charts.

2.1.2 The engine manifold.

The operation of the manifold is straightforward, based on the following equations. For the engine's air/fuel intake volume flow rate the cylindrical capacity of the engine is used as follows:

$$\dot{V}_c = k_v \omega_e \qquad [2.18]$$

Suction is caused by the engine's angular velocity (first stroke of the Otto cycle). The air density in the manifold as affected by the Throttle control input can be expressed as:

$$f(\rho_m) = \frac{k}{k-1}\frac{P_a}{\rho_a} + \left(\frac{\rho_m}{\rho_a}\right)^2 \frac{V_m^2}{2} - \frac{P_a}{\rho_a}\frac{k}{k-1}\left(\frac{\rho_m}{\rho_a}\right)^{k-1} + \frac{V_m^2}{2}(k_t+1) \qquad [2.19]$$

where the Throttle parameter is assumed to vary as $0 < k_t < 1$. This quantity can be evaluated by the mathematical method used for the inlet air density:

$$\rho_m = \rho_{m_o} - \frac{f(\rho_{m_o})}{f'(\rho_{m_o})} \qquad [2.20]$$

Since the manifold pressure is displayed in the manifold pressure indicator, it may be expressed in terms of air density as follows:

$$P_m = P_a \left(\frac{\rho_m}{\rho_a}\right)^k \qquad [2.21]$$

Although the fuel and air mixture is varied in the carburator of the engine, for simulation purposes it is more efficient to introduce its effect in the cylinder model by function generation. This is included in the next part of the aeroengine model.

2.1.3 The cylinder/crankshaft assembly model.

This model describes the output of the engine in terms of the angular velocity of the crankshaft; first the Gross generated torque has to be defined:

$$Q_e = k_e \rho_m \eta_\omega \eta_x \qquad [2.24]$$

In this form the gross torque product of the engine is derived as affected by the efficiency factors of the engine's volumetric characteristics and the fuel/air ratio. Since we are interested in

the net engine product, that can be expressed as the Brake engine torque as per:

$$Q_b = Q_e - k_f \omega_e - k_o \qquad [2.25]$$

All the dry and viscous frictional losses are included in the above equation. Finally, the engine's angular velocity can be derived by integrating the angular acceleration of the engine as follows:

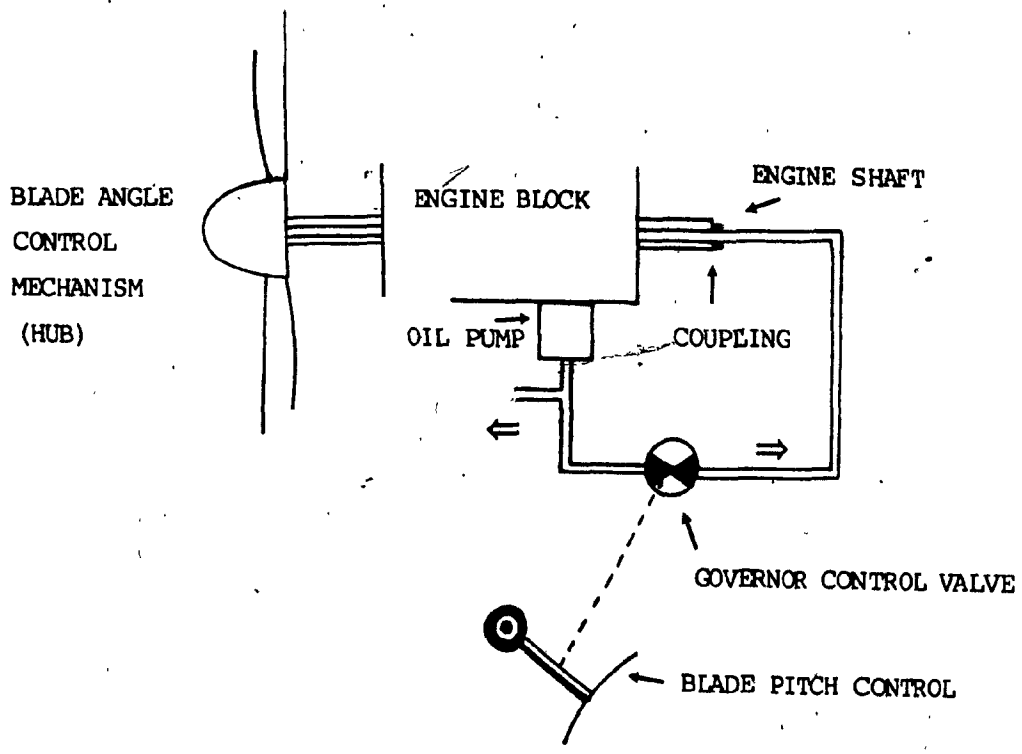
$$\omega_e = \frac{1}{J} \int \sum Q \, dt \qquad [2.26]$$

All the significant parameters that may affect angular velocity are included in this equation; the propeller demand torque is defined in the power dissipation model and the same applies to the starter torque and alternator demand torque quantities.

In summary, the above sets of equations describe adequately the operation of an Otto aeroengine for simulation purposes without introducing insignificant details; however, these descriptive equations are dependant on some definitions of the other system models since the powerplant is regarded as a single dynamic system.

2.2 THE PROPULSION MODEL.

As described in Chapter one, a light twin airplane is equipped with propellers that convert torque into thrust. In this case a two blade configuration applies with a 1:1 power transmission ratio. (Fig. 2.4). In general a propeller can be considered as a twisted wing; the blade cross sections are essentially of the same shape as those of a wing. The twisting is necessary to ensure that each blade section operates at a suitable angle of attack with respect to its rotational velocity. In action, the propeller participates in the forward motion of the airplane and rotates about its axis which usually coincides with the direction of flight. Of importance is the variable blade angle that allows for optimum propeller operation by adjusting the generated thrust to the characteristics of flight at any given condition, take off or cruise for example.



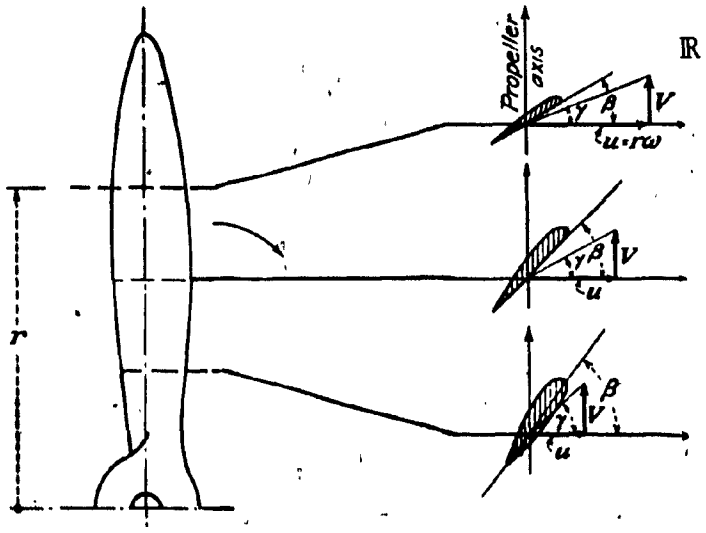


Fig. 2.4 Propeller and control.

2.2.1 The propeller model.

In order to derive the description of the propeller demand torque the following analysis has to be considered. For every propeller (Fig. 2.4) any blade section has a velocity component V in the direction of its axis - or flight, and a rotational velocity component u parallel to the cross sectional plane. Since the V component is constant for all cross sections and the u component proportional to the radius r then the propeller's angular velocity can be expressed as:

$$u = \omega r = 2\pi r n \qquad [2.30]$$

where n is the propeller's rotational velocity in rpm and ω its angular velocity. If we call γ the angle between the velocity vector u and the plane normal to the direction of flight then,

$$\gamma = \tan^{-1}\frac{V}{u} \qquad [2.31]$$

Increasing r will decrease the angle γ. In order to define the propeller advance ratio - or propeller slip, we denote as β the angle between the plane of rotation $\mathbb{R}$ and the propeller 'chord' - the blade section reference axis; then the angle of attack α of this cross sectional element is:

$$\alpha = \beta - \gamma \qquad [2.32]$$

The angle of incidence can then be defined as:

$$\alpha' = \beta' - \tan^{-1}\left(\frac{d}{2\pi r}\frac{V}{nd}\right) \qquad [2.33]$$

where β' is the angle between the null lift vector of the profile and the IR plane. If r is the radius of the propeller, then the propeller advance ratio is expressed as:

$$J = \frac{V}{nd} = \frac{2\pi V}{\omega d} \qquad [2.34]$$

Corresponding blade sections have the same angle of attack since they are geometrically similar. A useful quantity is the propeller thrust, and it can be expressed as:

$$T_p = \rho n d^4 C_t \qquad [2.35]$$

where C_t is the thrust coefficient and depends on Reynold's number at a given advance ratio. Finally the propeller demand torque can be defined as:

$$Q_p = \rho n^2 d^5 C_q \qquad [2.36]$$

where the torque coefficient, C_q, depends on the advance ratio, J, Reynold's number and propeller pitch, p. Another important quantity is the propeller power, P_p, which can be expressed in terms of torque and angular velocity and is equal to:

$$P_p = \omega_p Q_p = \rho n^3 d^5 C_p \qquad [2.37]$$

Values for the C_p parameter, called the coefficient of power as well as the thrust and torque coefficients are difficult to calculate and are usually supplied by the manufacturer of the propeller in the form of the so called 'propeller chart'. This chart is tabulated and used for function generation in this study (Fig. 2.5). Since the coefficients of torque were not available, the coefficients of power were used as follows:

$$Q_p = \frac{P_p}{\omega_p} \qquad [2.38]$$

Using these equations of propulsion a fairly detailed propeller model suitable for simulation can be derived. Since the aeroengine model requires an expression for the propeller in terms of torque, that will be;

$$Q_p = \rho n^2 d^5 C_q \qquad [2.45]$$

or in terms of power,

$$Q_p = \frac{\rho n^3 d^5 C_p}{\omega_p} \qquad [2.46]$$

In order to obtain the coefficient of power from the propeller chart, the pitch angle, β, must be calculated as well as the propeller's advance ratio, J. These quantities are expressed as:

$$\beta = \tan^{-1} \frac{4 p k_p}{3 \pi d} \qquad [2.47]$$

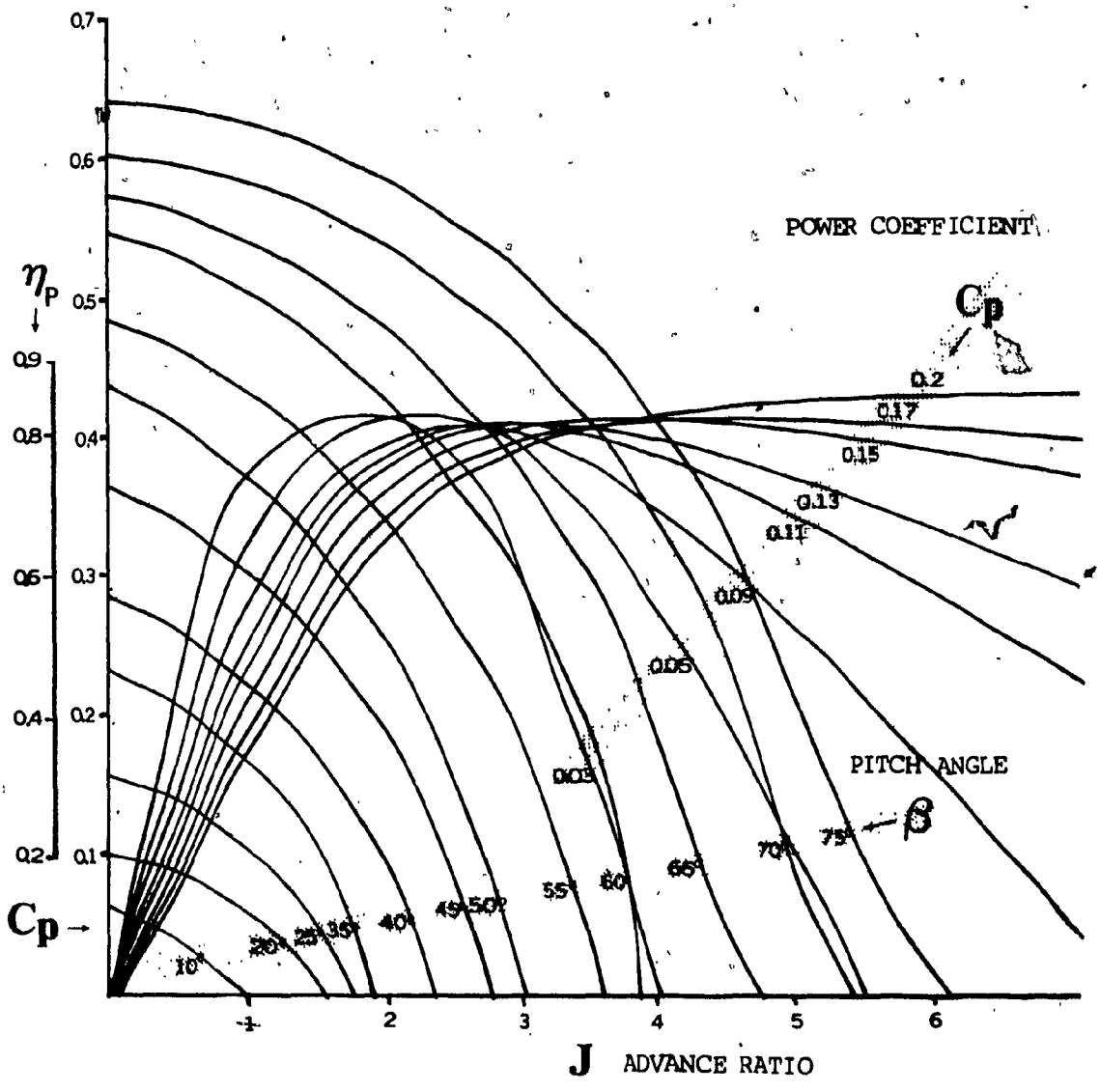


Fig. 2.5 The Propeller chart.

where d is the diameter of the propeller, and p is the pitch input from the control system of the powerplant. The advance ratio is then defined as:

$$J = \frac{2\pi V_f}{\omega d} \qquad [2.48]$$

the velocity, V_f, is the forward velocity of the aircraft and is supplied externally.

Propeller efficiency and thrust can also be calculated in a similar manner based on data from the propeller chart. The model as derived to this point incorporates the critical functions of the powerplant system. The next section describes the ancillary systems that influence the behaviour of the powerplant externally.

2.3 THE ANCILLARY AND CONTROL SYSTEM MODELS.

The associated systems of the powerplant are of importance in the simulation as well. Some play a critical role and others have only marginal effects. A selection therefore has to be made as to which systems are to be simulated so that a satisfactory ratio of fidelity to engineering cost is reached. The critical ancillary systems are thought to be the fuel and ignition systems, the electrical system, the lubrication (oil) system, the exhaust system in the form of the thermal model and the starter system as part of the ignition system. Control originates from the throttle quadrant and the panel and pedestal switches. The functions of these controls are represented in the models that use them as input devices. Displayable output is generated in a similar manner for the instruments and gauges of the panel that relate to powerplant operation. Considering fuel as the main 'ingredient' of flight, we may examine its path from the wing fuel tanks to the carburators at this point.

2.3.1 The fuel system model.

In light twin engined aircraft fuel is typically carried in tanks within the wings in an approximate total quantity of four hundred litres. Before it is supplied to the engine it can be switched on or off or, be crossfed from tanks to engines as required. This function is established by the fuel selector valve which is remotely controlled from a switch in the pedestal of the cabin. (Fig. 2.6).

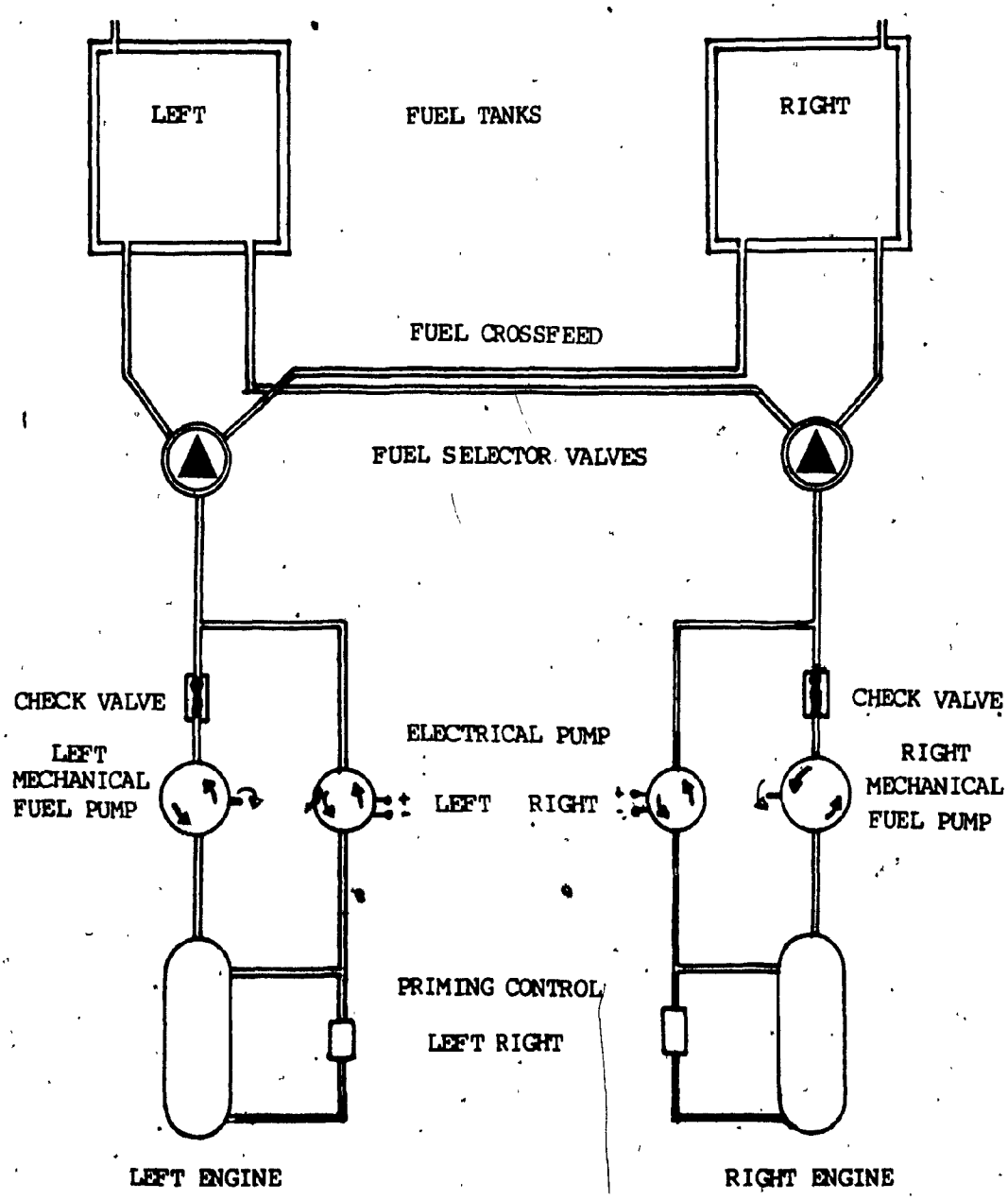


Fig. 2.6 Fuel flow diagram.

Fuel consumption is controlled by engine demand (or leakage) according to the Throttle quadrant settings and the ambient conditions. It is measured for various purposes. Fuel pressurisation is necessary so that adequate fuel quantities are always available at the carburator. The fuel is removed from the tanks by a main pump which is geared on the crankshaft of the engine and an auxiliary pump that is electrically driven from the power bus or the battery.

The rate at which fuel flows into the carburator can be expressed by the following equation where w_f is the specific fuel gravity; ΔP_a is the difference in ambient and Venturi tube pressures; d_j and c_f refer to the fuel sprayer jet coefficient of friction and diameter: (Ref. 2.6)

$$\Delta P_a = P_a - P_v$$

$$\dot{m}_f = c_f d_j^2 \sqrt{w_f \Delta P_a} \qquad [2.50]$$

knowing the fuel flow rate we can obtain an expression for fuel consumption by integration;

$$m_f = m_{f_o} - \int_o^t \dot{m}_f \, dt \qquad [2.51]$$

where m_f and m_{f_o} refer to the initial and remaining fuel

quantities in the tank. In order to calculate the specific fuel consumption of the engine, the delivered power has to be used as follows:

$$m_s = \frac{641.4}{e_c \eta_t P_e} \qquad [2.52]$$

Since the fuel is pressurised by two independant pumps and its pressure is displayed to the operator, we can express fuel pressure as the variable sum of the level of the two pressures at any given time, The mechanical fuel pump is of the 'diaphragm' type and its pressure varies with the engine's angular velocity; normal fuel pressure can be decreased by leakage or failure of the driving mechanism and can be increased by fuel line blockage or injection malfunction. As such it can be expressed as:

$$P_f = P_{f_o} k_f \omega_e \qquad [2.53]$$

In order to simulate the function of the fuel selector valve its operation can be expressed in logical terms of whether fuel is available through it and to which engine(s); in the CROSS position fuel can be routed to both engines as shown in the following table:

VALVE	POSITION	LEFT ENGINE	RIGHT ENGINE
LEFT	OFF	-	-
	ON	f/a	-
	CROSS	f a	f/a
RIGHT	OFF	-	-
	ON	-	f/a
	CROSS	f/a	f/a

[2.54]

As seen fuel is pumped into the carburator by two pumps; since the main pump adds a Torque demand to the engine its effect has to be calculated; this quantity is fairly constant under normal operation and is as follows:

$$Q_f = Q_p \; k_t \; \omega_e$$

[2.55]

A similar expression can be derived for the electrical pump in terms of pressure output and electrical load to the power bus.

2.3.2 The electrical system model.

Electrical power on this powerplant system is generated as seen in Chapter one by two alternators, one on each engine; these units produce a powerful AC current which is regulated and rectified. At this point the DC current enters the power bus and is distributed for consumption to the various electrical devices of the aircraft, the landing gear system, the flap mechanism, the avionics, the lighting etc. The power rating of the system is approximately of the order of 3 kW and the electrical load effects are noticeable primarily at low engine power or high propeller load. The following expressions represent the electrical system with a preset power load condition. The setting can also be varied depending on loading requirements. Fig. 2.7 shows in detail the electrical system of a light twin presented in Chapter one.

DC power is delivered to its respective power bus (left and right) with the possibility of tieing both busses together. Regulation is effected by means of varying the voltage of the excitation field of each alternator. This is managed by a solid state voltage regulator. In case of malfunction a protective mechanism shuts down the excitation circuitry; interestingly, an oscillatory condition may develop if demand is low and the angular velocity of the engine is high. The power capacity of each alternator can be expressed as a function of ω_e.

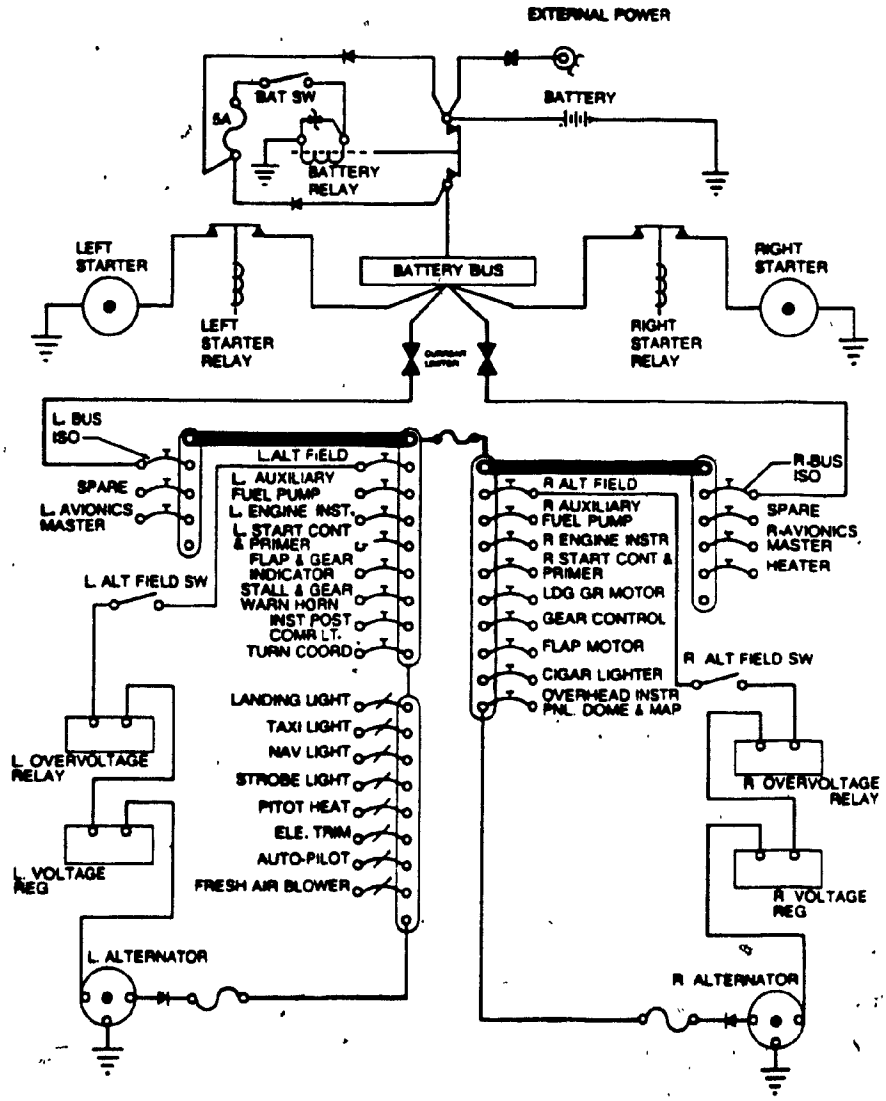


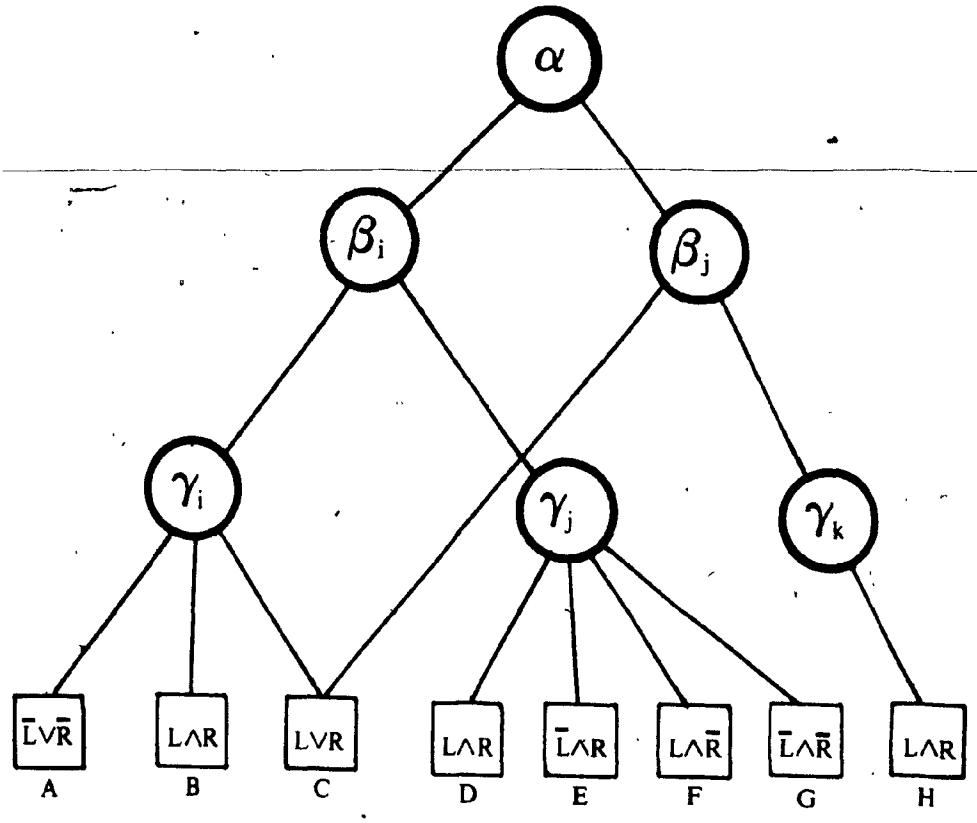
Fig. 2.7 Electrical power and control.

When the Bus Tie circuit is engaged, the power load is shared proportionatelly by both alternators. The battery of the system can be isolated or engaged in either power bus. If the power of the alternators exceeds or equals the demand, then the battery is charged otherwise it contributes in sharing the load. These functions are defined as:

$$P_a = \int (\eta_\omega, \omega_e)$$

$$\sum P_T = \sum P_{BI} + \sum P_{BA} \qquad [2.58]$$

It is possible to configure electrical power distribution in more than one way due to the possible switching. Out of sixteen possible permutations only eight are significant and these are represented by the decision tree of Fig. 2.8; at level one the battery switch is represented, at level two the bus tie circuit breaker and at level three the bus isolation circuit breakers (from the battery bus). The terminal nodes represent the selected configurations of the electrical system.



$$\Delta := \{\alpha\{\beta\{\gamma\{A,B,C\}\}, \gamma\{D,E,F,G\}, \beta\{\gamma\{H\}\}\}\}$$

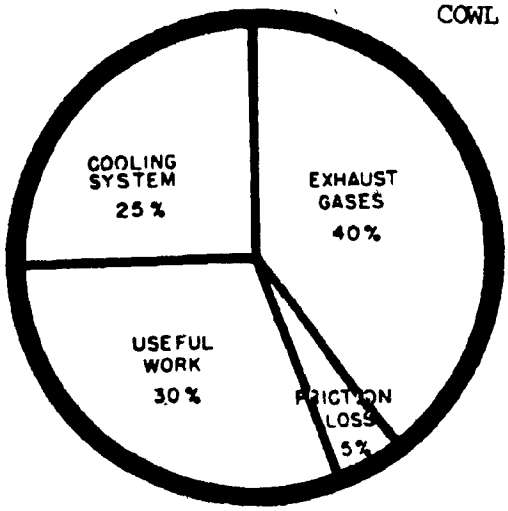
Fig. 2.8 The decision tree.

2.3.3 The thermal model.

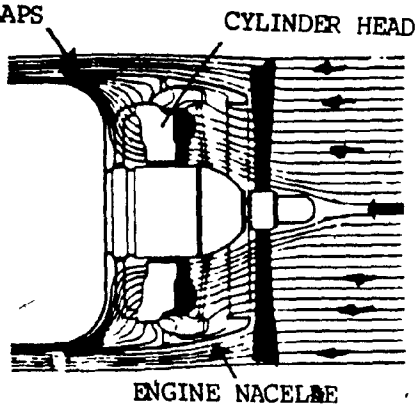
Heat being one of the byproducts of energy conversion in the engine, it plays a major role in the operation of the powerplant. Heat is generated mainly by the third stroke of the Otto cycle in the cylinders as defined by the thermal efficiency of the machine, the power delivered and other factors such as the enthalpy of the fuel used etc. Heat is also generated by viscous and dry friction on other components of the engine. Of interest are the heat present in the cylinder heads, the heat dissipated in the air and the heat present in the exhaust gases. Displayable quantities to the operator are the cylinder head temperature and the exhaust gas temperature. Heat dissipation is regulated by the so called cowl flaps that control ambient air flow around the cylinders of the engine. (Fig. 2.9). A small proportion of the generated heat is routed back into the engine system via the carburator heating mechanism and is used to prevent formation of ice within the manifold and the throttle valve as seen in section 2.1.

The amount of heat generated when the mixture is ignited can be expressed according to the first law of thermodynamics as: (Ref. 2.3)

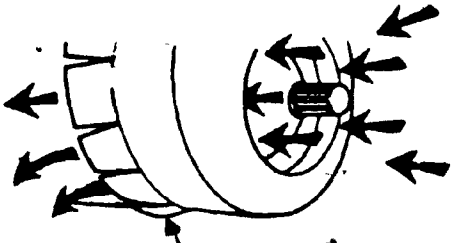
$$q_c \doteq k_{f/a} q_k k_h \omega_e \qquad [2.60]$$



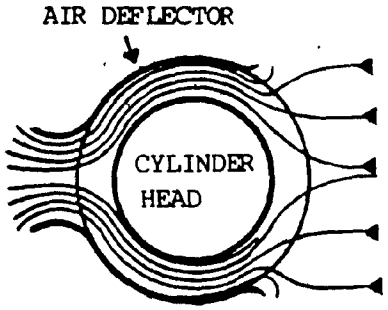
GENERATED HEAT DISTRIBUTION

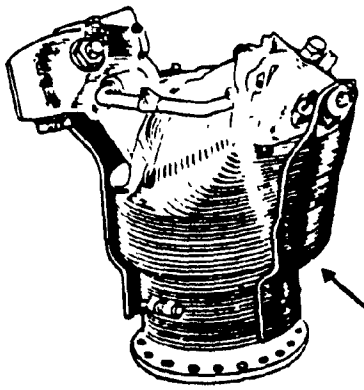


AIRFLOW AROUND ENGINE

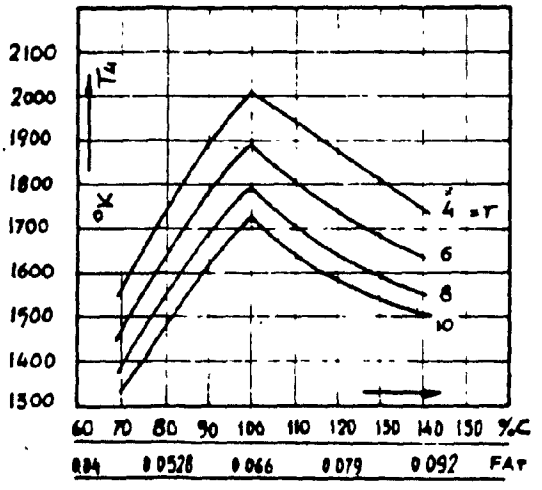


COWL FLAPS (Adjustable)





TYPICAL CYLINDER HEAD



F/A RATIO EFFECT ON EGT

(T : temperature of combustion)

Fig. 2.9 Heat flow and control.

where is the heat generated by the complete combustion of 1 kg of fuel - AVGAS 100/130 type in this case. This amount of heat is produced in each of the cylinders every two revolutions in two combustion cycles per revolution. Due to the short length of time that the exhaust gases remain in a cylinder, only a small amount of heat is conducted onto the cylinder heads, roughly estimated at two per cent; however, this heat quantity accumulates very fast and varies primarily with angular velocity, the fuel/air ratio and cooling air flow and temperature. The temperatures of the cylinder heads are to be prevented from reaching levels over 250 °C therefore it must be monitored by the operator. This quantity is displayed with the other engine parameters and is expressed as follows:

$$T_h = \int \sum \dot{q}_h dt = k_h \int \left[\dot{q}_c - k_c T_h - T_a \right] dt \qquad [2.62]$$

where the heat flow rates $\dot{q}$, in and out of the component express the quantities of heat gained and lost. Cooling is controlled by the nacelle cowl flaps of the engine. These allow or restrict airflow around the cylinder heads as set by the cowl flap control lever. The cooling rate varies with ambient temperature and pressure, the aircraft's velocity and the deflecting baffle pressure drop around the cylinders. Function generation is used for this part of the simulation. The engine's cooling requirements are shown in Appendix C.

Some heat from the combustion chamber flows through the pistons, the rods and the crankshaft case into the oil used for lubrication. This effect is presented in the oil system model. The temperature of the exhaust gases is used for fuel control purposes and is displayed to the operator by the EGT indicator. This temperature is that of the exhaust gases on departure from the cylinder. It can be calculated as:

$$\dot{q}_X = c_v \left[T_o r^{k-1} + \frac{f/a\ \dot{m}_f\ q_c}{c_v} \right) r^{1-k} - T_o \right] - \dot{q}_c$$

[2.65]

where r is the compression ratio of the engine; the adiabatic exponent k is the ratio of the specific heat of the exhaust gases under constant pressure and volume.

The thermal efficiency of the powerplant can be calculated as follows:

$$\eta_\theta = \frac{\omega_\theta}{q_t}$$

[2.66]

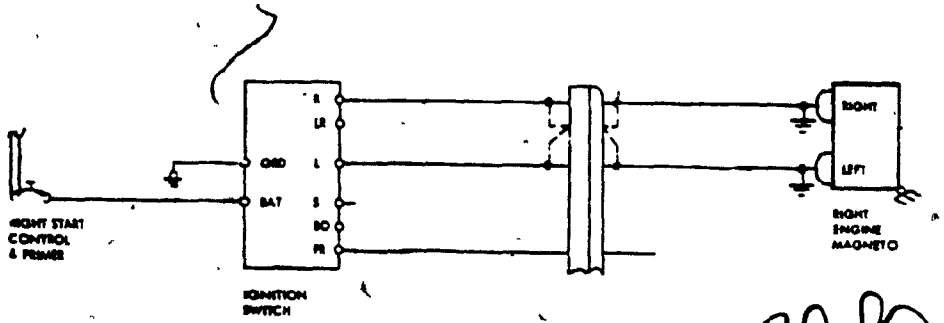
where thermal efficiency depends on the end temperatures of the compression stroke.

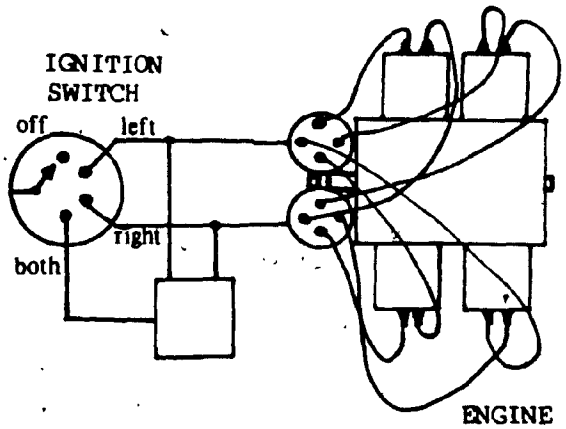
2.3.4 The ignition system model.

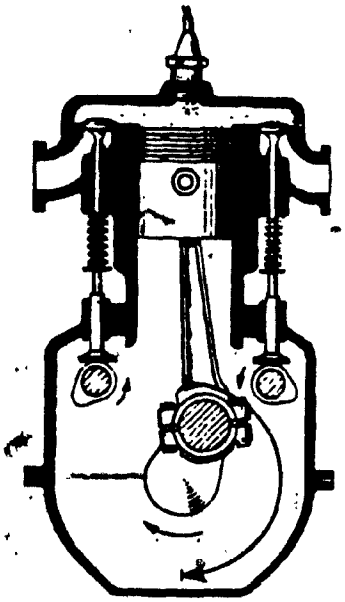
Ignition of the induced mixture is provided by pairs of igniters (sparkplugs) mounted on each cylinder head (combustion chamber) of the engines. The high voltage needed for the arc to ignite the fuel mixture is generated by a small generator driven by the engine (magneto); the high voltage is then distributed to the spark plugs through the distributor system where it is timed so that ignition occurs at the end of the compession cycle in each cylinder. The ignition system is duplicated in each engine for various reasons; selection is made through the start/ignition switch for each engine on the panel. This system is mostly of logical nature as it affects the operation from the simulation viewpoint. It is presented in Fig. 2.10 and it is modelled in Boolean terms.

$$I \equiv \{ x_i \wedge \bar{c}_i \} \vee \{ x_j \wedge c_j \} \wedge F_a \qquad [2.67]$$

When both ignition systems are engaged combustion is more uniform and at times faster. That can take effect when the fuel mixture is very lean and may offset knocking under certain conditions.







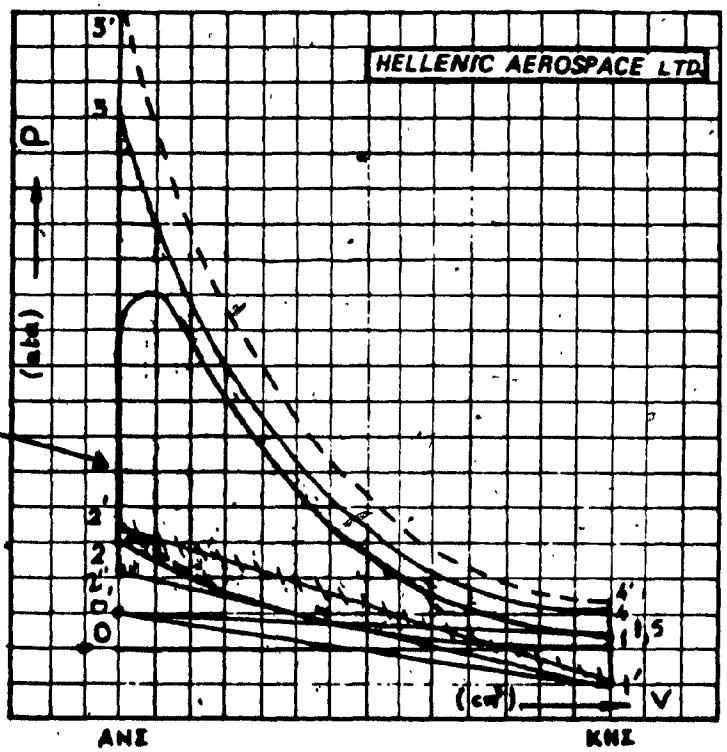


Fig. 2.10 Ignition and control.

2.3.5 The oil system model.

Oil is used in aeroengines as a lubricant. As such it may affect the output of the engine in terms of frictional losses due to its viscosity at varying levels of temperature. Oil in aeroengines is usually injected onto the crankshaft - piston rod joints and the crankcase as well as the joints of the pistons and their rods. It is collected in a sump, filtered, cooled if necessary, and recirculated to the injectors via a pressurising pump or in some cases two pumps (scavenge/return). The oil pressure pump supplies pressure to the propeller governor which controls the blade angle of the propeller at the operator's discretion. The pressurised oil reaches the propeller hub mechanism through the crankshaft. Fig. 2.11 shows the simulated oil system.

Quantities useful for the purpose of simulation are the temperature and the pressure of the lubricant and the viscous friction factor at different temperatures; since the latter varies with time it is simpler to model it by function generation based on given engine data, without affecting the fidelity of the simulation. Oil pressure varies with angular velocity, pressure demand and circulation rate. It can be expressed as:

$$P_o = P_d\ (k_{\omega_e} - k_f) \qquad [2.70]$$

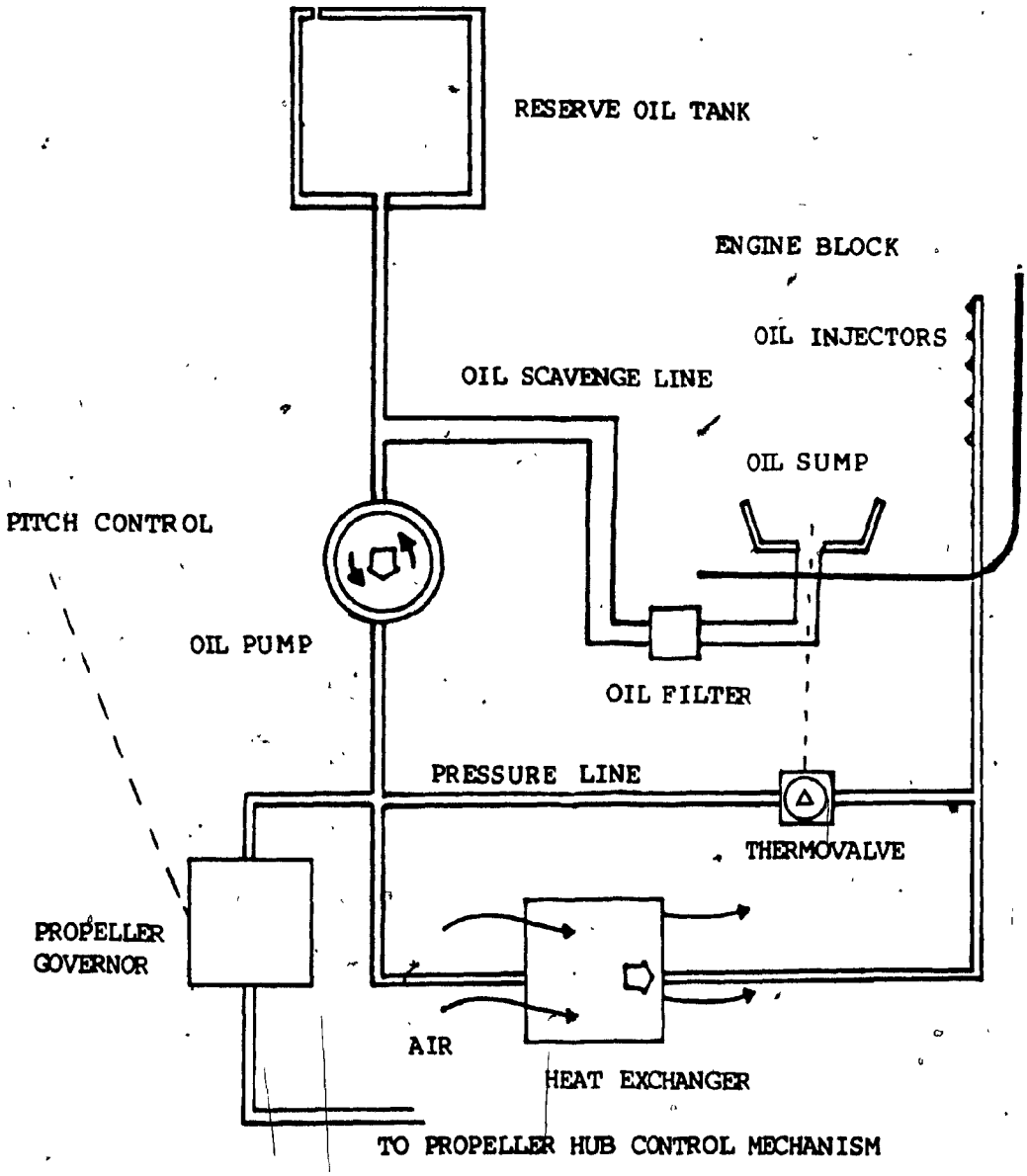


Fig. 2.11 Oil flow and control.

Oil temperature can be expressed as the sum of the amount of heat gained and lost by the oil mass. Gains are due to conduction with hotter components and friction. Losses are due to cooling mostly through the oil heat exchanger unit.

$$T_o = \int \sum \dot{q}_o \, dt \qquad [2.7\text{T}]$$

2.3.6 The control system model.

Commands to the powerplant system originate from the control panel and pedestal of the aircraft as described in Chapter one; these controls and their functions are simulated in the models that receive input from them. Some of them are of discrete nature such as all the switches and circuit breakers of the system. The balance is of analogue nature and are treated as such. The following list shows the distribution and nature of engine controls. (Fig. 2.12).

Analogue control:	**Function:**
RPM levers	Engine power
Pitch levers	Propeller thrust
Mixture levers	Fuel Air ratio
Carburator Heat levers	Intake air warming
Cowl Flap levers	Air Cooling

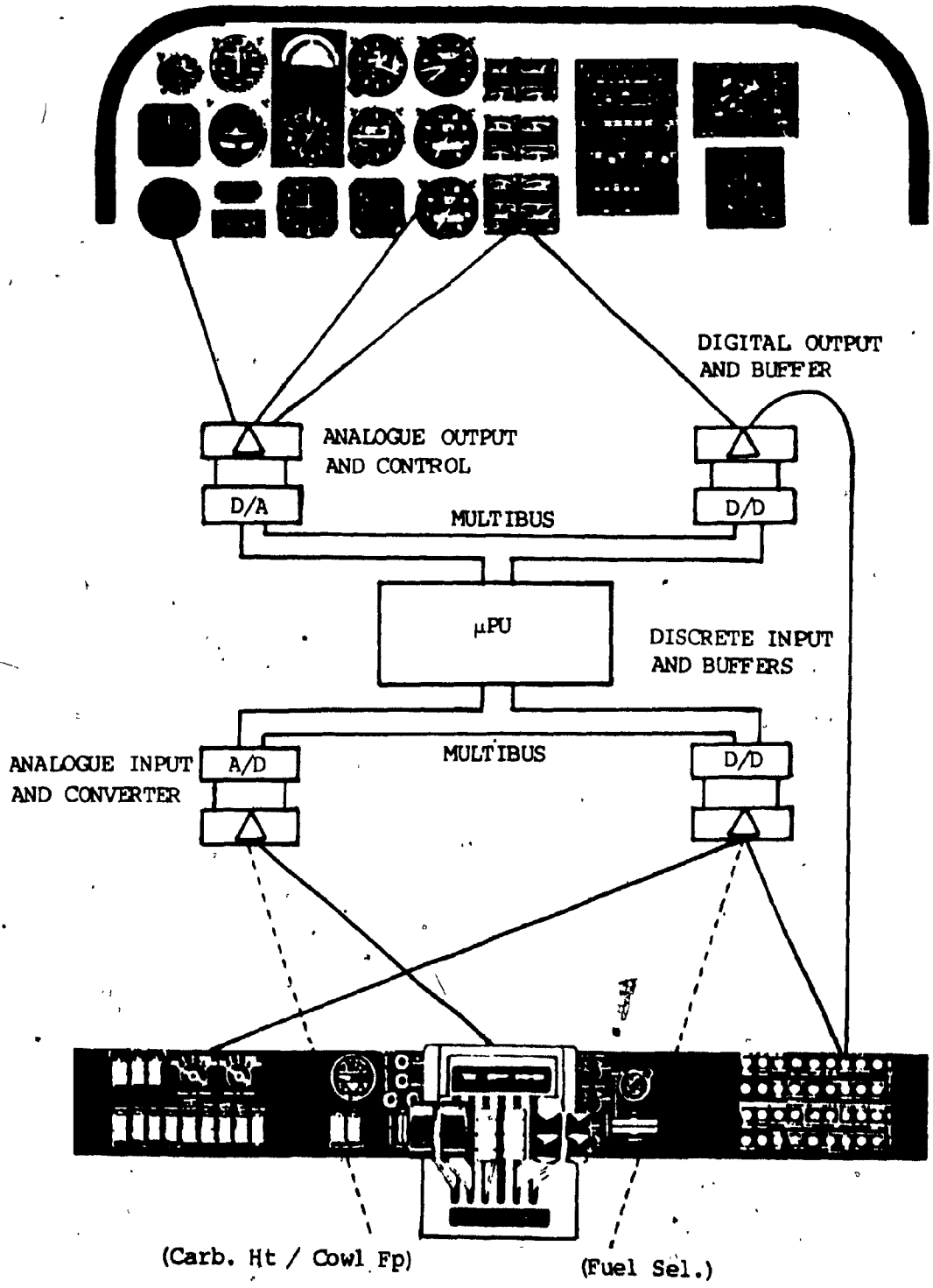


Fig. 2.12 The powerplant control system.

Discrete Control:	**Function:**
Ignition switches	Ignition, Priming, Start
Aux. Fuel Pump sw.	Electrical fuel pumps
Alternator Field sw.	Alternator Excitation
Battery switch	Battery control
Bus Tie C.B.	Power Busses Tied together
Bus Isolation C.B.	Battery isolation from bus
Alt. Field C.B.	Field isolation and protection
Aux. Pump C.B.	Aux. Pump protection

Engine status is displayed on various indicators on the instrument panel. These are the Engine RPM indicators and the Engine manifold pressure indicators (servodriven instruments); the fuel quantity, fuel pressure, oil temperature, oil pressure, cylinder head temperature, electrical power load and the exhaust gas temperature indicators (galvanometres). Simulated operation of all the controls and indicators is described in Chapter five. Command and feedback input from these devices is transferred to the computer circuitry via the appropriate transducers, viz. potentiometres and switches.

All the system models derived above compose the complete engine system model - either the left component or the right. They are used in the next Chapter to derive the Simulation Software modules.

chapter 3

SIMULATION SOFTWARE

This part of the study presents the Software module design for the various powerplant systems based on the derivations of Chapter two. They are presented in block notation so that their structure is illustrated with the fundamental equations that model each powerplant system. A general view of the model software is given first exposing the system interrelations; the software structures of this Chapter include only the powerplant systems that are simulated. The Real Time executive is presented in the next Chapter; specific programming information for all the software models can be found in Appendix A in procedural notation.

Integration is performed by the trapezoidal rule which is a variation of Simpson's rule (Ref. 2.4). This is done by computing the sucessive ordinates of the expression that is to be evaluated, multiply by Δx to find the 'areas' of the trapezoids and add them by simple arithmetic. Since the error is a rather small fraction of the total, this method gives a good approximation and is quick to calculate by a digital machine. The trapezoidal rule is expressed by the formula:

$$\int_a^b y\,dx = \frac{\Delta x}{2}(y_0 + 2y_1 + 2y_2 + \cdots + 2y_{n-1} + y_n)$$

In order to approximate the true system operation, the software system modules are executed sequentially and continuously at a selected rate. Continuous solutions of the model equations can only be obtained from an analogue computer which is a continuous computing device (see Appendix C); however, since the simulated system is a rather slow first order system, a bandwidth of 4 kHz such as that of an analogue computer is far above the estimated Nyquist criterion cutoff of 100 Hz. In practical terms a 20 to 30 Hz iteration rate reached a satisfactory level of fidelity and allow for some background activities.

The Δx (dt) variable of the trapezoidal rule expression is assigned values equal to the time frame being used. For a 20 Hz iteration rate $\Delta x = 0.05$ seconds. The criteria for the selection of the iteration rate are functions of what computing capacity is available. Fidelity is established by degree of conformity to performance data of the simulated system and sensory comparison. Figure 3.1a shows the simulation block diagram.

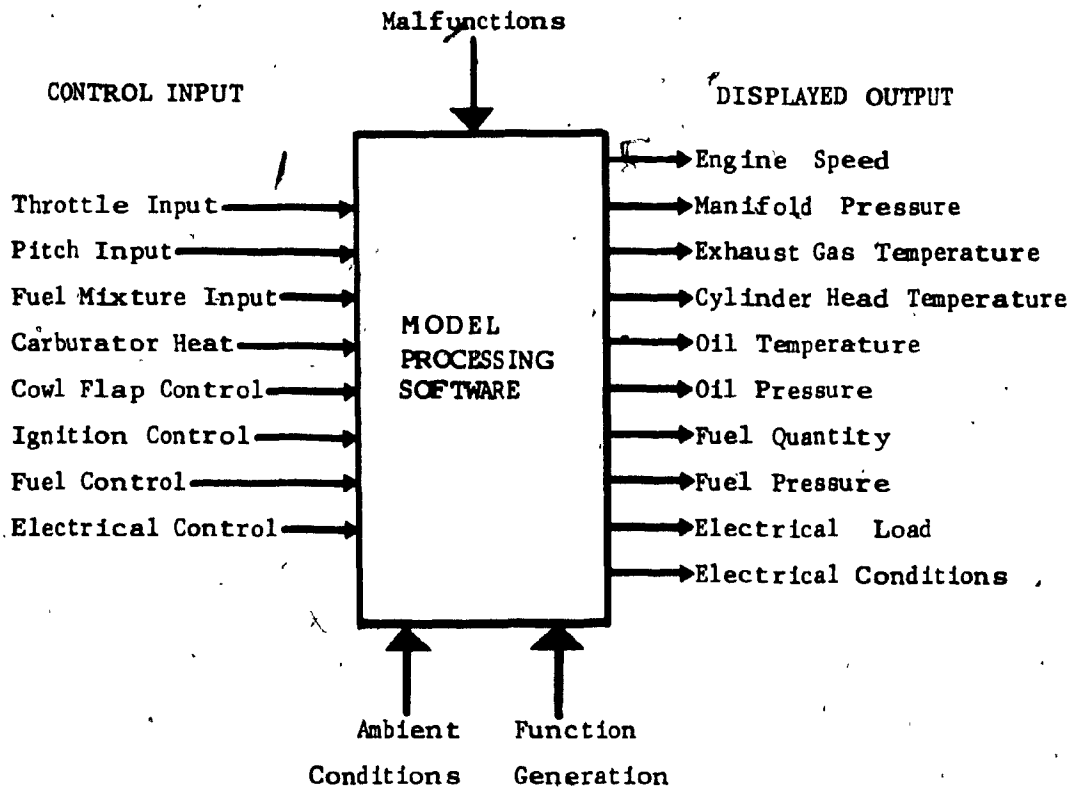


Fig. 3.1a The general simulation model.

3.1 THE SYSTEM MODEL SOFTWARE

This section includes the software modules that represent the power generation, dissipation and control systems of the powerplant. These are: the Aeroengine module, the Electrical system module, the Fuel system module, the Ignition system module, the Thermal module, the Oil system module and the Propulsion system module. Most of these models are closely related and interdependant. Since instability, looping and sequential problems can result in such an arrangement, very careful sequencing and crosschecking of the model equations had to be established so that stability and fidelity would be maintained under most possible configurations. Fig. 3.1b shows the interaction among the system modules in exchanging variable data. The level of interrelating is highest with the Aeroengine module since its main output, the angular velocity, depends on almost all parameters of the related systems at varying degrees of sensitivity and vice versa; data traffic is also high in external I/O transfers, that is information to and from the panel controls and indicators via the data transfer and conversion systems. These software modules are executed at differing priorities by the Real Time executive discussed in the next Chapter. The input/output and function generation modules belong to the system software and are presented in the next Chapter as well. Tables of system and other variables are listed with the models for easy reference. Integration can be performed at dt values varying with the iteration rate of the model software.

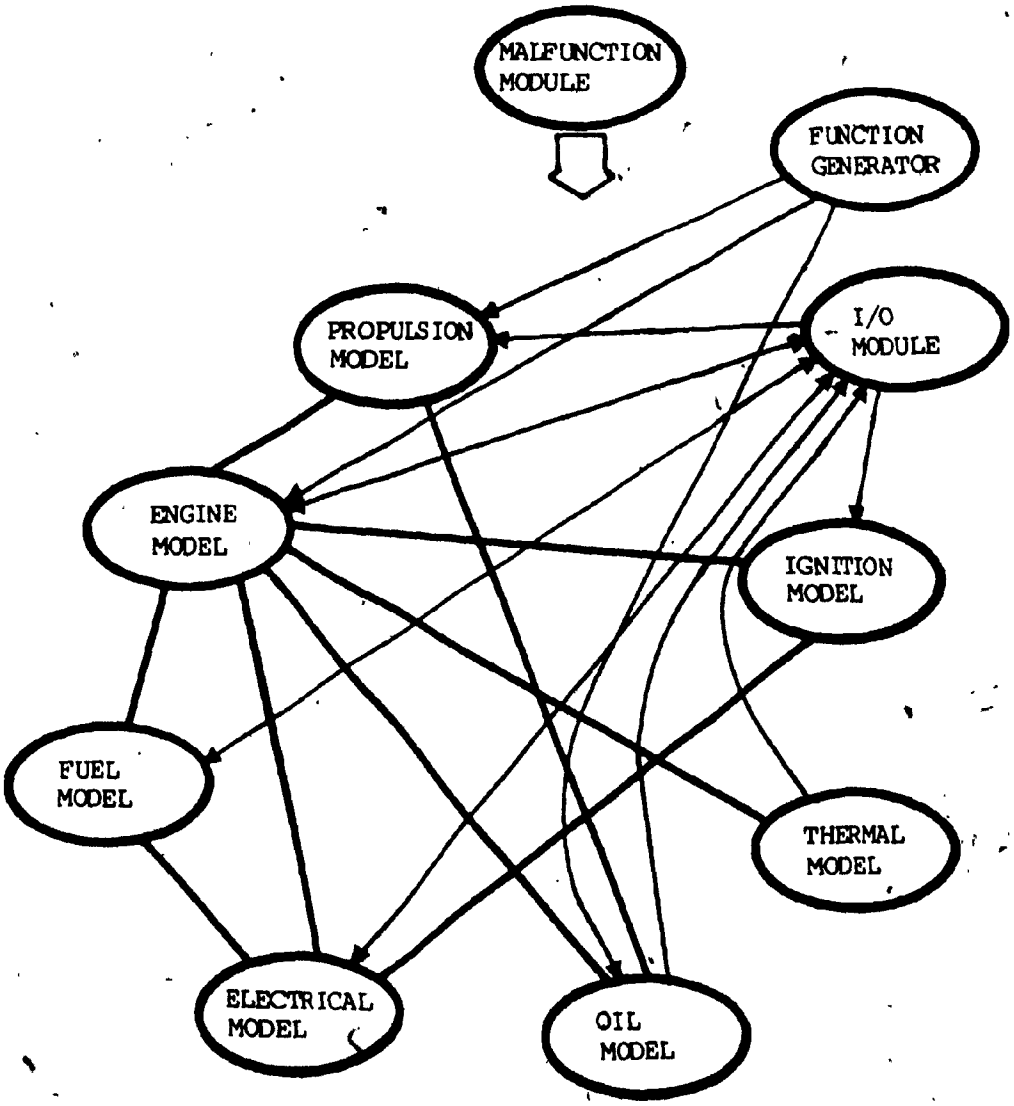


Fig. 3.1b System model interaction.

3.1.1 The Fuel System Module.

Both fuel systems of the Powerplant are related via the fuel control valves. Fuel sloshing in the tanks is not part of this study since it is usually categorised as a flight model component. The two fuel systems are referenced as Left (L) and Right (R); Mechanical and electrical variables are computed in the aeroengine and electrical Modules; Fig. 3.2 outlines the structure of the fuel system module. The critical computation is that of the Fuel Available flags; these flags signal the presence of fuel in the carburators.

Fuel pressure is seen to vary with angular velocity and/or power bus voltage for the electric pump. Fuel flow is computed from a single expression [2.50] and the tank fuel quantities produced by integration of the fuel flow [2.51]. The specific fuel consumption calculation of each engine and the system are based on [2.52]; Malfunctions affect the various parameters directly.

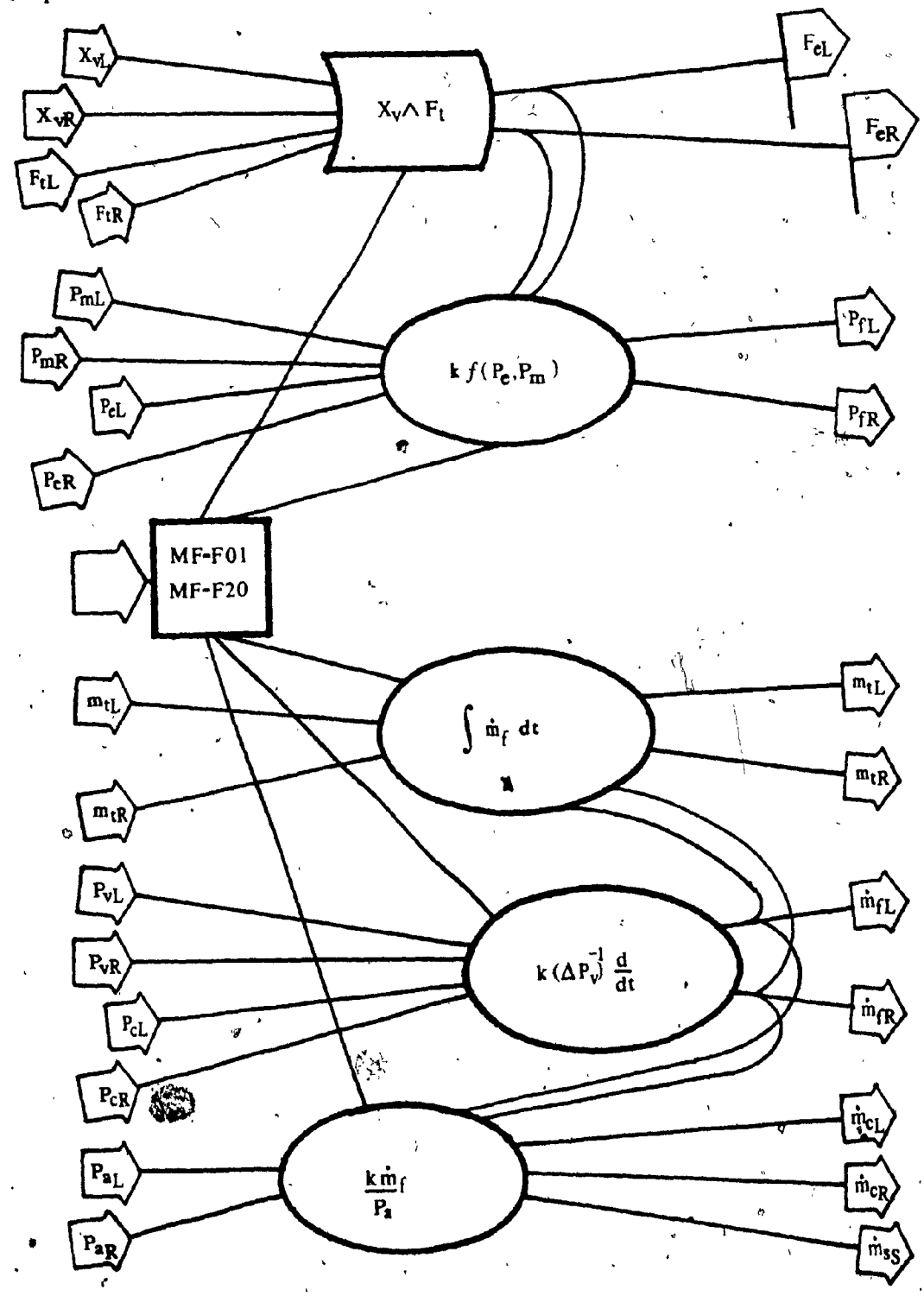


Fig. 3.2 Fuel system Module outline.

FUEL SYSTEM

System Variables.

Variable	Description	Units
F_{aL}	Fuel available in Left tank	flag
F_{aR}	Fuel available in Right tank	flag
F_{eL}	Fuel available to Left engine	flag
F_{eR}	Fuel available to Right engine	flag
X_{vL}	Fuel selector valve Left system	boolean
X_{vR}	Fuel selector valve Right system	boolean
$\dot{m}_{pL}$	Fuel flow at priming, Left eng.	kg/sec
$\dot{m}_{pR}$	Fuel flow at priming, Right eng.	kg/sec
P_{eL}	Fuel pressure, Aux. pump Left	kg/m
P_{eR}	Fuel pressure, Aux. pump Right	kg/m
P_{mL}	Fuel pressure, Mech. pump Left	kg/m
P_{mR}	Fuel pressure, Mech. pump Right	kg/m
P_{fL}	Fuel pressure true Left	kg/m
P_{fR}	Fuel pressure true Right	kg/m
P_{fiL}	Fuel pressure indicated Left	kg/m
P_{fiR}	Fuel pressure indicated Right	kg/m
$\dot{m}_{fL}$	Fuel flow to Left engine	kg/sec
$\dot{m}_{fR}$	Fuel flow to Right engine	kg/sec
$\dot{m}_{cL}$	Fuel consumption Left engine	kg/sec
$\dot{m}_{cR}$	Fuel consumption Right engine	kg/sec
$\dot{m}_{sL}$	Specific fuel consumption Left	kg/kNsec
$\dot{m}_{sR}$	Specific fuel consumption Right	kg/kNsec
$\dot{m}_{sS}$	System average spec. fuel cons.	kg/kNsec

FUEL SYSTEM

Malfunction Variables.

Variable	Description	Units
MFF01	Fuel flow cutoff Left system	flag
MFF02	Fuel flow cutoff Right system	flag
MFF03	Fuel leak in Left system	flag
MFF04	Fuel leak in Right system	flag
MFF05	Fuel leakage rate Left system	kg/sec
MFF06	Fuel leakage rate Right system	kg/sec
MFF07	Auxiliary pump out of order Left	flag
MFF08	Auxiliary pump out of order Right	flag
MFF09	Quantity sensor defective Left	flag
MFF10	Quantity sensor defective Right	flag
MFF11	Selector valve inoperative Left	flag
MFF12	Selector valve inoperative Right	flag
MFF13	Check valve internal leak Left	flag
MFF14	Check valve internal leak Right	flag
MFF15	Quantity Indicator defective Left	flag
MFF16	Quantity Indicator defective Right	flag
MFF17	Pressure Indicator defective Left	flag
MFF18	Pressure Indicator defective Right	flag
MFF19	False Pressure Indication Left	kg/m
MFF20	False Pressure Indication Right	kg/m

3.1.2 The Electrical System Module.

The electrical system is simulated in this module according to the model derived in the previous chapter. The key configuration of the system is the Bus Isolation circuit breakers from the battery bus and the Bus Tie circuit breaker which connects in parallel both power busses. In this mode with the parallel option of the voltage regulators both alternators share the load equally. Of interest are also the battery current limiters to each bus; these allow approximately 40 Amperes maximum current to pass to both busses; depending on the type used, that allows the same amount of current to cross from one bus to the other when both Isolation CBs are closed. The valid permutations are represented by a decision tree of degree four, reduced from sixteen possible combinations to only eight significant ones at depth four as seen in Chapter two.

The module computes values for three system flags that signal power availability from the alternators or the battery; two other flags are raised for irregular voltage conditions and are mutually exclusive for each engine system. (Fig. 3.3). Power computations are made by [2.70], and [2.72]; of importance is the implementation of the configuration decision tree as a case statement (Appendix A). Circuit breakers are taken care by another part of the module.

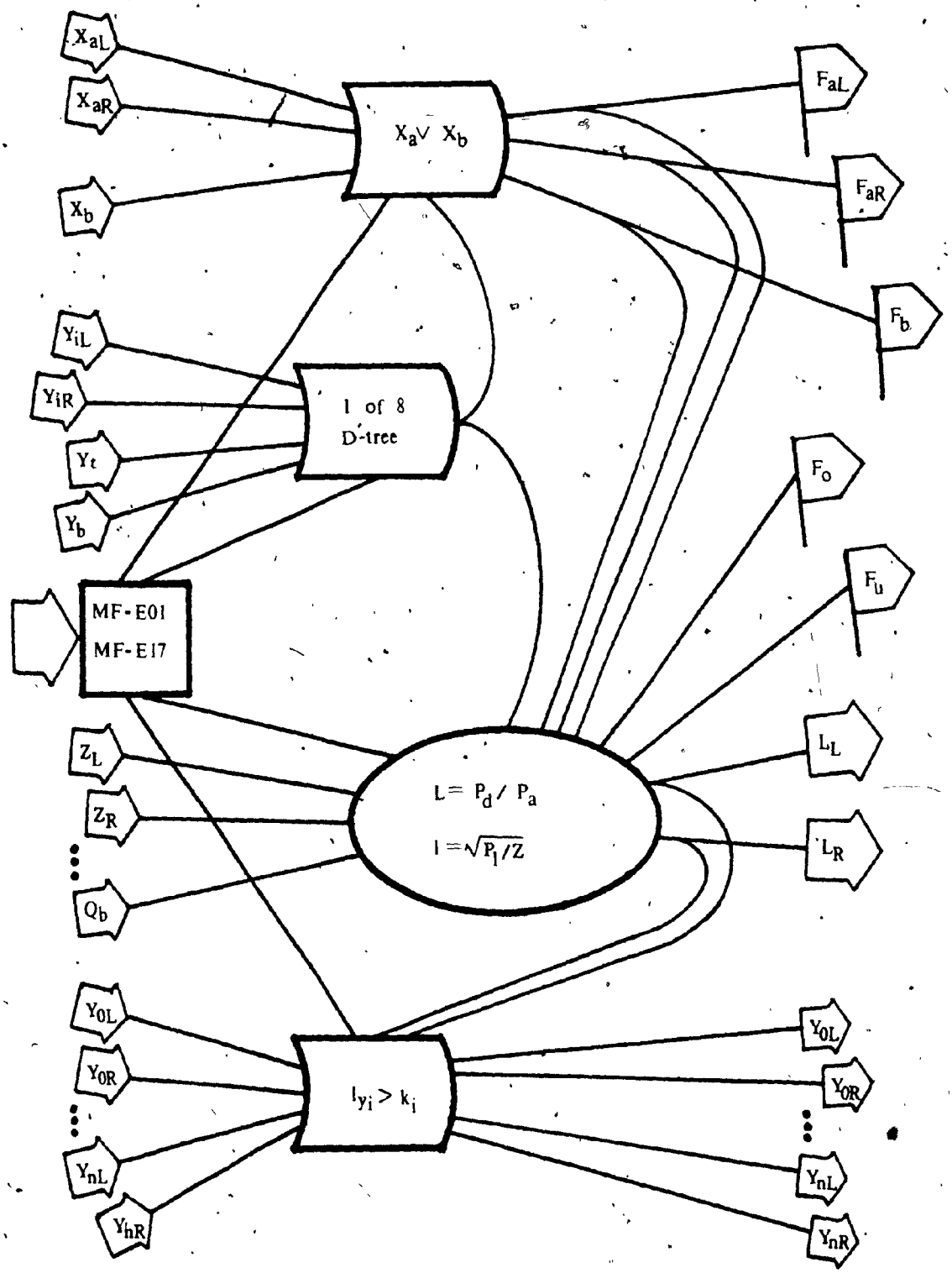


Fig. 3.2 Electrical system Module outline.

ELECTRICAL SYSTEM

System Variables.

Variable	Description	Units
X_{fL}	Switch Auxiliary fuel pump Left	flag
X_{fR}	Switch Auxiliary fuel pump Right	flag
X_{sL}	Switch Starter Left	flag
X_{sR}	Switch Starter Right	flag
X_{ilL}	Switch Ignition Left subsys. Left	flag
X_{irL}	Switch Ignition Right subsys. Left	flag
X_{ilR}	Switch Ignition Left subsys. Right	flag
X_{irR}	Switch Ignition Right subsys. Right	flag
X_b	Switch Battery relay	flag
X_{aL}	Switch Alternator Field Left	flag
X_{aR}	Switch Alternator Field Right	flag
Y_{aL}	C. B. Alternator Field Left	flag
Y_{aR}	C. B. Alternator Field Right	flag
Y_{sL}	C. B. Start, Prime Control Left	flag
Y_{sR}	C. B. Start, Prime Control Right	flag
Y_{iL}	C. B. Battery Isolation Left Bus	flag
Y_{iR}	C. B. Battery Isolation Right Bus	flag
Y_t	C. B. Bus Tie together	flag
Y_{pL}	C. B. Auxiliary Pump Left	flag
Y_{pR}	C. B. Auxiliary Pump Right	flag
I_{sL}	Starter Current Intensity Left	Amp
I_{sR}	Starter Current Intensity Right	Amp
F_b	Battery power available	flag
F_{aL}	Alternator power available Left	flag
F_{aR}	Alternator power available Right	flag
$F_{\omega L}$	Minimum Ang. Vel. available Left	flag
$F_{\omega R}$	Minimum Ang. Vel. available Right	flag
P_{aL}	Power Available (Alt.) Left	Watt
P_{aR}	Power Available (Alt.) Right	Watt
C_b	Battery charge	Amp/hr
P_{dL}	Power Demand Bus Left	Watt
P_{dR}	Power Demand Bus Right	Watt
P_d	Power Demand System	Watt
Z_L	Load Impedance Bus left	Ohm
Z_R	Load Impedance Bus Right	Ohm
Z_S	Load Impedance System	Ohm
E_L	Bus Voltage (rms) Left	Volt
E_R	Bus Voltage (rms) Right	Volt
E_S	Bus Voltage System	Volt
I_L	Bus Current Intensity Left	Amp
I_R	Bus Current Intensity Right	Amp
I_S	Bus Current Intensity System	Amp
E_{bL}	Battery Bus Voltage	Volt
I_b	Battery Bus Current Intensity	Amp
L_L	Loadmeter indication Left	(%)
L_R	Loadmeter indication Right	(%)

ELECTRICAL SYSTEM. Malfunction Variables.		
Variable	Description	Units
MFE01	Battery out of order	flag
MFE02	Battery connection intermittent	flag
MFE03	Battery charge	(%)
MFE04	Alternator field or switch out	flag
MFE05	Alternator field or switch out	flag
MFE06	Alternator or regulator or overvoltage control or fuse open. Left	flag
MFE07	Alternator or regulator or overvoltage control or fuse open. Right	flag
MFE08	Shorted voltage regulator Left	flag
MFE09	Shorted voltage regulator Right	flag
MFE10	Overvoltage relay cutoff Left	flag
MFE11	Overvoltage relay cutoff Right	flag
MFE12	Loadmeter out of order Left	flag
MFE13	Loadmeter out of order Right	flag
MFE14	Aux. fuel pump defective Left	flag
MFE15	Aux. fuel pump defective Right	flag
MFE16	Starter defective Left	flag
MFE17	Starter defective Right	flag

3.1.3 The Thermal Module.

The displayable thermal quantities are computed in this module. These are the cylinder head temperature and the exhaust gas temperature. The temperature of the oil is calculated in the oil system model. Inputs to the model are the position of the cowl flaps, the engine cooling characteristics (by function generation) and variables from the engine subprogrammes such air temperature etc.

In Fig. 3.4 the cylinder head temperature is computed as per [2.62] by integration; Exhaust gas temperature is computed using [2.65]. Thermal efficiency is based on [2.66] taking into account fuel flow and power.

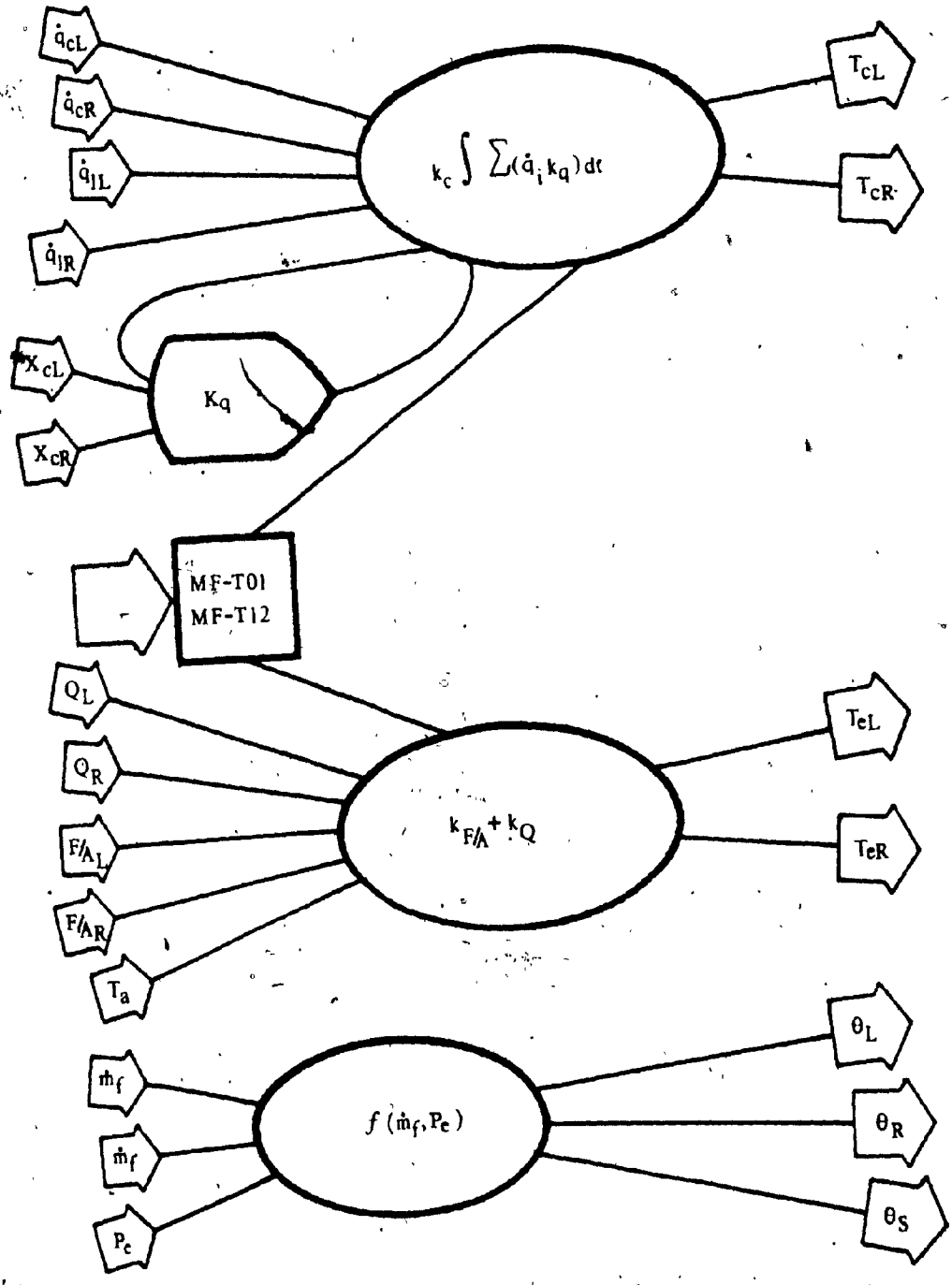


Fig. 3.3 Thermal Module outline.

THERMAL MODEL

Model variables.

Variable	Description	Units
$\dot{q}_{cL}$	Cylinder heat flow rate Left	Kcal/sec
$\dot{q}_{cR}$	Cylinder heat flow rate Right	Kcal/sec
T_{cL}	Cylinder Head temperature Left	K
T_{cR}	Cylinder Head temperature Right	K
$\dot{q}_{lL}$	Heat loss flow rate Left engine	Kcal/sec
$\dot{q}_{lR}$	Heat loss flow rate Right engine	Kcal/sec
k_{QL}	Cooling rate factor Left nacelle	(%)
k_{QR}	Cooling rate factor Right nacelle	(%)
$\dot{q}_{eL}$	Exhaust gas Heat flow rate Left	Kcal/sec
$\dot{q}_{eR}$	Exhaust gas Heat flow rate Right	Kcal/sec
T_{rL}	Exhaust temperature rise Left	K
T_{rR}	Exhaust temperature rise Right	K
T_{eL}	Exhaust gas temperature Left	K
T_{eR}	Exhaust gas temperature Right	K
T_{dL}	Displayable CHT Left engine	K
T_{dR}	Displayable CHT Right engine	K
T_{gL}	Displayable EGT Left engine	K
T_{gR}	Displayable EGT Right engine	K
k_q	Cowling effect (FUNGEN)	(%)
θ_L	Left engine thermal efficiency	(%)
θ_R	Right engine thermal efficiency	(%)

Malfunction Variables.

Variable	Description	Units
MFT01	Inoperative cowl flaps Left	flag
MFT02	Inoperative cowl flaps Right	flag
MFT03	Seized Cowl flap mechanism Left	flag
MFT04	Seized Cowl flap mechanism Right	flag
MFT05	CHT ind. out of order Left	flag
MFT06	CHT ind. out of order Right	flag
MFT07	CHT false indication Left	K
MFT08	CHT false indication Right	K
MFT09	EGT ind. out of order Left	flag
MFT10	EGT ind. out order Right	flag
MFT11	EGT false indication Left	K
MFT12	EGT false indication Right	K

3.1.4 The Ignition system module.

The function of the simulated ignition systems of both engines is described in Fig. 3.5. Input originates from the ignition switches of the aircraft panel and some engine system variables. This system is mostly of logical nature and is modelled as such. Most of its variables are discrete parameters. Two system flags are used to show that the engines are on and running. The start sequence is reproduced by a timing loop for each engine.

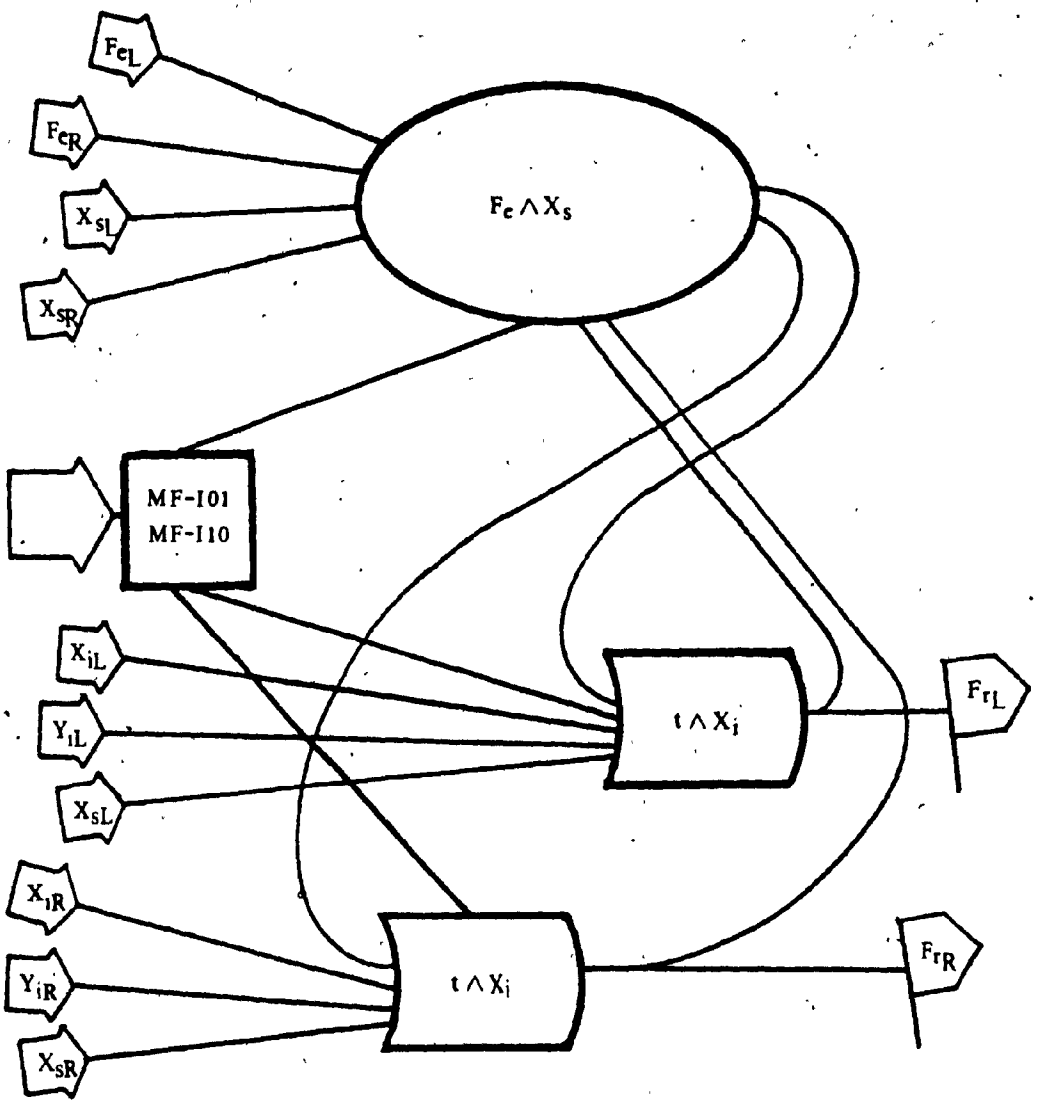


Fig. 3.4 Ignition system Module outline.

IGNITION SYSTEM System Variables.		
Variable	Description	Unit
Fr_L Fr_R	Left engine on and running Right engine on and running	flag flag
Malfunction Variables.		
Variable	Description	Unit
MFI01	Magn./Distr. defective Left Left	flag
MFI02	Magn./Distr. defective Right Left	flag
MFI03	Magn./Distr. defective Left Right	flag
MFI04	Magn./Distr. defective Right Right	flag
MFI05	Spark plugs out of order Left	L/R
MFI06	Spark plugs out of order Right	L/R
MFI07	Ignition switch defective Left	flag
MFI08	Ignition switch defective Right	flag

3.1.5 The Oil System Module.

As discussed in Chapter two, oil temperature and pressure are two important parameters that enable the pilot to monitor engine status. These quantities are computed by this module along with the viscous friction of the oil on the revolving crankshaft and the reciprocating piston rods. Oil temperature rises due to that friction and heat transferred by coming in contact with components of higher temperature such as the rods, the pistons, and cylinder walls. The engine is assumed to incorporate an automatic mechanism that when oil temperature reaches a certain level it routes the fluid through a heat exchanger where oil heat is dissipated in the atmosphere. Oil viscocity drops exponentially with oil temperature rise. This effect is reproduced by function generation.

In Fig. 3.6 the expression used for oil pressure computation is that of [2.70]; the OGVFST function provides the viscous friction factor. Equation [2.71] is used to compute the oil temperature with the OGTMPR variable supplied by the function generator. Malfunctions include overpressure and overheating.

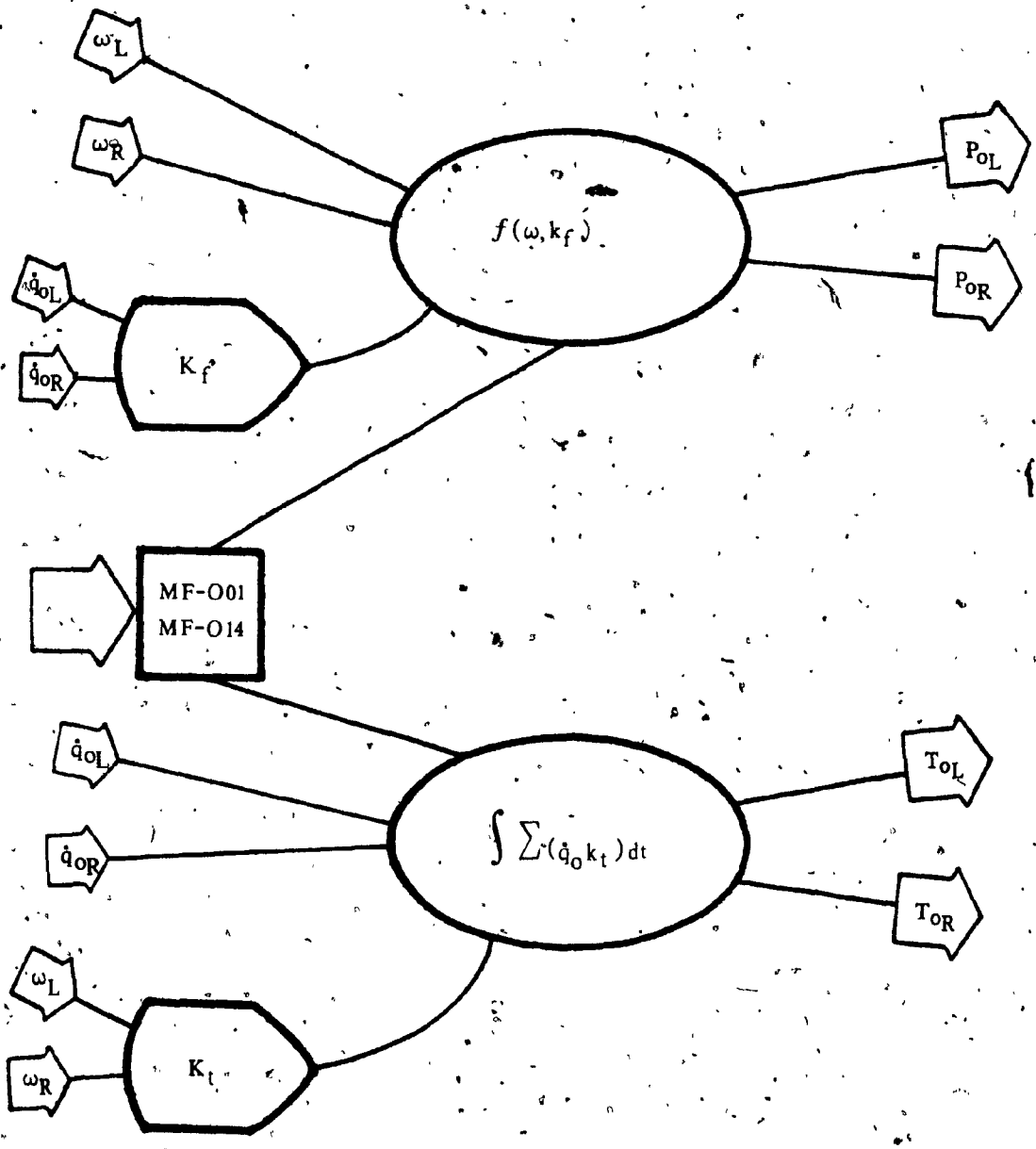


Fig. 3.5 Oil system Module outline.

OIL SYSTEM

System Variables.

Variable	Description	Unit
T_{oL}	Oil temperature Left engine	K
T_{oR}	Oil temperature Right engine	K
P_{oL}	Oil pressure Left engine	kg/m
P_{oR}	Oil pressure Right engine	kg/m
$\dot{q}_{oL}$	Oil heat gain flow rate Left	Kcal/sec
$\dot{q}_{oR}$	Oil heat gain flow rate Right	Kcal/sec
$\dot{q}_{lL}$	Oil heat loss flow rate Left	Kcal/sec
$\dot{q}_{lR}$	Oil heat loss flow rate Right	Kcal/sec
k_f	Viscous friction factor (FUNGEN)	Nmsec
k_t	Oil Temp. rise - mech. (FUNGEN)	K
T_{dL}	Displayed oil Temperature Left	K
T_{dR}	Displayed oil Temperature Right	K
P_{sL}	Displayed oil Pressure Left	kg/m
P_{sR}	Displayed oil Pressure Right	kg/m

Malfunction Variables.

Variable	Description	Unit
MF001	Heat exchanger/valve def. Left	flag
MF002	Heat exchanger/valve def. Right	flag
MF003	H/E valve stuck open Left	flag
MF004	H/E valve stuck open Right	flag
MF005	Oil temp. defective Left	flag
MF006	Oil temp. defective Right	flag
MF007	False temp. indication Left	K
MF008	False temp. indication Right	K
MF009	Oil press. ind. defective Left	flag
MF010	Oil press. ind. defective Right	flag
MF011	False press. Indication left	kg/m
MF012	False press. Indication Right	Kg/m
MF013	Oil circuit open Left	flag
MF014	Oil circuit open Right	flag

3.1.6 The Propulsion System Module.

The propulsion system is simulated as shown in Fig.3.7; It is considered as a disturbance torque input to the engine model along with the propeller governor. The propeller torque is calculated from data pertinent to the propeller by function generation; since the chart is expressed in terms of propeller power, torque is derived from that quantity. Input to the model are the blade pitch settings and variables from the engine and external models.

Basically the required torque expression is derived in the module by computing the propeller's advance ratio by [2.48] and the blade pitch angle by [2.47]. The coefficient of propeller power is obtained from the propeller chart by the JGPWRP function. This quantity is then converted into the propeller demand torque by [2.45]. Since the propeller control system (governor) draws power in the form of pressurised oil to control blade pitch, its value is calculated and added to the demand torque figure.

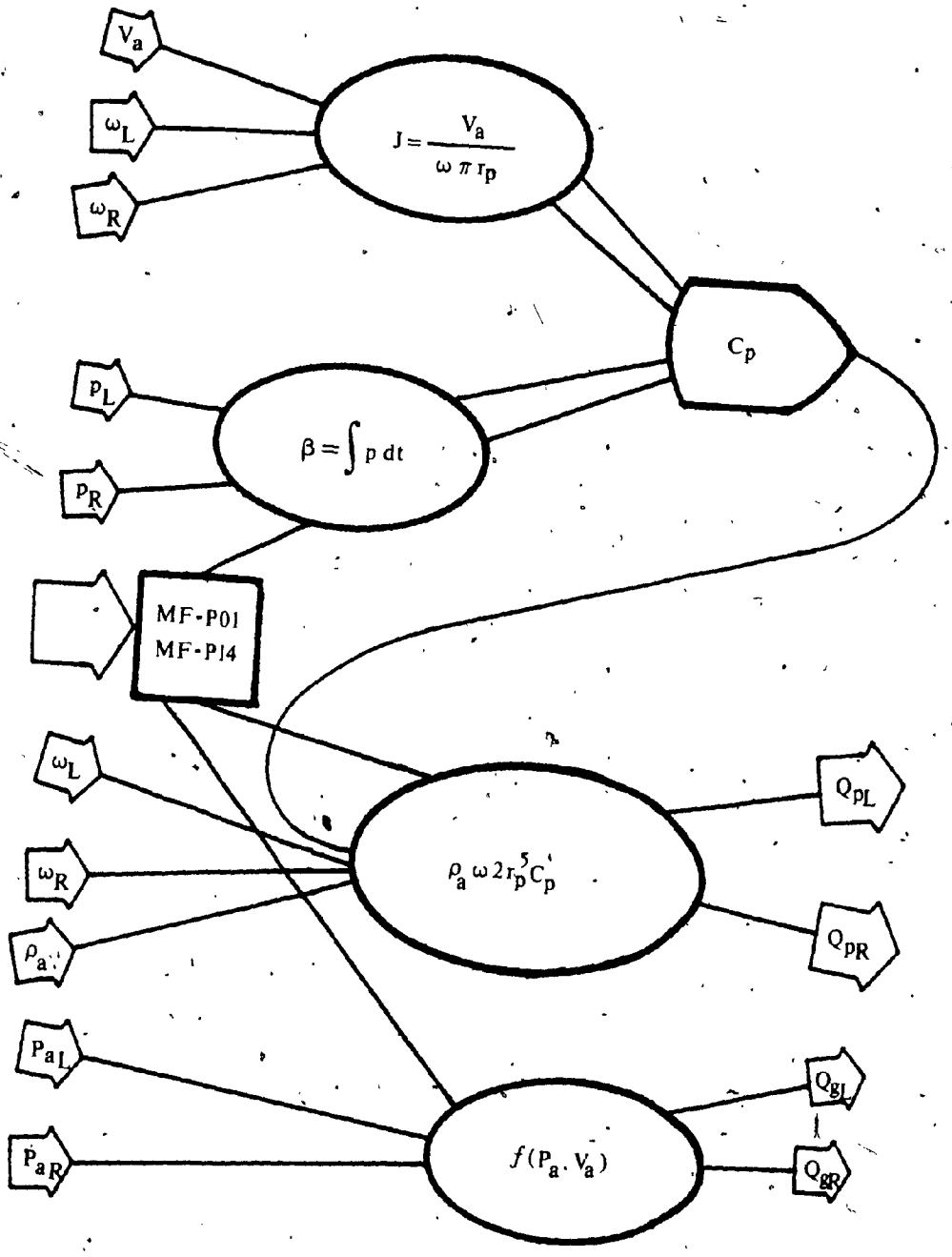


Fig. 3.6 Propulsion system Module outline.

PROPULSION SYSTEM

System Variables.

Variable	Description	Unit
J_L	Propeller Advance ratio Left	d/l
J_R	Propeller Advance ratio Right	d/l
T_L	Propeller Thrust Left	kg
T_R	Propeller Thrust Right	kg
Q_{pL}	Propeller Torque Left	Nm
Q_{pR}	Propeller Torque Right	Nm
P_{pL}	Propeller Power Left	Jle/sec
P_{pR}	Propeller Power Right	Jle/sec
C_p	Coefficient of Power	d/l
β_L	Blade angle left propeller	deg.
β_R	Blade angle right propeller	deg.
p_L	Blade pitch left propeller	m
p_R	Blade pitch right angle	m

d/l : dimensionless quantity.

Malfunction Variables.

Variable	Description	Unit
MFP01	Uncontrollable pitch Left	flag
MFP02	Uncontrollable pitch Right	flag
MFP03	Pitch linkage error Left	flag
MFP04	Pitch linkage error Right	flag
MFP05	False linkage factor Left	(%)
MFP06	False linkage factor Right	(%)
MFP07	Pitch stuck at feather Left	flag
MFP08	Pitch stuck at feather Right	flag
MFP09	Pitch stuck at high Left	flag
MFP10	Pitch stuck at high Right	flag
MFP11	No feathering Left	flag
MFP12	No feathering Right	flag
MFP13	Pitch stuck at low Left	flag
MFP14	Pitch stuck at low Right	flag

3.1.7 The Aeroengine System Module.

The aeroengine module is shown in Fig. 3.8. All the external parameters are derived at this point and can be used in the aeroengine model. Angular velocity is calculated by integrating angular acceleration using [2.26]. After computing each engine's brake torque by [2.25], all the disturbance inputs in terms of loading torque are subtracted so the system becomes balanced. Frictional losses, Coulomb and viscous, are introduced in the torque calculations. Carburator icing is included in both engine systems by function generation (EGICNG). Other function generation covers the viscous friction factor and the torque efficiency computation. Two system flags are used to signal the presence of ice formation conditions; this computation is based on [2.8] and [2.15]. The manifold air density is then calculated using one -economy minded- pass of a Newton approximation of [2.12] as shown in [2.13]. Displayable manifold pressure is computed by [2.21]; finally torque and angular velocity are computed by [2.24], [2.25] and the 'end-of-the-cycle' [2.26] equation.

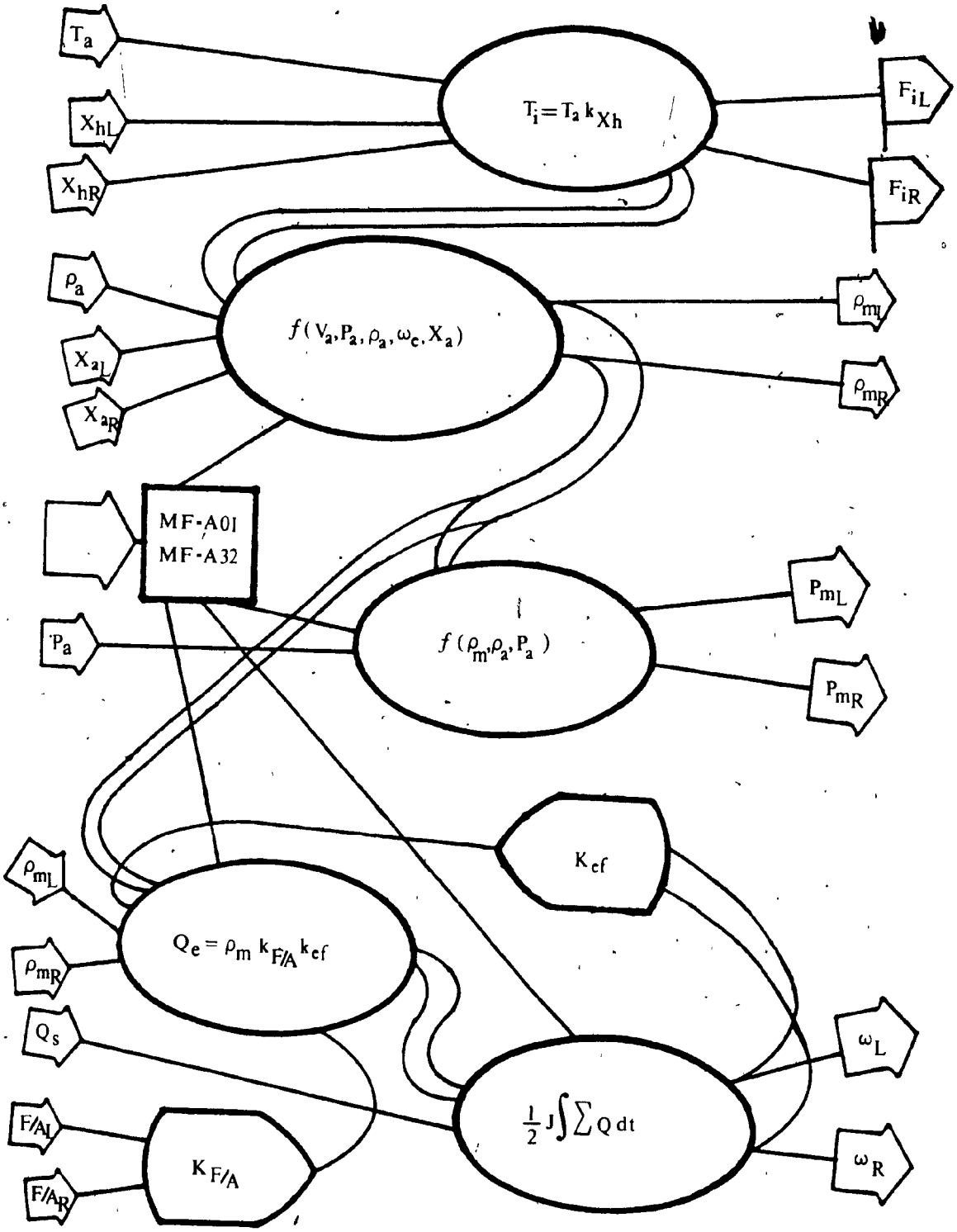


Fig. 3.7 Aeroengine system Module outline.

AEROENGINE SYSTEM.

System Variables.

Variable	Description	Unit
ω_{eL}	Angular Velocity Left engine	rad/sec
ω_{eR}	Angular Velocity Right engine	rad/sec
ω_{dL}	Displayable angular velocity Left	rad/sec
ω_{dR}	Displayable angular velocity Right	rad/sec
Q_{eL}	Gross generated torque Left	Nm
Q_{eR}	Gross generated torque Right	Nm
Q_{bL}	Brake torque Left engine	Nm
Q_{bR}	Brake torque Right engine	Nm
ρ_{aL}	Carburator inlet air density Left	kg/m
ρ_{aP}	Carburator inlet air density Right	kg/m
T_{aL}	Carburator inlet air temperature L	K
T_{aP}	Carburator inlet air temperature R	K
v_{aL}	Carburator inlet air velocity Left	m/sec
v_{aP}	Carburator inlet air velocity Right	m/sec
T_{vL}	Venturi tube air temperature Left	K
T_{vP}	Venturi tube air temperature Right	K
ρ_{vL}	Venturi tube air density Left	kg/m
ρ_{vR}	Venturi tube air density Right	kg/m
v_{vL}	Venturi tube air velocity Left	m/sec
v_{vR}	Venturi tube air velocity Right	m/sec
P_{mL}	Manifold air pressure Left	kg/m
P_{mR}	Manifold air pressure Right	kg/m
ρ_{mL}	Manifold air density Left	kg/m
ρ_{mR}	Manifold air density Right	kg/m
v_{mL}	Manifold air velocity Left	m/sec
v_{mR}	Manifold air velocity Right	m/sec
$P_{\alpha L}$	Ambient air pressure	Pa
T_{α}	Ambient air temperature	K
T_{δ}	Ambient air Dew point	K
ρ_{α}	Ambient air density	kg/m
X_{tL}	Throttle lever position Left	(%)
X_{tR}	Throttle lever position Right	(%)
X_{mL}	Mixture lever position Left	(%)
X_{mR}	Mixture lever position Right	(%)
X_{hL}	Carburator Heat lever pos. Left	(%)
X_{hR}	Carburator Heat lever pos. Right	(%)

AEROENGINE SYSTEM

Malfunction variables.

Variable	Description	Unit
MFA01	Inoperative RPM meter Left	flag
MFA02	Inoperative RPM meter Right	flag
MFA03	Invalid RPM indication Left	rpm
MFA04	Invalid RPM indication Right	rpm
MFA05	Intermittent RPM meter Left	flag
MFA06	Intermittent RPM meter Right	flag
MFA07	Inop. Manifold pressure meter Left	flag
MFA08	Inop. Manifold pressure meter Right	flag
MFA09	False manifold press. indication L	(%)
MFA10	False manifold press. indication R	(%)
MFA11	Intermittent manifold pressure L	flag
MFA12	Intermittent manifold pressure R	flag
MFA13	Carburator icing Left	flag
MFA14	Carburator icing Right	flag
MFA15	Icing congestion rate Left	m/sec
MFA16	Icing congestion rate Right	m/sec
MFA17	Inoperative Carburator Heat Left	flag
MFA18	Inoperative Carburator Heat Right	flag
MFA19	Stuck Carb. Heat lever Left	flag
MFA20	Stuck Carb. Heat lever Right	flag
MFA21	Stuck lever position Left	(%)
MFA22	Stuck lever position Right	(%)
MFA23	Throttle linkage error Left	(%)
MFA24	Throttle linkage error Right	(%)
MFA25	RPM hangup Left engine	flag
MFA26	RPM hangup Right engine	flag
MFA27	Hangup crossover point Left	rad/sec
MFA28	Hangup crossover point Right	rad/sec
MFA29	Time counter inoperative Left	flag
MFA30	Time counter inoperative Right	flag
MFA31	Invalid counter indication Left	hr
MFA32	Invalid counter indication Right	hr

chapter 4

SYSTEM SOFTWARE

Using a high level language to set up a dynamic system simulation often results in a situation where the available computing power is not made best use of. Since the compilation process is not always efficient, closed iteration is required to ensure that timing bounds and restrictions are respected. This requirement is controlled by the Real Time executive software. It is presented in this Chapter in block form since a detailed analysis is not within the bounds of this study. Use of a real time operating system with multitasking capacity is certainly desired since such a system would optimise operation provided there is room for it. The use of such a utility, though, would be justified when the simulation is used as part of a major system such as a flight simulator. The system software consists of two major components, the Data Control and the Real Time systems.

4.1 THE DATA CONTROL SYSTEM.

The next sections show the flow of control regarding data acquisition, conversion and transfer. Record data structures are used for easy manipulation of both kinds of data.

4.1.1 The Data Transfer System Module.

The displayable data is processed by this module in order that it be converted, scaled and limited in an a suitable format for output. These functions take place for analog values that are to drive the panel indicators. Discrete variables are grouped in byte and/or nibble formats for faster channelling through the Multibus to the latching ports where after they are broadcast singly to the driver circuitry. These output transfers are executed by Assembler subroutines in two byte formats so that most overhead processes are avoided. Following there is a list of the output variables and data structures that are used and then the data path is shown in block form; Figure 4.1 shows the flow and processing phases of the data stream.

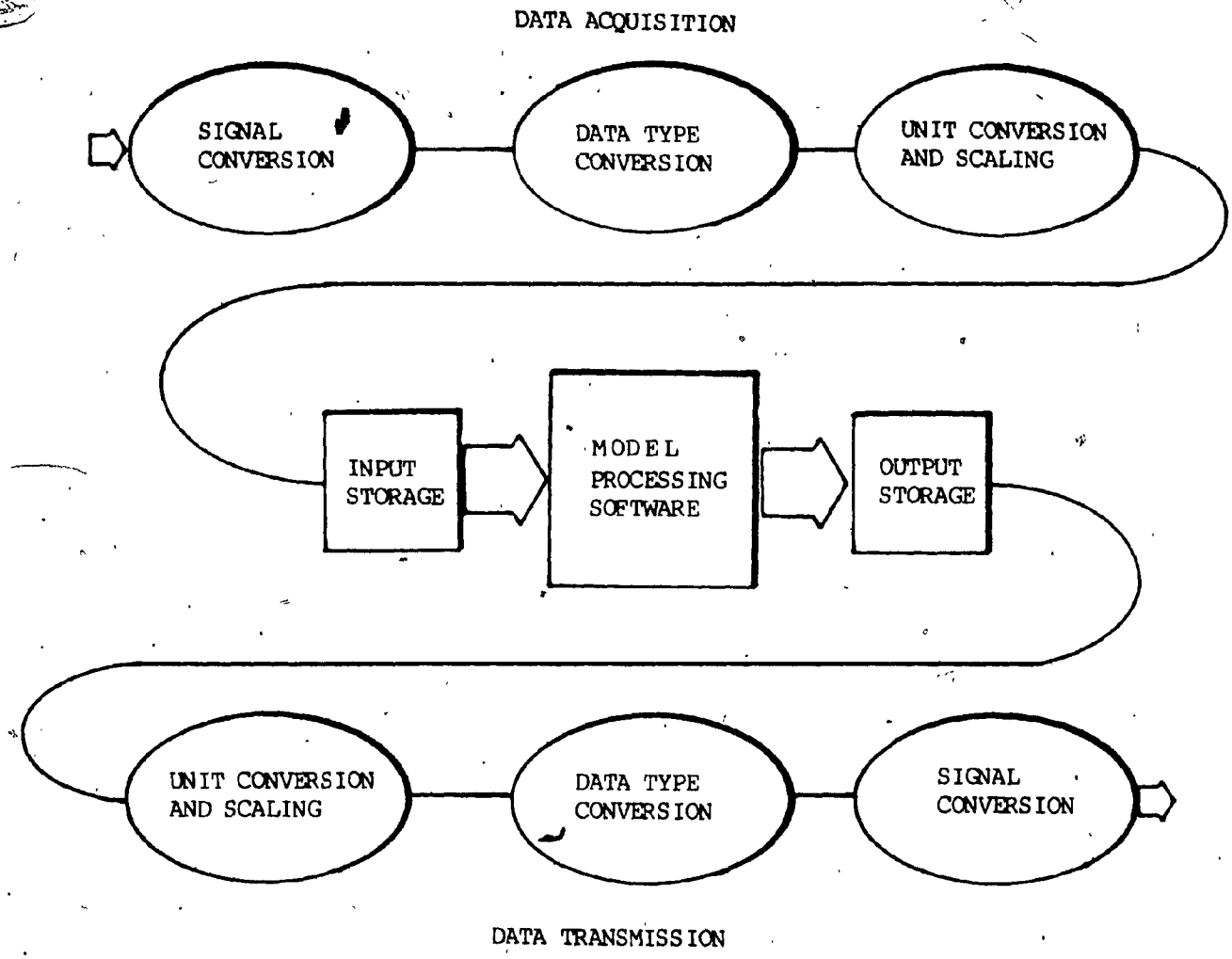


Fig. 4.1 The data control system.

DATA TRANSFER SYSTEM

System Variables.

Analogue O/P record ::= Analogue value field
::= Analogue port name field
::= Slew rate code field
::= Pointer field

Discrete O/P record ::= Discrete group field
::= Discrete port name field
::= Pointer field

Variable	Description	Unit
v_{eL}	RPM Left engine	rpm
v_{eR}	RPM Right engine	rpm
P_{mL}	Manifold pressure Left	in Hg
P_{mR}	Manifold pressure Right	in Hg
T_{xL}	Exhaust gas temperature Left	F
T_{xR}	Exhaust gas temperature Right	F
V_{fL}	Fuel quantity Left tank	US Gl
V_{fR}	Fuel quantity Right tank	US Gl
P_{fL}	Fuel pressure Left	psi
P_{fR}	Fuel pressure Right	psi
T_{oL}	Oil temperature Left	F
T_{oR}	Oil temperature Right	F
P_{oL}	Oil pressure Left	psi
P_{oR}	Oil pressure Right	psi
T_{cL}	Cylinder head temperature Left	F
T_{cR}	Cylinder head temperature Right	F
L_{aL}	Alternator load Left	%
L_{aR}	Alternator load Right	%
U_{vL}	Undervoltage Indicator Left	flag
U_{vR}	Undervoltage Indicator Right	flag
C_{pL}	CB Auxiliary Pump Left	flag
C_{pR}	CB Auxiliary pump Right	flag
C_{sL}	CB Start and prime Left	flag
C_{sR}	CB Start and prime Right	flag
C_{eL}	CB Engine instrument cluster Left	flag
C_{eR}	CB Engine instrument cluster Right	flag
C_{iL}	CB Bus Isolation Left	flag
C_{iR}	CB Bus Isolation Right	flag
C_{aL}	CB Alternator stator Left	flag
C_{aR}	CB Alternator stator Right	flag
C_t	CB Bus Tie	flag

4.1.2 The Data Aquisition System Module.

Both analogue and discrete data types are handled in this module. The analogue data is read in by an Assembler routine, entered in integer format and stored in a record data structure. The same applies to the digital input which is afterwards masked and decomposed into the respective Boolean variables. Analogue variables are all of type Real after internal conversion and scaling.

DATA ACQUISITION SYSTEM.

System Variables.

```
Analogue I/P record ::= Analogue value field
                    ::= Port name field
                    ::= Pointer field
Discrete I/P record ::= Group value field
                    ::= Port name field
                    ::= Pointer field
```

Variable	Description	Unit
K_{tL}	Left Throttle input	(%)
K_{tR}	Right Throttle input	(%)
$P\partial L$	Left Pitch input	(%)
$P\partial R$	Right Pitch input	(%)
M_{mL}	Left Mixture input	(%)
M_{mR}	Right Mixture input	(%)
H_{cL}	Left Carb. Heat input	(%)
H_{cR}	Right Carb. Heat input	(%)
F_{cL}	Left Cowl Flap input	(%)
F_{cR}	Right Cowl Flap input	(%)

4.2 THE REAL TIME SYSTEM.

The Real Time executive process is presented in this section of the study. Primarily, control originates from external input, viz: a terminal unit or a peripheral controller of a flight simulator. The Master Control program accepts commands through the external communications module which monitors the serial input channel. It may also display Status information, statistical data regarding the simulation process or the operation of the computer itself. Malfunctions can be inserted or removed at will in the same manner as system commands. The software modules that comprise the Real Time executive system are those of Master Control, Statistical Data Collection, Priority Control File, External Communications, Execution Control, Function Generation, Initialisation Routine, Malfunction Control and Input/Output Control. They are all described briefly at this point. Fig. 4.2 shows the modules of the total system and their control relationship.

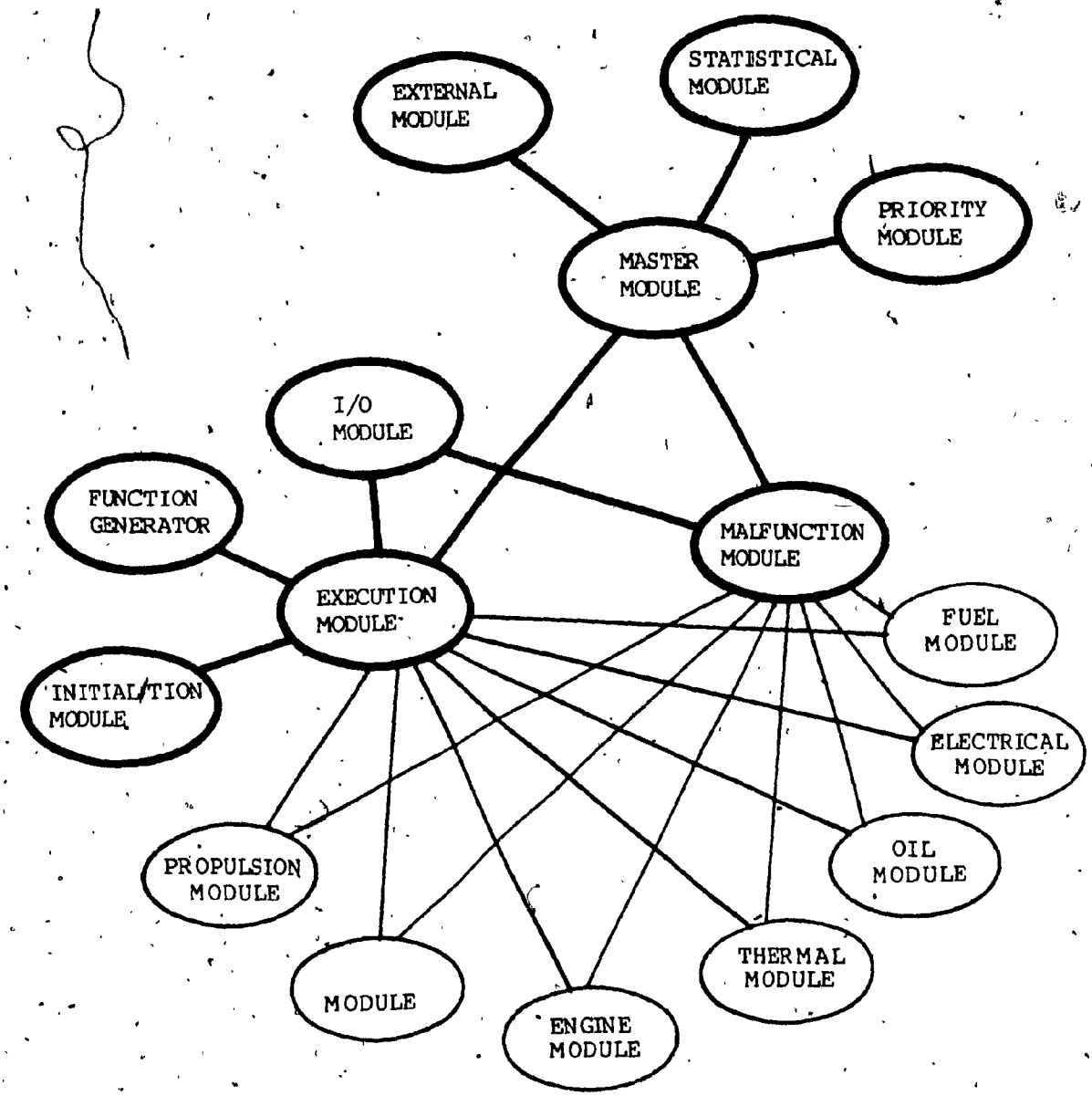


Fig. 4.2 The real time executive system.

4.2.1 The Master Control Module.

The master control module functions as the system monitor at a supervisory level and establishes the operational and policy directives to the other system modules. Valid input from the external communications controller is applied to the system via this subprogram. According to the accepted command, the master control module may start the simulation process by invoking the initialisation routine and the execution loop. It may also halt processing, alter or restore malfunctions, allow collection of input or output data, alter the prioritisation scheme and feed data to a printer or to a central facility. Since Real Time execution must adhere to established minima of performance, many of the master control functions such as the statistical data transfer are performed during the available spare computing time.

4.2.2 The External Communications Module.

As the name suggests this module controls the interaction of the simulation process with the external environment. Chiefly, it monitors the serial communications port of the computer system which can be connected to a terminal so that the operator can control the system. Otherwise, it can be attached to a flight simulator computer peripheral and accept control commands from that source. This module displays command descriptions, validates received input and requests the attention of the master control program when a control command is cleared. Depending on system design this module can also operate as a

parameter transferring mechanism. Data to a printer can be handled in a similar manner.

4.2.3 The Statistical Data Module.

A common training requirement is the collection and analysis of flight performance data for a variety of reasons, such as procedural compliance, optimised handling or general training recording. This type of data are referred to as 'snap shots' and they must be converted copies of the actual model variables. Regarding the powerplant, data pertinent to engine characteristics are usually collected and then plotted one versus another. This operation should not disturb the simulation process however and so it is performed on the margins of the available computing power. The module simply copies specified variables into record and file structures for later transfer or printing and plotting.

4.2.4 The Priority Control Module.

By this subprogram the priority policy of the system is established. Priority of execution for each model or system software module is tabled in a file that can be altered by the operator only or the controlling computer. There is a default prioritisation scheme that is enabled on system reset. To avoid complicated decision algorithms no two modules have the same execution priority. The default scheme is as follows:

DEFAULT PRIORITY ASSIGNMENT TABLE

Module	Condition	Type	Level
Master	Idle	System	0
	Run		9
External	Idle	System	1
	Run		10
Statistical	Idle	System	7
	Run		12
Priority	Idle	System	3
	Run		13
Malfunctions	Idle	System	6
	Run		8
Execution	Idle	System	5
	Run		0
I/O Control	Idle	System	4
	Run		1
Fun. Gen.	Run	System	5
Initialisation	Idle	System	2
Thermal	Run	Model	8
Oil	Run	Model	11
Electrical	Run	Model	6
Fuel	Run	Model	4
Engine	Run	Model	2
Propulsion	Run	Model	3

This priority set up may vary with the particular requirements of different modes of operation.

4.2.5 The Executive Module.

Execution of system and model software is managed by this module. It establishes the system timing by setting up counters and initiating system timers with interrupt controlled response. If a software module has used its time allotment, it is suspended after it finishes up its current task -round robin or time slice. Certainly, this is to

be avoided carefully for the model software since it may result in erratic operation due to the fact that this Real Time executive does not approach the level of sophistication of a real time operating system. Similarly, system requests are preempted in the same way. To remedy the situation, the executive module processes a system request such as a data copy for display in the time intervals fitting the request's priority. The main function that the executive module has to perform is to run the simulation process. In a flight simulator there is a separate system dedicated to operator interaction (Instructor's station).

The model subprograms are invoked in a preset sequence dictated by their relative variable dependencies. As such, the order is as follows: Ignition, Oil, Thermal, Fuel, Electrical, Propulsion and Engines. These subprogrammes are preceeded by the Input module and at end the Output operations take place. This constitutes the basic iteration cycle that 'converts' the digital computer into an analogue-like machine. The iteration rate may vary but an acceptable minimum frequency of 20 Hz sets the lower end for this type of simulation. A 100 Hz rate can easily be reached with the iSBC 8612A microcomputer. Higher rates are of no advantage or benefit in simulating a relatively slow first order system.

4.2.6 The Initialisation Module.

Before the simulation process is run all the system variables are assigned suitable initial values. This is necessary due to the fact

that even if the compiler for the programming language used for coding resets all the variables to zero, there are numerous computations in the model software that will be undefined at system run time causing execution to be aborted. Although extensive, the initialisation routine is executed only once; on successful termination the executive module sets the system into simulation run mode.

4.2.7 The Malfunction Module.

Malfunctions such as carburator icing or fuel starvation have a definitive training value - if they happen in a flight simulator. The trainees can acquaint themselves with adversity and learn how to deal with it. Malfunctions have to be reproduced then by the computer system and this is accomplished by the malfunction control subprogram which is designed to accept input from the operator and alter model and system variables so that a certain component or function of the powerplant either fails or acts erraticaly. On system start all malfunctions are cleared. The operator may activate abnormal conditions and their characteristics as required through the terminal. The malfunction variables were listed with each system model. Malfunction profiles may also be preset by grouping various variables and values in file or record structures.

4.2.8 The Input / Output Module.

This subprogram invokes the I/O subroutines presented in the beginning of this Chapter. Before model execution all input data are

refreshed by read operations. At the end of execution all outputs are updated. Stale data problems are not encountered due to the relatively high iteration frequency that is used. If such a situation arises however, then the I/O operations may easily be interleaved with the model software.

4.2.9 The Function Generator Module.

This section presents the outline of the Function generation process. The extent of this system module was reduced as much as possible due to the reason that it consumes much computing power and memory. Function generation is necessary though for propeller data, cooling requirements, oil viscosity, and torque and fuel ratio efficiencies. Data is stored in the form of arrays and records in memory and when needed each function reproduces a value from the stored information. In the case that the value supplied is not present in memory then averaging and interpolation take effect to supply the required parameter. The functions generated are the following:

function HGCFLK : This function provides the heat loss rate at different baffle pressures, pressure altitudes and ambient temperatures according to the data of Fig. 4.3. It is used in the Thermal model.

function OGVSFT : Oil viscous friction is computed by this function since it varies exponentially with temperature changes. It is used in the Oil system module. Its output is a factor with which the maximum oil viscosity is to be multiplied (Fig. 4.4a).

function OGTMPR : This function computes the temperature rise of the oil used for lubrication with time and angular velocity and it is used in the engine model (Fig. 4.4b).

function JGPWRP : Propeller chart data (Fig. 2.5) are reproduced by this function. Given the blade pitch angle and the propeller advance ratio it supplies the coefficient of propeller power which is used in turn to derive the propeller torque in the propulsion model.

function EGEFTQ : Engine volumetric efficiency (Fig.4.5) data is supplied by this function to the engine system models.

function EGFAMX : Fuel / Air mixture effects on the generated torque by each engine are reproduced by this function in the form of an efficiency coefficient; the key variable is the Mixture lever position for each engine system; (Fig. 4.6).

function EGICNG: The Dew point of the mixture is computed by this function using the psychrometric chart of Chapter two and a 'steam' table (Appendix C). Graphical forms of the above functions appear in the next pages.

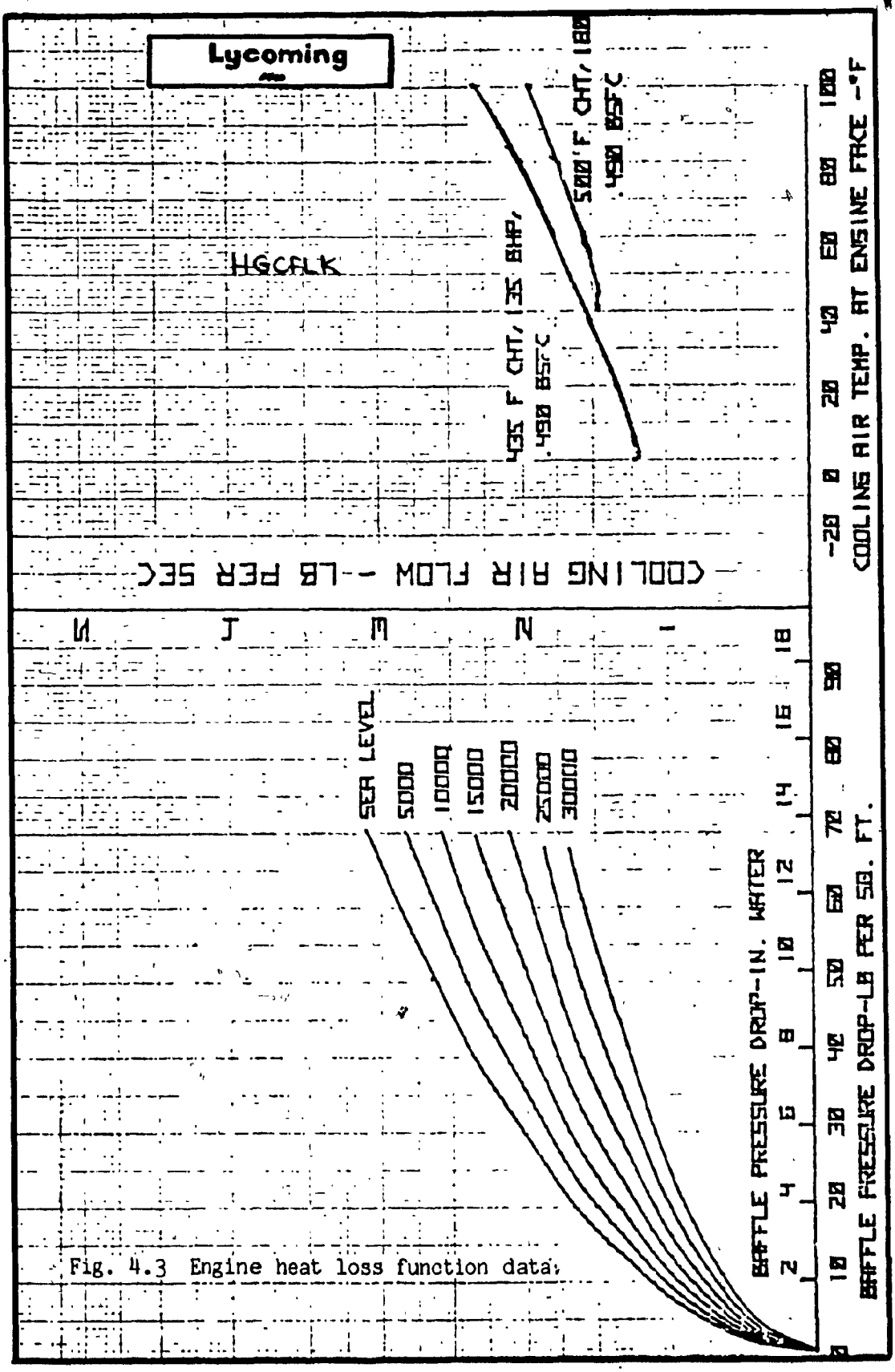


Fig. 4.3 Engine heat loss function data.

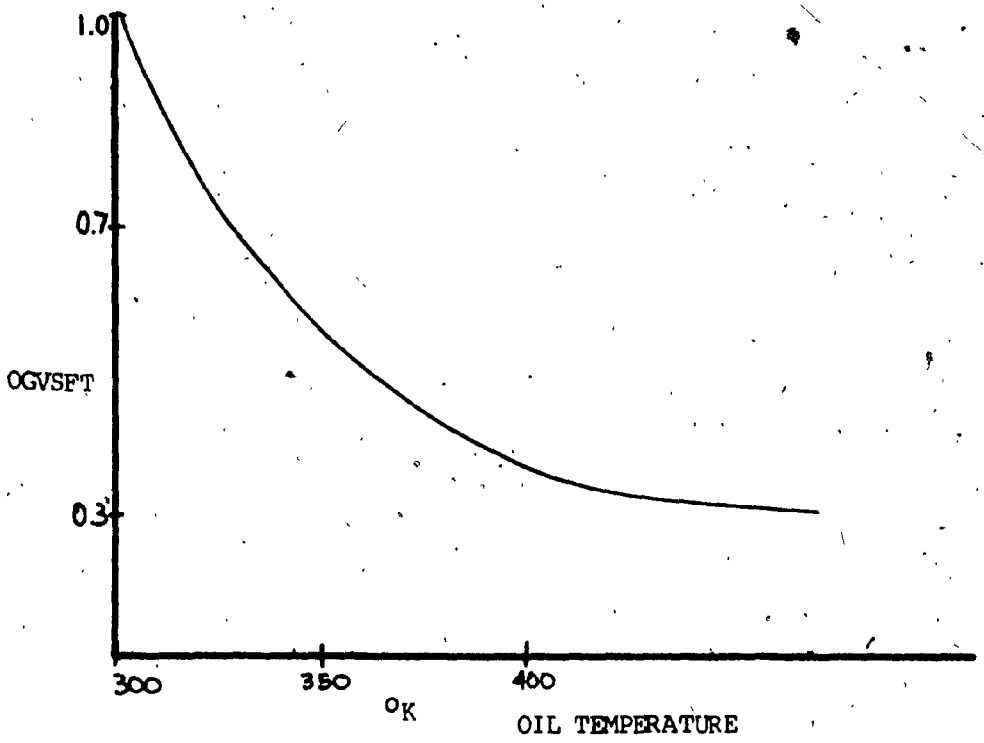


Fig. 4.4a Oil viscous friction function data.

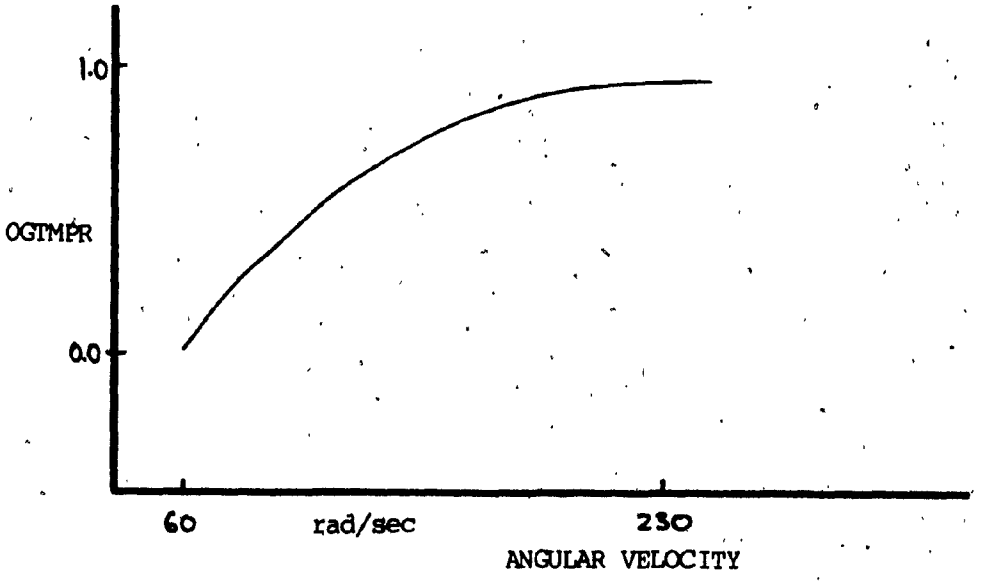


Fig. 4.4b Oil temperature rise function data.

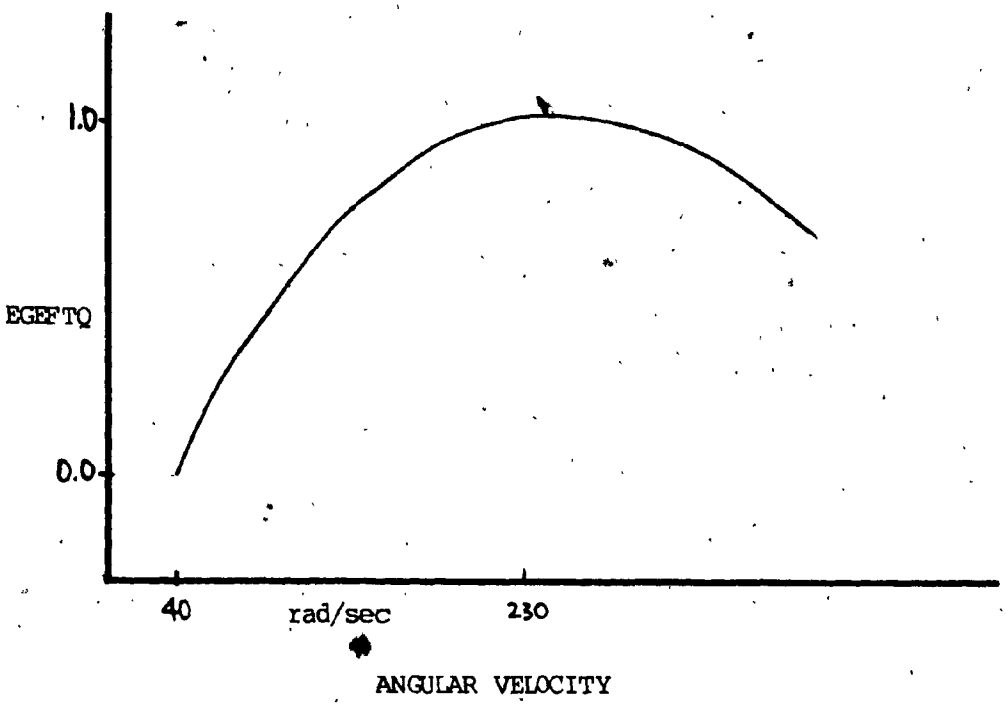


Fig. 4.5 Engine volumetric efficiency data.

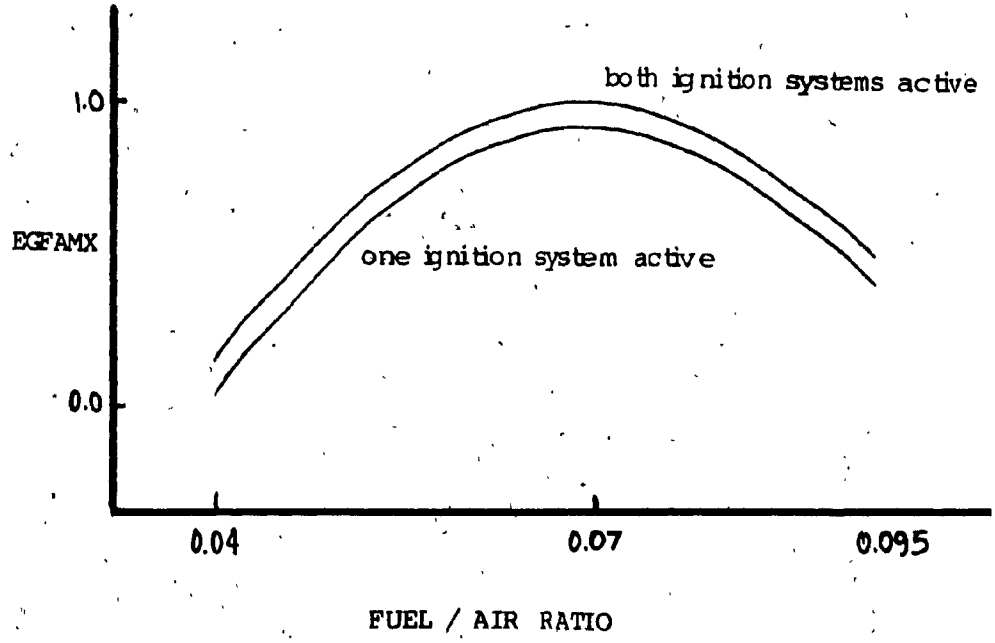


Fig. 4.6 Fuel / Air ratio efficiency function data.

chapter 5

SYSTEM HARDWARE

The system hardware developed for the implementation of the simulation is presented briefly in this Chapter. It consists primarily of the necessary interface to the computer board and the peripheral units. The driver circuitry for the indicators is presented as well as the engine and propeller sound generator. Control connections to the external environment are made via the serial and parallel ports of the computer. The rest of the input and output circuitry is memory mapped on the Multibus. It must be noted that the bus arbitration scheme of the Multibus was frozen to dedicated priority operation and therefore the system operated as any ordinary system bus. All transactions were initiated by the PU (master) without allowing any other device to take direct control of the bus. Data is transferred via the system I/O Interface which is examined at this point.

5.1 THE SYSTEM I/O INTERFACE.

Due to the nature of the iSBC 8612A computer the required number of I/O channels could be implemented only by external hardware. The onboard capacity is limited to one parallel and one serial port configuration which is used for external communications only. For that reason the Multibus addressing space is used in memory mapping all external hardware. Taking advantage of the sixteen bit data width available improved the system throughput since two eight bit data items could be transferred concurrently.

5.1.1 Data Input.

Data input to the computer consists of eight or sixteen bit quantities that have been acquired from the analogue to digital data conversion system or from the digital buffer system. The onboard I/O facilities are not included here. The input data represents the position of the engine and propeller control levers and the status of the various system switches and circuit breakers. The state of the latter has to be known since they are often used as switches and they can be reset.

5.1.1.1 The Analogue Data Input.

Data of analogue nature is generated from the transducers of the

engine and control levers. These devices are mechanically linked with the controls in such a way that there is an almost linear transformation of the rotational input into translational motion. The transducers are rectilinear potentiometers that convert the position of each control lever to analogue voltages that are subsequently processed by the conversion circuitry. Then they are read into the computer via the Multibus data path. Differential transformers (LVDTs) could have been used but the added precision was not considered to merit the extra expense. There are ten such lever - transducer arrangements as follows; (Fig. 5.1).

i. **Two Power control levers:** / Idle to Take off /.

ii. **Two Propeller control levers:** / Feather to High /.

iii. **Two Fuel / Air mixture levers:** / Lean to Rich /.

iv. **Two Carburator Heat levers:** / Off to Full /.

v. **Two Cowl Flap levers:** / Closed to full Open /.

The signal at the wiper of each potentiometer is input to a multiplexed analogue to digital converter (Fig. 5.2) and the two other terminals to reference voltages. The converter unit is a high speed, ready made device. Data for the transducers and the converter are listed in Appendix E. The circuit uses DC voltage levels of +15 Volts.

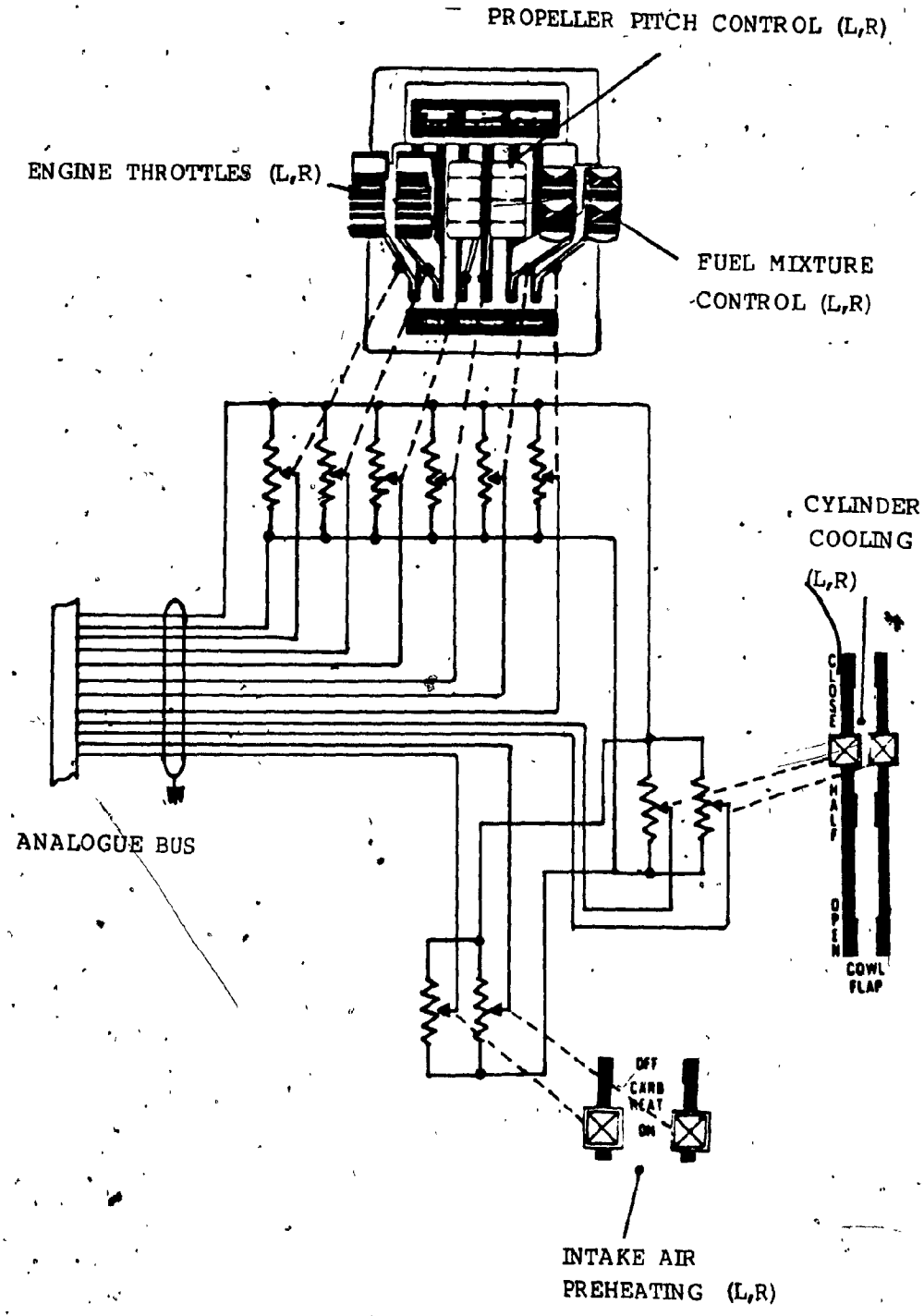


Fig. 5.1 The analogue input transducers.

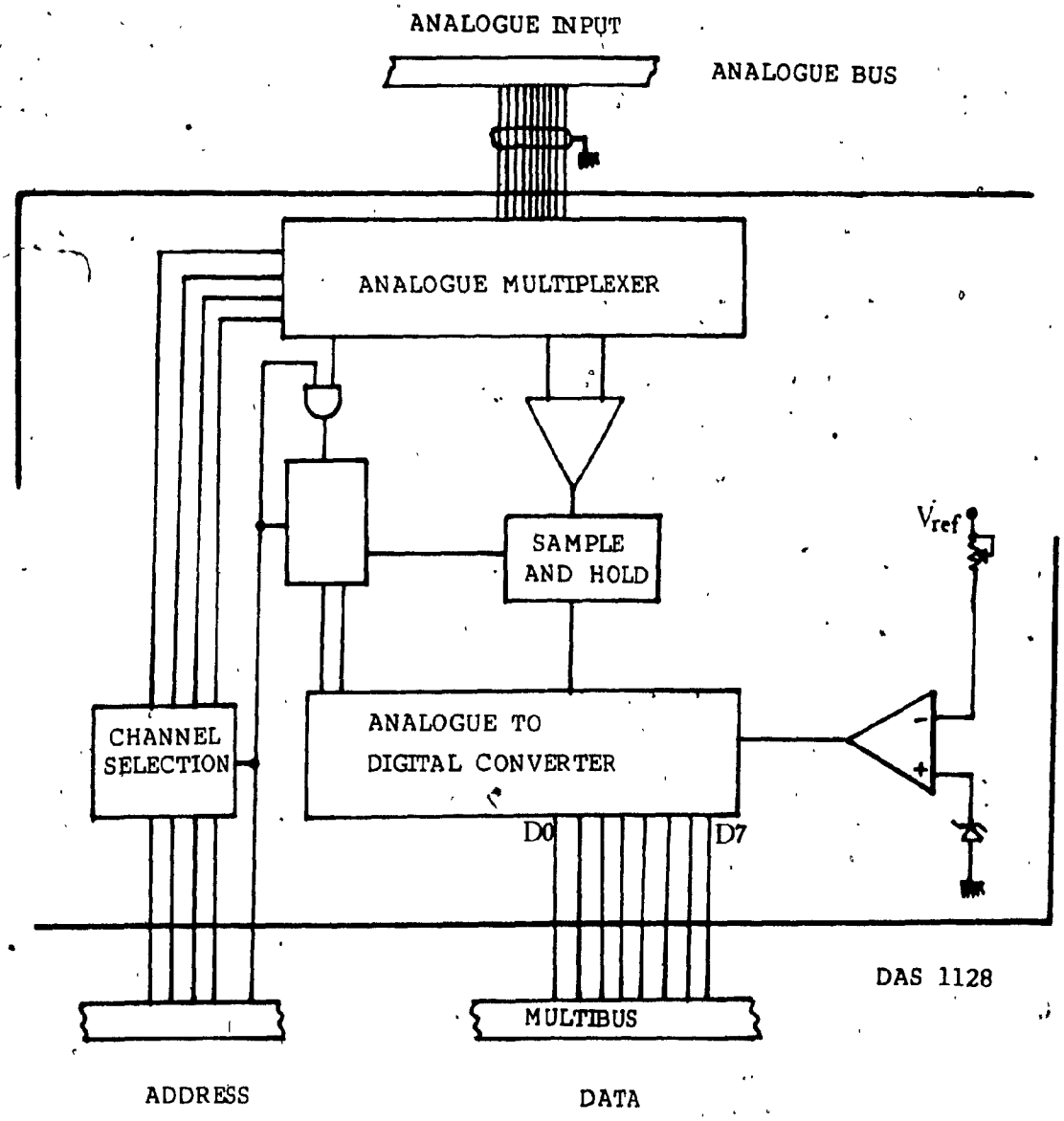


Fig. 5.2 The analogue data conversion circuit.

5.1.1.2 The Digital (Discrete) Data Input.

Discrete data is acquired from the following sources;

i. **Two Ignition switches:** / Off, Left, Right, Both, Prime /.

ii. **One Battery switch:** / Off, On /.

iii. **Two Alternator Field switches:** / Off, On /.

iv. **Two Auxiliary Pump switches:** / Off, On /.

v. **Eleven Circuit breakers:** / In, Out /.

The electrical state of each of the above devices (Fig. 5.3) is buffered and tranferred to the computer system over the Multibus. The buffer circuit (Fig. 5.4) consists of octal Schmidt trigger input, non inverting drivers. Data is tranferred in word format.

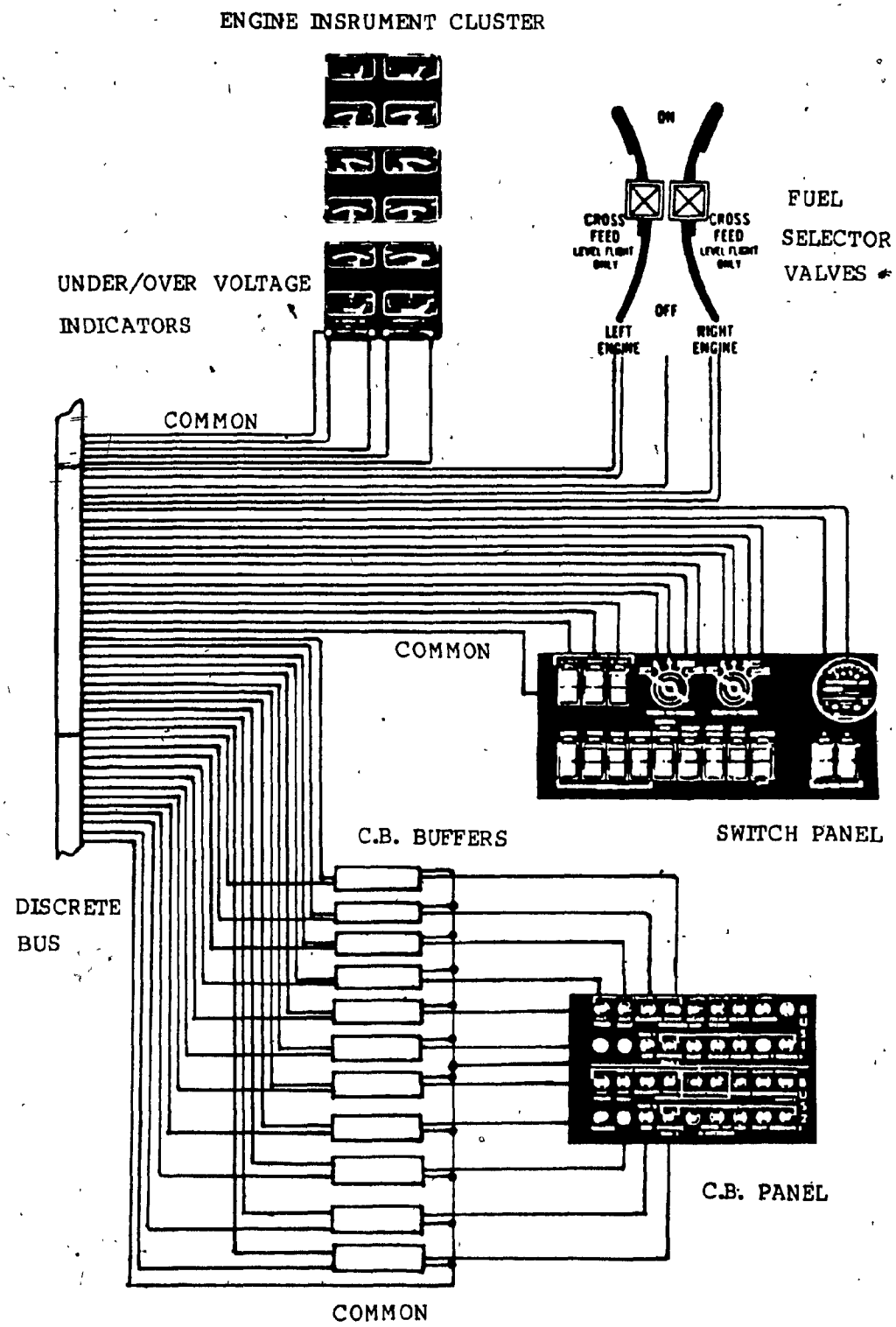


Fig. 5.3 The discrete input and output devices.

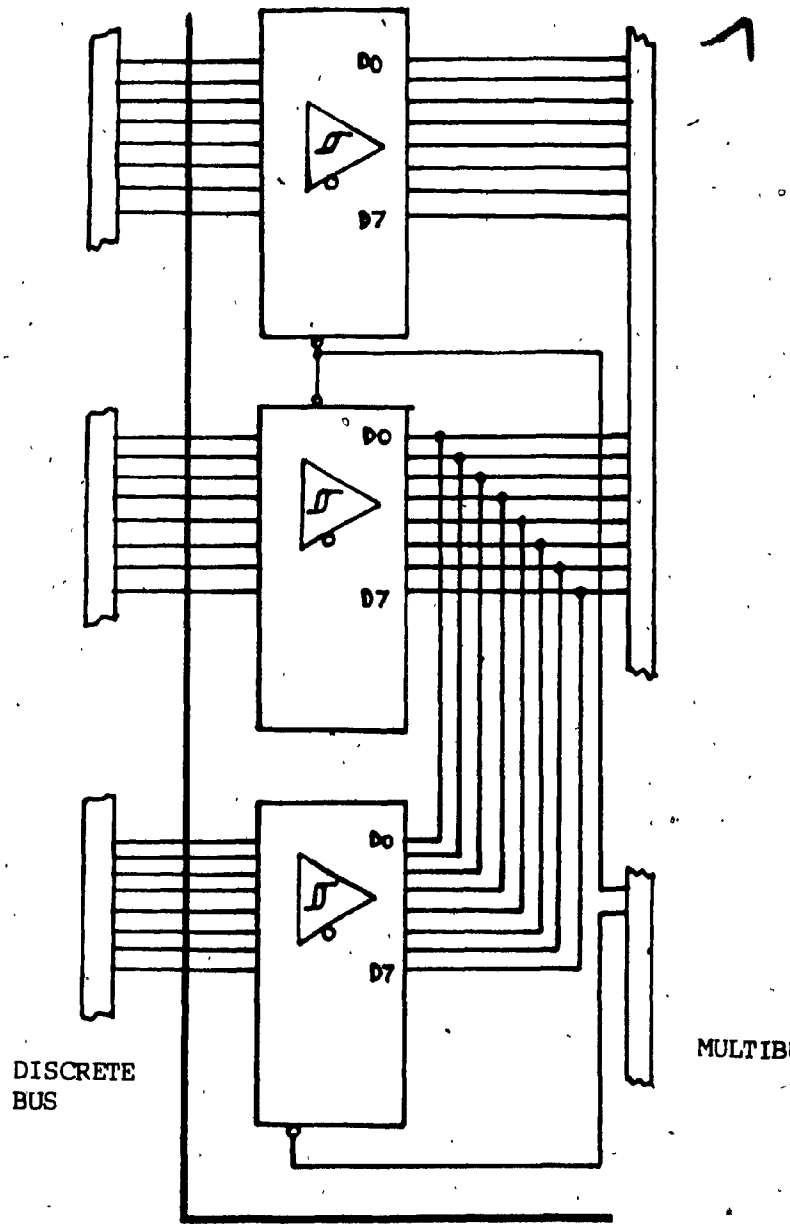


Fig. 5.4 The discrete input buffer circuit.

5.1.2 Data Output.

Two forms of output signals are produced by the peripheral output devices. These are analogue signals that control the engine instrument cluster and the gauge servomotors and digital signals that operate the discrete units, i.e. the circuit breakers etc. Output data transferred via the Multibus is latched and converted to analogue form or buffered and current sink amplified. In these forms it is then directly used in the panel indicators. Specifications for the above circuitry are listed in Appendices F and G.

5.1.2.1 The Analogue Data Output.

Analogue signals are necessary to drive the servomotor controllers of the RPM and Manifold pressure indicators, the engine status galvanometers as well as the voltage controlled oscillator of the sound generator. After conversion into eight bit quantities, the analogue values are transferred to three low cost converters that latch the data. These D/A converters can handle up to eight analogue output ports. Due to low current requirements the signals to the galvanometers are transferred directly while analogue data to the servomotors is buffered, amplified and controlled by the servocontroller circuitry to maintain a stable indication. Data is purposely slewed and oscillated to simulate 'real life' effects. The conversion circuitry is shown in Figs. 5.5 and 5.6.

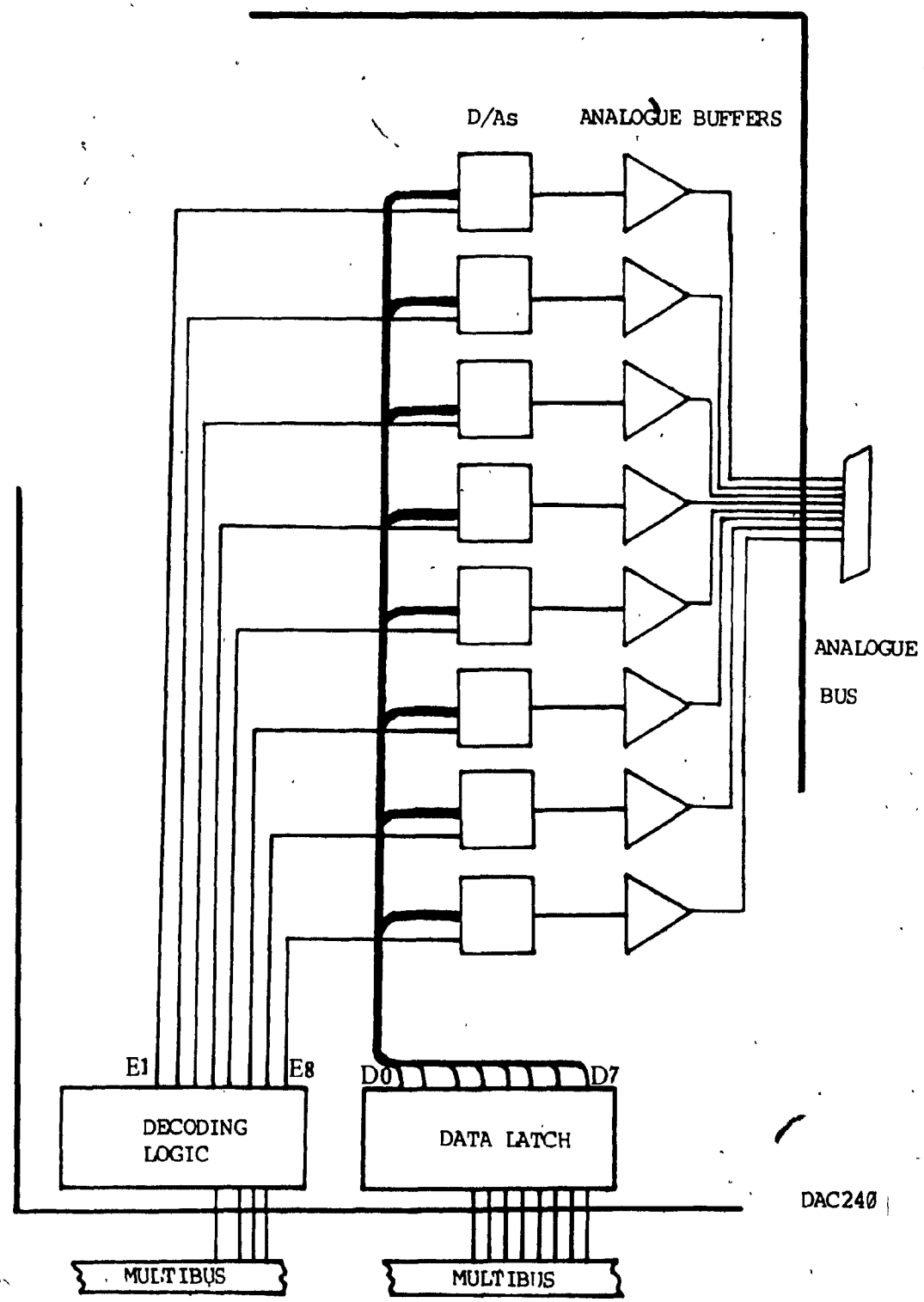


Fig. 5.5 The analogue output conversion circuit.

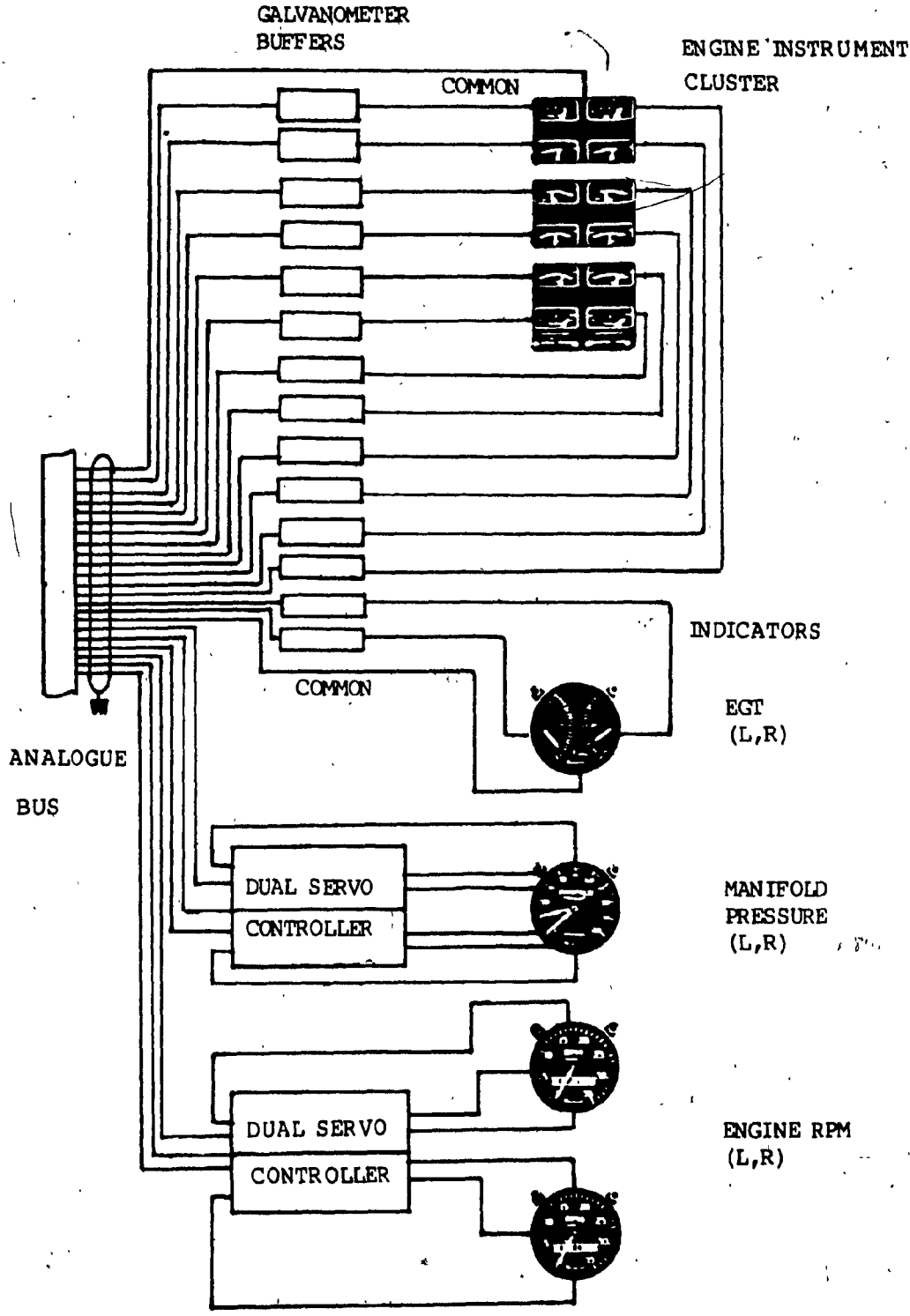


Fig. 5.6 The analogue output indicators.

5.1.2.2 The Digital Data Output.

Discrete data output is used to trip the circuit breakers of the panel when an overload condition occurs, and illuminate the irregular voltage indicators of the engine instrument cluster. Data transferred over the Multibus is latched in a low cost digital buffer circuit where it is decoded and expanded to octal latch form (Figs. 5.3, 5.7). These signals are subsequently transmitted to the discrete output devices. (Fig. 5.8)

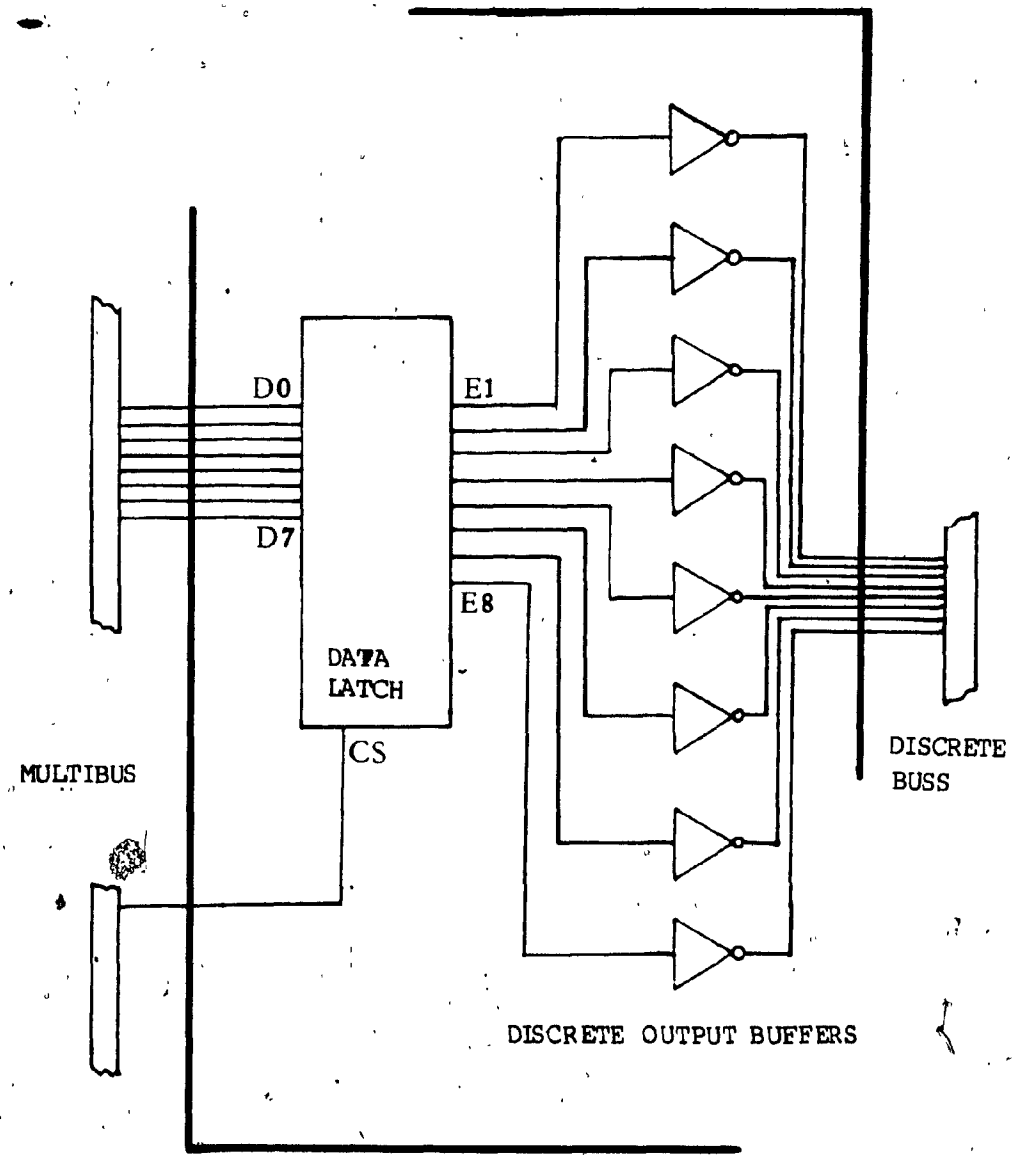


Fig. 5.7 The discrete output buffer circuit.

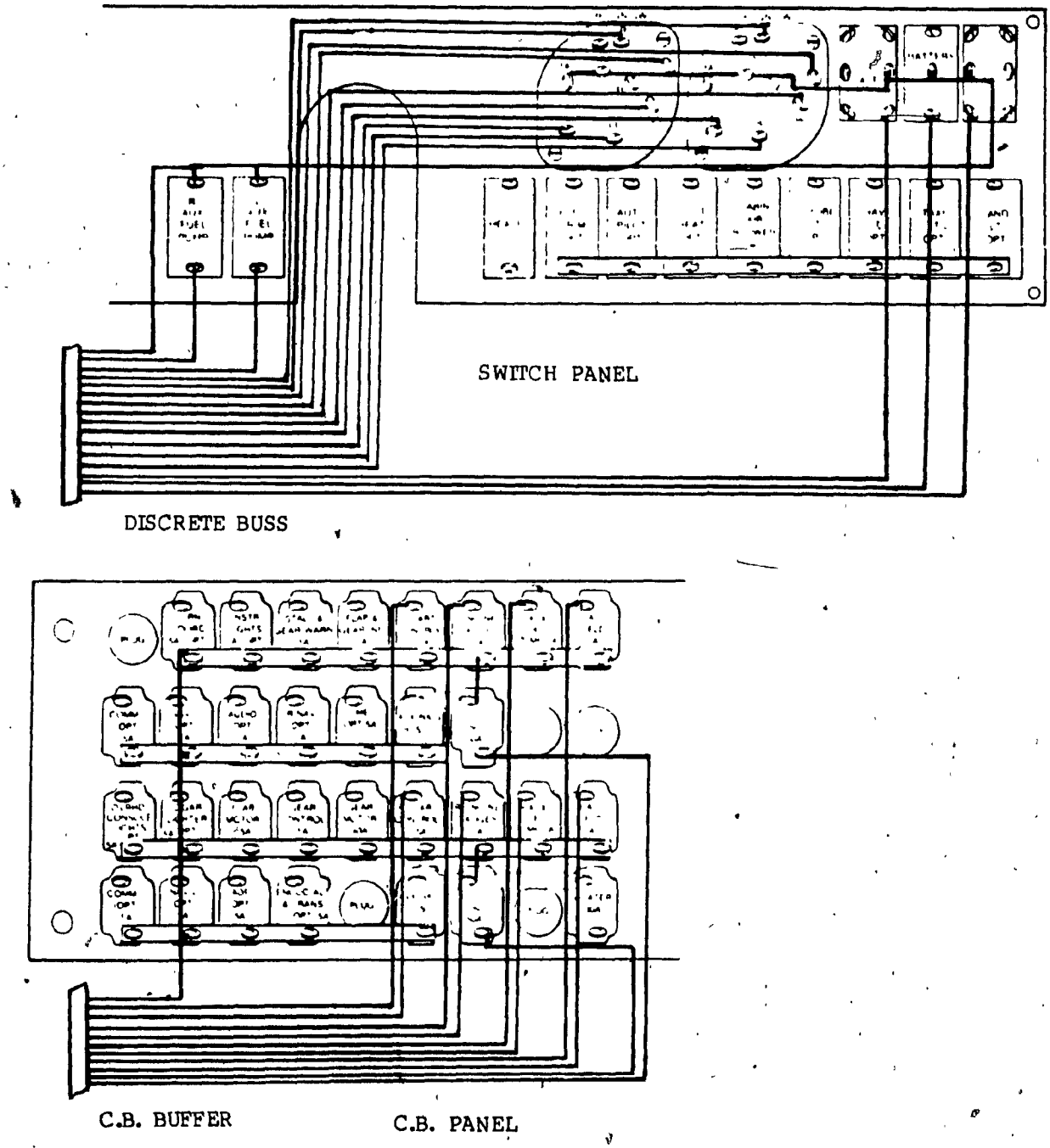


Fig. 5.8 The discrete input and output controls.

5.1.3 External Data Communications.

Since the iSBC 8612A microcomputer is equipped with a serial and a parallel port (Appendix D), that are not used to interface with the panel hardware, they are configured as follows:

i. External serial or parallel link with a host computer such as that of a flight simulator. In this scheme parameters can be passed very easily between remote software modules, commands issued etc.

ii. Connection of a terminal unit, a VDU or a printer. The process can be controlled directly and data displayed to the operator.

iii. A combination of the above to suit the requirements of the application. High data rates can be reached when using the parallel port whose twentyfour I/O lines can be configured differently, e.g. separate receive and transmit ports, handshaking control etc.

5.2 AUXILIARY HARDWARE.

Apart from the hardware modules for input and output operations and conversions, some other circuitry had to be developed to control and interface properly some of the panel I/O controls and instruments. A general purpose servomotor controller was developed along with a galvanometer impedance matching and protection circuit and a circuit breaker 'trip / read' circuit. Power to the system was provided by conventional low cost power supplies in the 8 to 18 V range unregulated current and 5 V, 15 V and -15 V regulated current for use in the computer and interface boards. The auxiliary circuits are described briefly at this point.

5.2.1 The Servo Controller.

The servo controller of Fig. 5.9 is a dual channel operational amplifier arrangement that requires DC input signals from a control output and a transducer mechanically linked to the controlled motor to close the feedback loop. The input-feedback error then operates the motor which moves the needle either of the RPM or the manifold pressure indicators. The circuit is designed for general purpose use and not restricted to the particular motor types of the above indicators. Two such circuit cards were developed and used to operate these indicator motors. The circuit provides for adjustment to the characteristics of the load in terms of damping, inertia, velocity and position. The required power need not be regulated since there is onboard regulation.

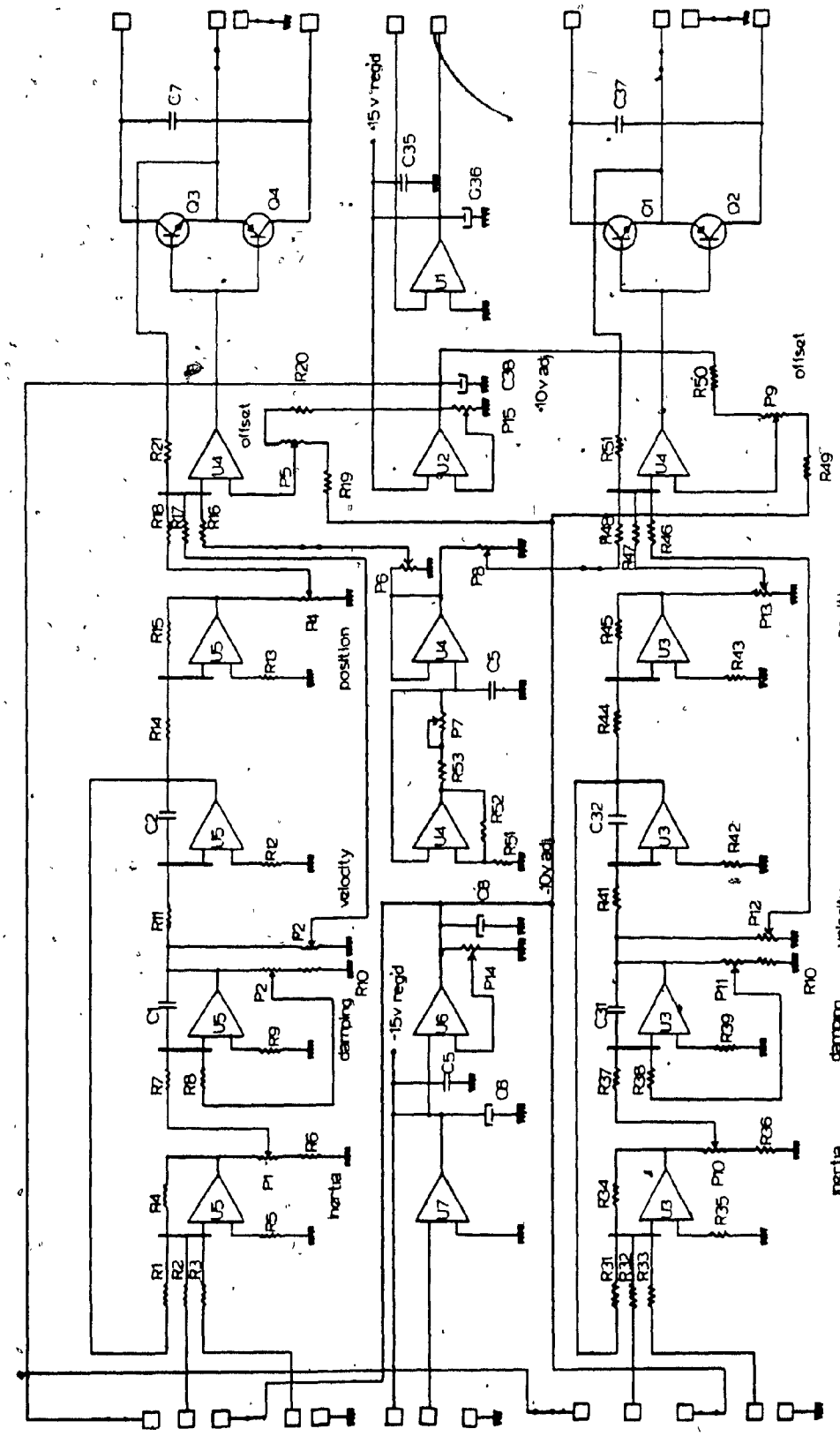


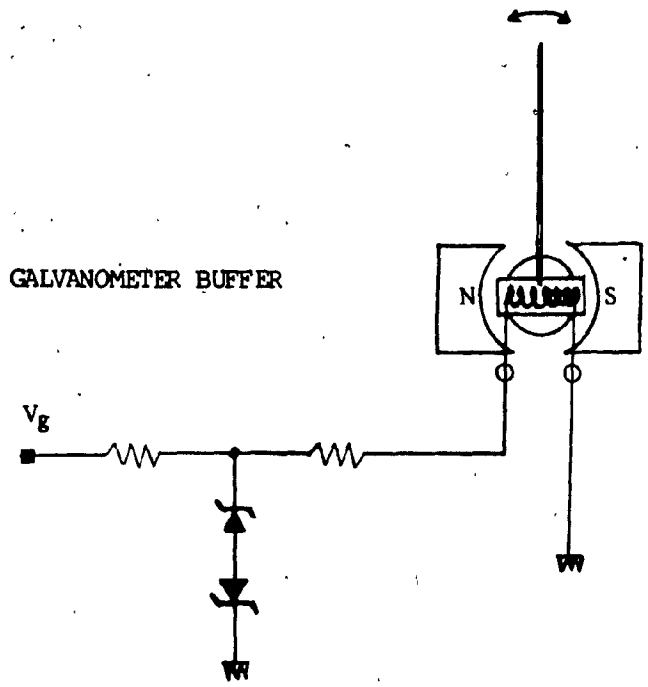
Fig. 5.9 The dual servo controller circuit.

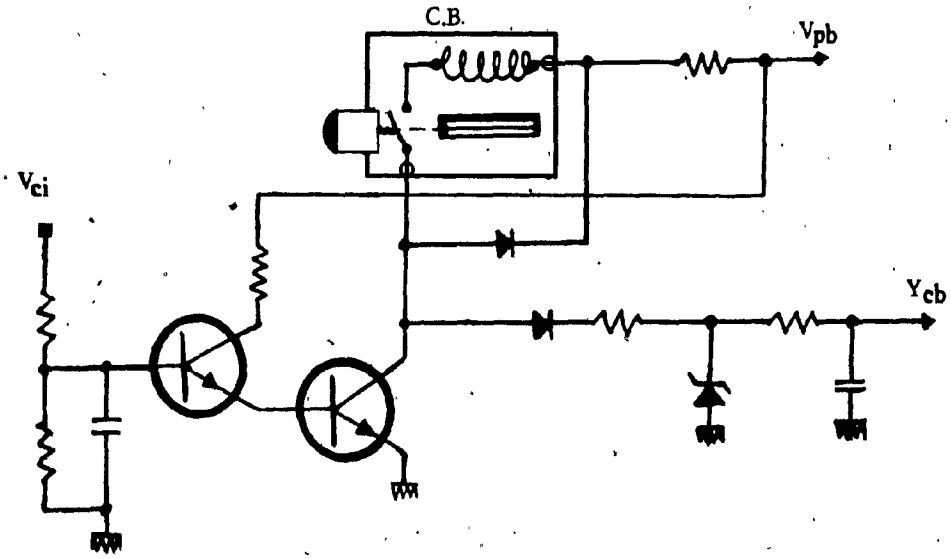
5.2.2 The Galvanometer Circuit.

The purpose of this simple circuit is to match the impedance of the Engine cluster and EGT indicators to that of the analogue outputs and provide protection by means of zener diodes for current surges and accidental overvoltage situations. (Fig. 5.10a).

5.2.3 The Circuit Breaker Circuit.

Since the C.Bs of the panel have to be tripped in case of overload and because they are also used as 'switches' by the operator, their status has to be known by the model software. They are both output and input devices. The circuit shown in Fig. 5.10b establishes that function by means of applying a low surge current and provides proper buffering to the input interface.





CIRCUIT BREAKER TRIP-AND-READ CIRCUIT

Fig. 5.10 The auxiliary circuits.

5.3 SOUND GENERATION

Sound is a very important sensory stimulus to the flight crew in an airliner, a military jet or a light aircraft. Familiarity with the use of sound cues is therefore important for proper training. The sound that the powerplant of a light twin aircraft produces is a 'composition' of the low pitch exhaust sound and the high pitch sound of the propeller 'cutting through' the airstream. These sounds vary with power application, propeller pitch and other factors; moreover a beat at harmonic frequencies occurs when both engines rotate at close angular velocities.

In order to reproduce realistic sound effects to match true engine behaviour some inflight measurements were made. These were analysed and the relations between engine control and measured sound frequencies were established, for example at idle the exhaust sound frequency was approximately 20 Hz. The next section examines the sound hardware that was developed for that purpose. Two sound channels are used for a twin engine aircraft powerplant.

5.3.1 The Sound Hardware.

The low frequency exhaust sound and the higher frequency propeller sound were reproduced by a dual circuit (Fig. 5.11) each consisting of two main components as follows:

5.3.1.1 The Voltage Controlled Oscillator.

This part of The circuit is formed by a dedicated timer (N555) generating a square wave. the frequency of that square wave is controlled by the analogue voltage input to the engine RPM indicator control circuitry. The free running frequency of the oscillator is preset in the centre of the required bandwidth. The output is varied by the RPM analogue voltage from 20 Hz at 600 rpm (idle) and two ignitions per revolution, to a maximum of 100 Hz at 3000 rpm. The square wave representing the low frequency exhaust sound is frtherther modulated by noise produced by a Gaussian noise generator. White noise is superimposed on the peak of the square wave at low amplitude and biased properly to reproduce the sound at the exhaust system.

5.3.1.2 The Propeller Sound generator.

During the space to mark transition of the square wave, an exponentially decaying noise signal from the same Gaussian noise generator is added to the signal to reproduce the sound of the propeller reacting with the airstream. This is accomplished by a simple R-C network connected in the final stage of the circuit. From

that point the generated sound signals of the engine and the propeller mixed in one composite signal are fed into a power amplifier and on to a speaker system of suitable power and frequency response. Three calibration controls for frequency offset, bandwidth and noise level were included in each channel; output curves of the circuit are shown in the next Chapter. The required audio power is 2 x 40 Watts for a maximum 90 dB level with a speaker system incorporating separate high and low frequency components ('tweeter', 'woofer' and a crossover network).

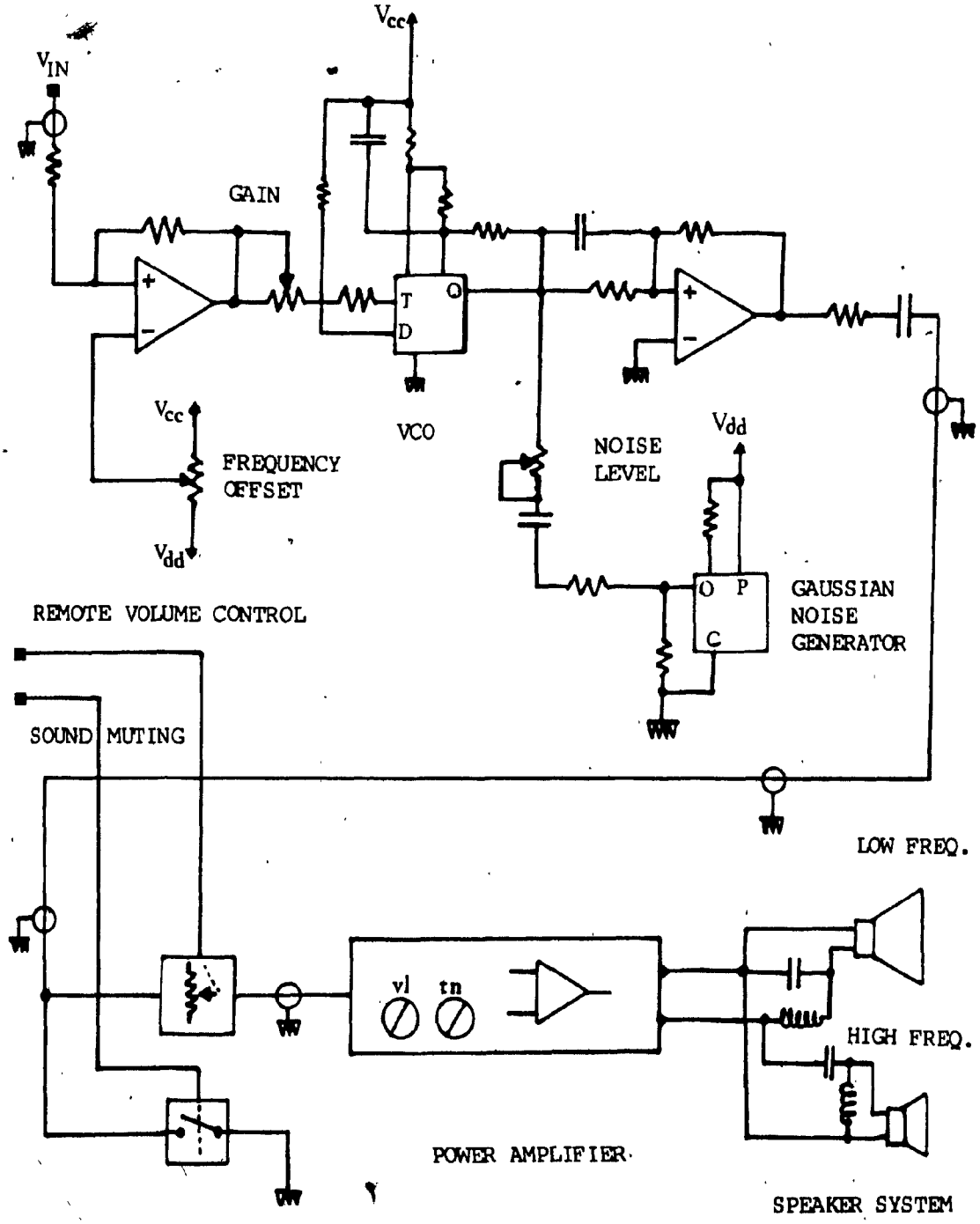


Fig. 5.11 The sound generation circuitry.

chapter 6

PERFORMANCE EVALUATION

This Chapter presents the overall performance of the Real Time computing system that was designed for this simulation and provides a general account of the behaviour of the model sofware in comparison to actual powerplant data. With respect to the computing system emphasis is placed on the analogue to digital and digital to analogue conversion schemes. The performance of the microcomputer system is seen from the real time computation viewpoint.

The performance of the simulation of the powerplant is examined by comparing with true aeroengine data and taking into account that the simulation is not particular to a specific engine model but rather to a general type of an aeroengine. Performance data were kept at an accuracy level sufficient for this type of simulation. The resultant output were obtained in number form and plotted. The resulting graphs appear in the next pages with the appropriate comments and explanations.

6.1 THE REAL TIME SYSTEM PERFORMANCE.

This section examines the performance level of the real time computer system used in the simulation. A separate look is taken at the behaviour of each of its two major components, that of the microcomputer and that of the signal conversion circuitry.

6.1.1 Real time computational performance.

Even though a high level language was used for modelling, the iSBC 8612A microcomputer was found capable of handling the memory, I/O and computational requirements of this simulation exceptionally well. The model and system software were compiled and linked in an INTEL development system. Then the executable images that were produced were loaded in the programme memory by means of the resident system monitor of the microcomputer via the serial communications port. Simple monitor commands such as GO AT 1000 were used to start execution at the origin of the image (e.g. 1000 Hex). The iteration rate was established at an optimum of 20 Hz with significant degradation below that limit and little improvement above it. (Fig. 6.1). Fidelity was measured by sensory comparison to that of the response of an actual aircraft powerplant in terms smoothness, likeness and validity. On the other hand, manufacturer's data were compared with indications and response times. An satisfactory level of fidelity then, was reached at 20 iterations per second. Below it, indicator needles were stepping and the response of the engines was timely incorrect. Above it, there was no noticeable improvement and

almost no spare computing power available.

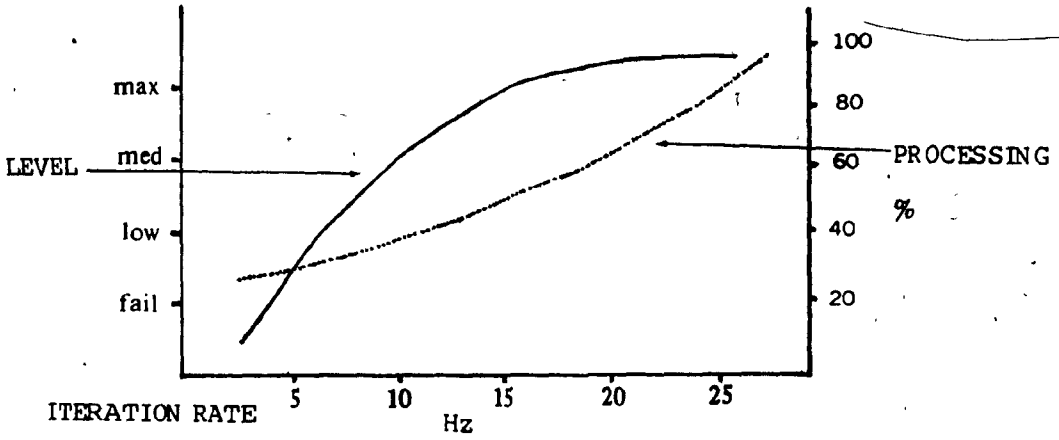


Fig. 6.1 The iteration table.

Lowering that figure to 5 iterations of model and I/O executions per second maintained an acceptable succession level of discrete and analogue data acquisition. The output however showed unacceptable stepping, especially in the servomotor indicators where the travel of the instrument needle is much longer than that of the galvanometers. Existing non linearity in some regions of indication and mechanical inertia worsened the effect. Since the measure of the level of fidelity of the simulation is at first empirical, made by sensory comparisons of visual and aural cues to real aircraft operation, the optimum iteration rate of the real time system was established as such. Due to the fact that the simulated system is a rather slow first order system, any iteration rates above the observed optimum did not produce any noticeable improvements in performance.

Floating point arithmetic was used throughout the models with most variables being of type Real. Four digit accuracy was sufficient. The microcomputer was equipped with an 8087 Numeric Data

Processor which limited the system clock frequency to 5 MHz but did not pose any computational speed constraints on the overall simulation. An average of five to ten percent of the computational capacity of the iSBC 8612A computer was available for background activity such as communications. In general, this microcomputer has a good reputation in industry and research for real time performance. That fact is reaffirmed in this study.

6.1.2 Analogue to digital conversion performance.

The task of converting signals from the ten analogue level transducers into digital format was assigned to a low cost multiplexed data acquisition module manufactured by Analogue Devices Corporation. The ten differential output signals required two hundred conversions per second at the iteration rate of the system. That was accomplished in a total of 200 sec by the chosen converter and so no interlacing of input and model execution was necessary. Of the 12 bit available resolution, only 8 bits were used since that was found sufficient and left the rest of the width of the data path of the Multibus free for other concurrent use. Good thermal stability was observed in the conversion and the signal to noise ratio was at a nominal 60 dB. RFI and EMI noise was not critical due to the shielding that was used. The maximum speed of this converter is of the order of 100 kHz. In case, however, that timing constraints were present, the channel selection and conversion times could be selected to overlap since the unit includes a sample and hold amplifier. Overall it can be stated that the analogue to digital conversion design performed better than

anticipated mostly due to the excellent characteristics of the conversion element.

6.1.3 The digital to analogue conversion performance.

Another integral unit was used for this type of signal conversion as seen in Chapter one. It comprised of three eight-channel digital to analogue converter circuit boards to accomodate the eighteen channels needed. Each of these units was directly connected with the Multibus due to available onboard decoding and data latching. A good level of performance were observed with no timing or other problems. Stability was measured at ±0.5%. Optoisolation of the analogue conversion sides was helpful in eliminating extra protection circuitry and noise propagation. Every I/O unit of the system was memory mapped on the Multibus. In general an excellent performance to cost ratio resulted by using these components instead of special purpose design.

The analogue output was distributed to the various analogue input instruments. In the case of the servomotor driven indicators a special inertia level adjustment on the servocontroller was useful in minimising stepping at low iteration rates. No capacitative filtering was included though at any stage of the circuit. The frequency response of the digital to analogue conversion system was observed to be acceptable and it was not required to be very high due to the dynamics of the instruments.

6.1.4 Discrete I/O system performance.

Discrete inputs to the system such as switches were filtered through Schmidt trigger buffers and included latching by software for small time intervals to avoid the need for debouncing circuitry. The buffers that were used were memory mapped. Discrete output was distributed to memory mapped latches where it was transmitted to the relevant devices. All discrete input and output operations were 8-bit and most were performed concurrently with analogue I/O. This scheme performed well. Finally, power consumption was kept at a 200 Watthour figure. However, forced air was used for thermal dissipation from the system.

To summarise, all the components of the Real Time computer system behaved within expected range and the total capacity of the system proved to be higher than the minimally required thus, leaving adequate space for possible expansion or modifications.

6.2 THE SIMULATION MODEL PERFORMANCE.

Having enough computing power available, the proper hardware design and a rather detailed model of the system to be simulated produced a realistic simulation at a low expenditure level. The fidelity of the results is described in this section. Data obtained from the computer in numeric form were plotted and are presented in comparison to relevant engine data. The most important quantity is certainly the response of the output to various inputs. In engineering studies three standard types of input are used, step, ramp and exponential. A combination of them is used in the samples to follow in an effort to approximate the original user input.

6.2.1 Simulation response to variant input.

Figure 6.2 shows the response of the simulation model to typical inputs. The variable parameter is the throttle (of one engine) at ordinary conditions, low altitude, approximately 2500 feet (QNH), 1 kPa atmospheric pressure, 20 °C temperature, and 50% relative humidity viz. standard atmosphere. Propeller load, Fuel/air ratio, aircraft forward velocity and other parameters were kept at typical values. The non linearities of the engine can be seen in the relationship between propeller pitch settings and aircraft velocities. The output is measured in terms of angular velocity.

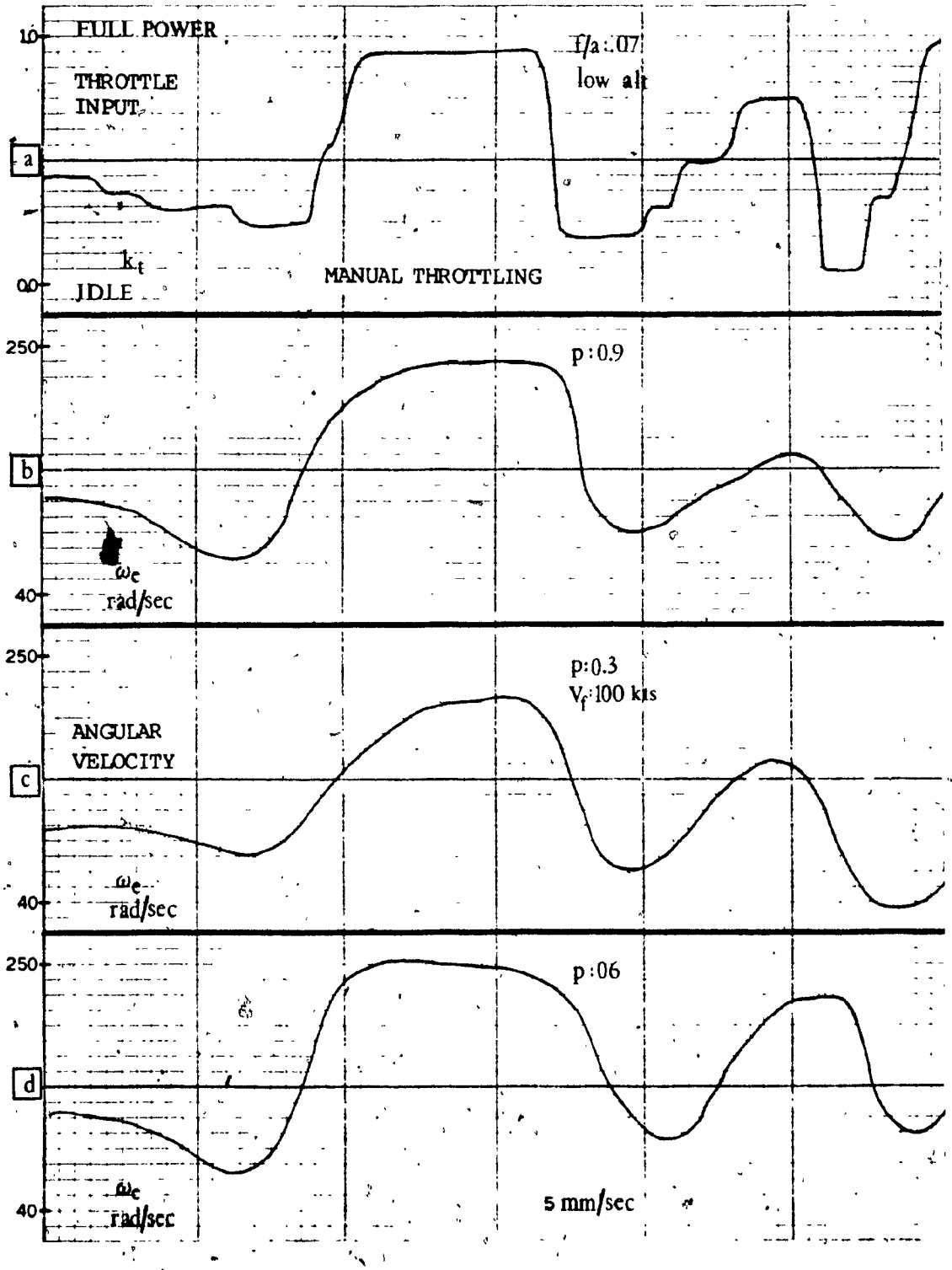


Fig. 6.2 Simulation response to manual input.

The response obtained is comparable to that of the preliminary analogue simulation model presented in Appendix B. The effects of high loading torque can be seen in Fig. 6.2c approximating take off rolling (middle segment). Fig. 6.2a represents the manual throttling applied to the model. Fig. 6.2b shows the simulated response at a cruise speed. Fig. 6.2d shows the model's response in an approach-like configuration. The simulation model performed as expected. Some problems could have arisen and they are discussed in the examination of the carburator model performance. Another example of the engine's throttling response can be seen in Fig. 6.3a at a different time scale.

The model output modulation by the Fuel/Air ratio setting is illustrated in Fig. 6.3b assuming airborne operation and the aircraft climbing. This effect is produced by function generation and produced the expected result. In general it can be stated that the behaviour of the simulation model reached the planned level of fidelity in terms of input throttling response. A very detailed engineering analysis of the output is not within the scope of the study since the model is of generic nature.

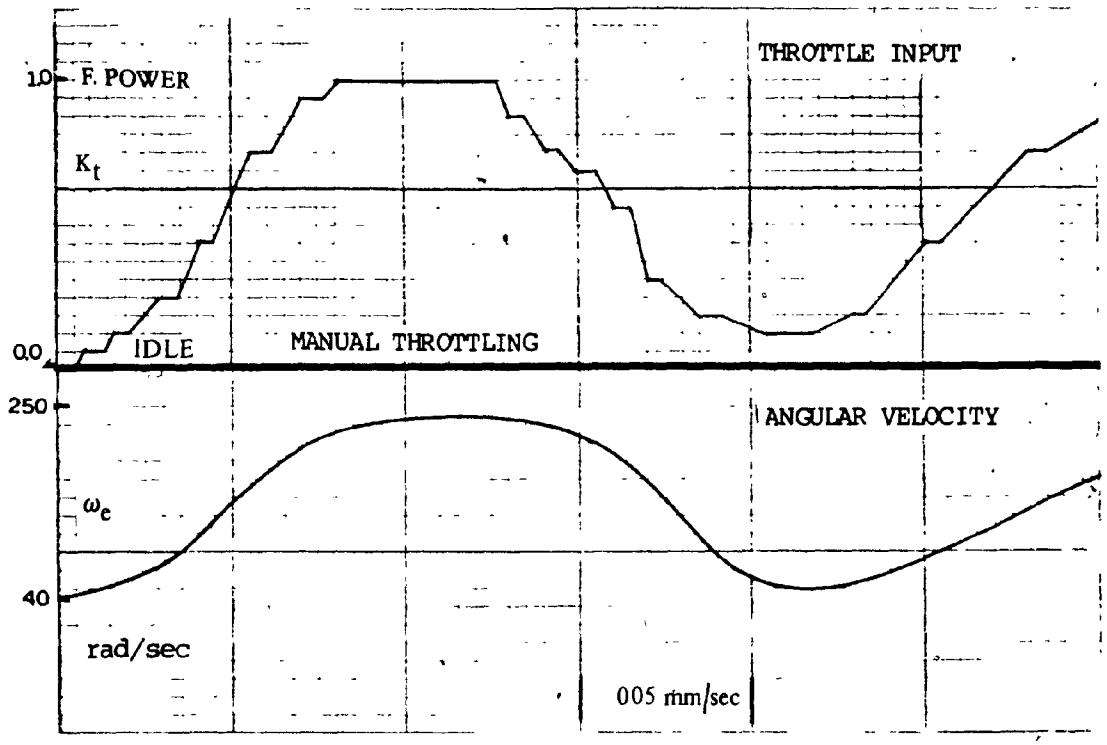


Fig. 6.3a Slower time scale response to manual input.

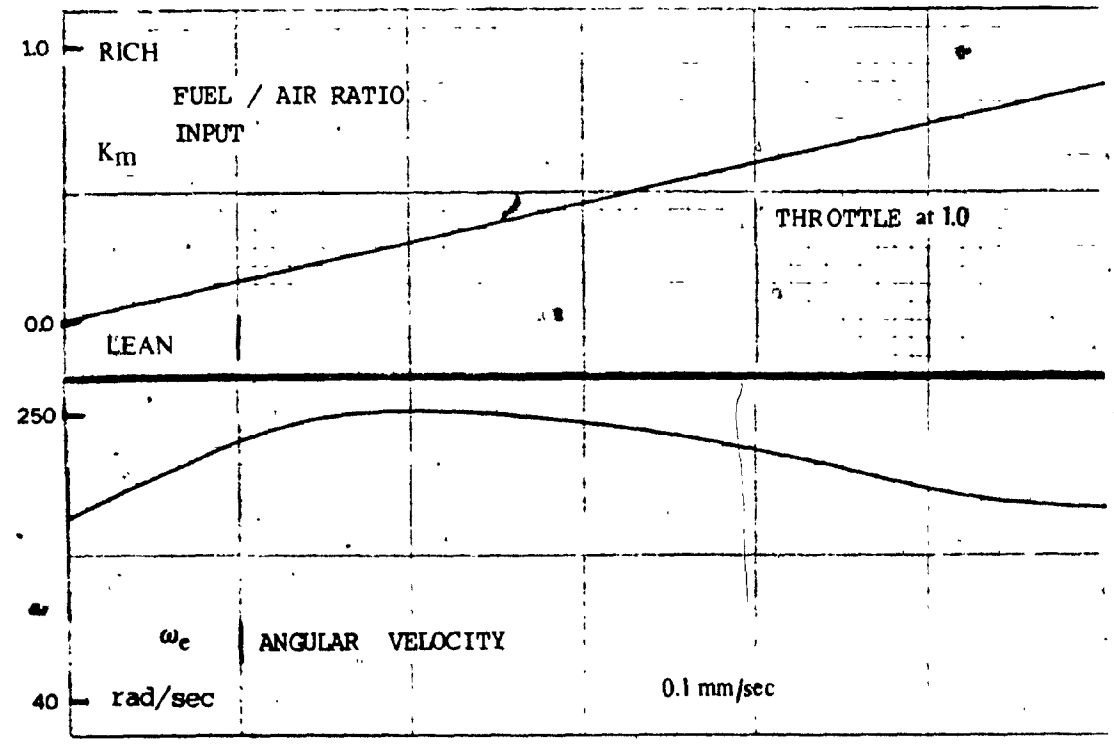
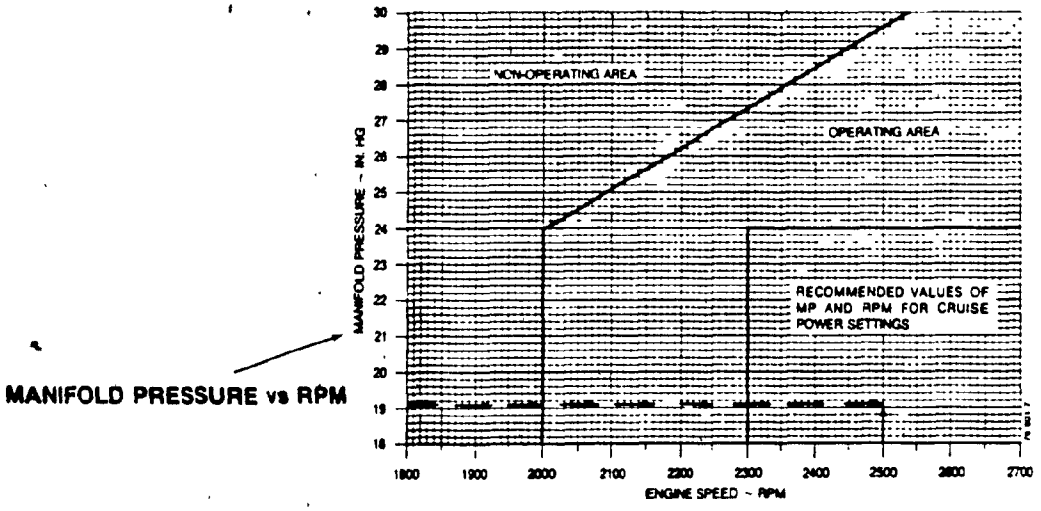


Fig. 6.3b Simulation response to mixture variation.

6.2.2 The carburator model performance.

Both in the aeroengine and the simulation model the carburator accepts the operator's input and translates it into mechanical regulation. As such it should be examined separately. This section presents the performance of the throttling process; fig. 6.4 is a plot of the pressure ratios at the carburator ends against angular velocity at different throttle settings. It can be seen that the pressure ratio varies with input throttle and the manifold pressure (suction) generated by the angular velocity of the engine disproportionately. The graph is a general description of the reproduced carburator behaviour assuming standard atmoshere conditions. A very similar plot was obtained by the analogue simulation model in Appendix B. The Manifold pressure / RPM chart of Fig. 6.4 shows the usual limits of carburator operation.



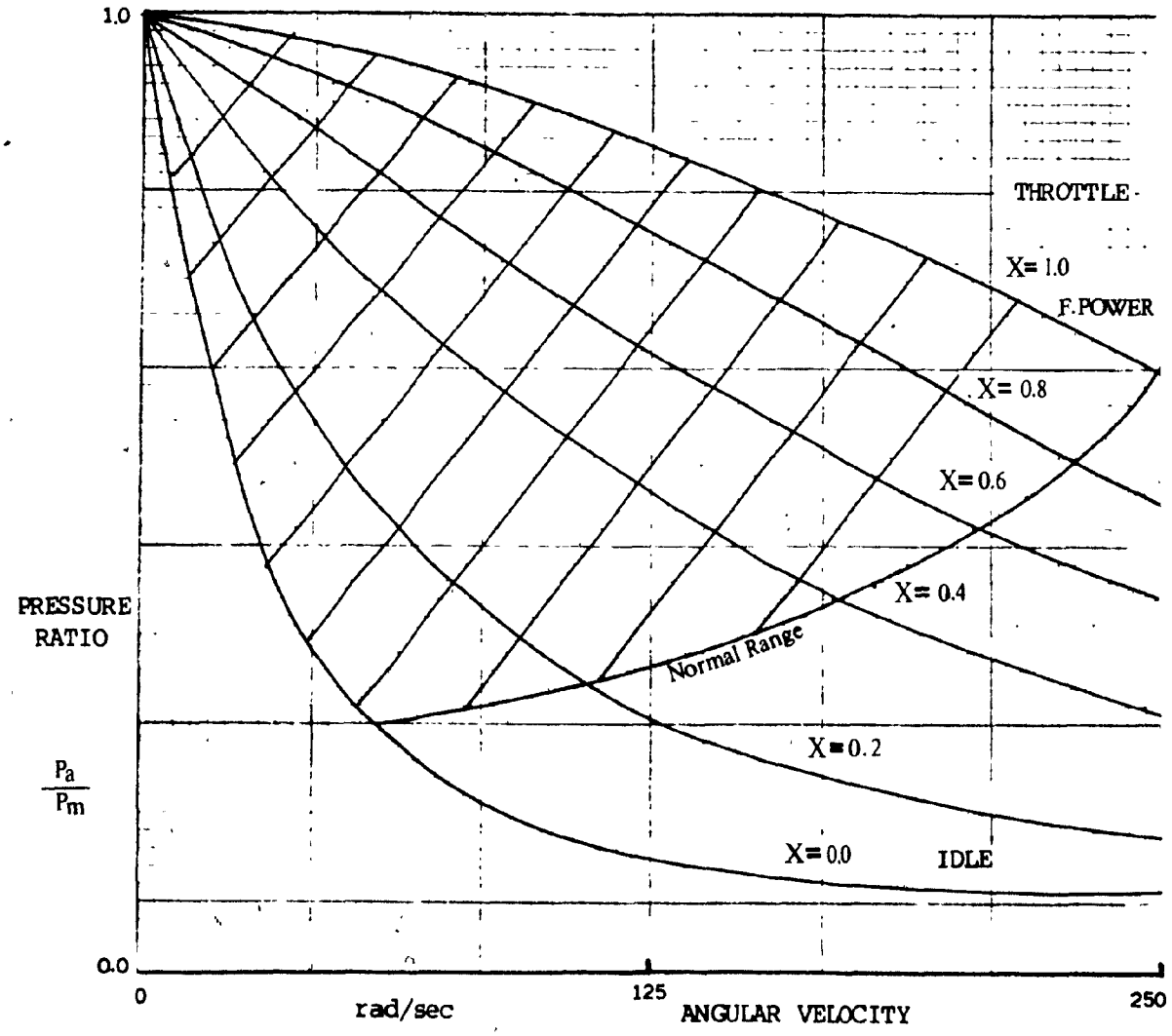


Fig. 6.4 The resultant throttling process.

6.2.3 Thermal simulation performance.

Monitoring of thermal parameters among others is a rather important part of the supervision and control of the powerplant of any aircraft. Illustrations of the simulated behaviour of the heat related system models appears in the next pages. Fig. 6.5a shows the simulated operation of the cowling mechanism with the cylinder head temperature dropping when air cooling is engaged. Different curves are shown for rich and lean mixtures. The operational conditions were assumed as those of a standard atmosphere and the aircraft in flight.

Fig. 6.5b illustrates the exhaust gas temperature variations in a transition from lean mixture to full rich assuming the previous standard conditions. It can be noticed that peak exhaust gas temperature occurs around the leaner mixture ratio setting for the operational conditions. This effect is reproduced by function generation. Finally fig. 6.6b shows the operation of the automatic oil temperature control when overheating occurs. The graph was produced under the previous operational conditions and constant angular velocity. Fig. 6.6a shows the table of the manufacturer's recommendations for various conditions. It was used as a general guide in testing the performance of the simulation. Appendix C contains some other recommendable operational guidelines for the O-360 aeroengine.

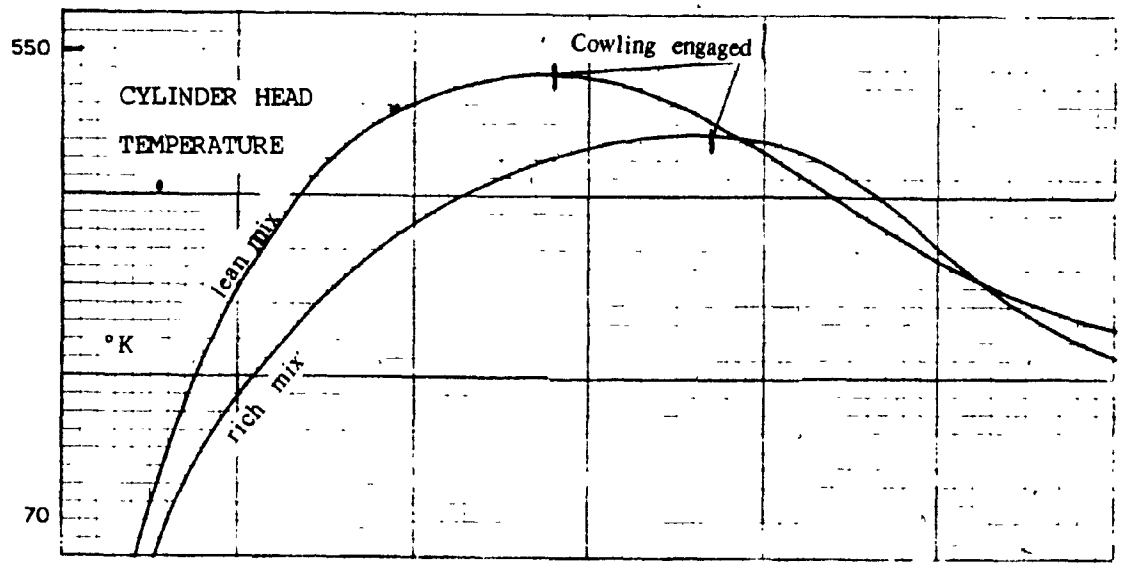


Fig. 6.5a Cylinder cooling simulated response.

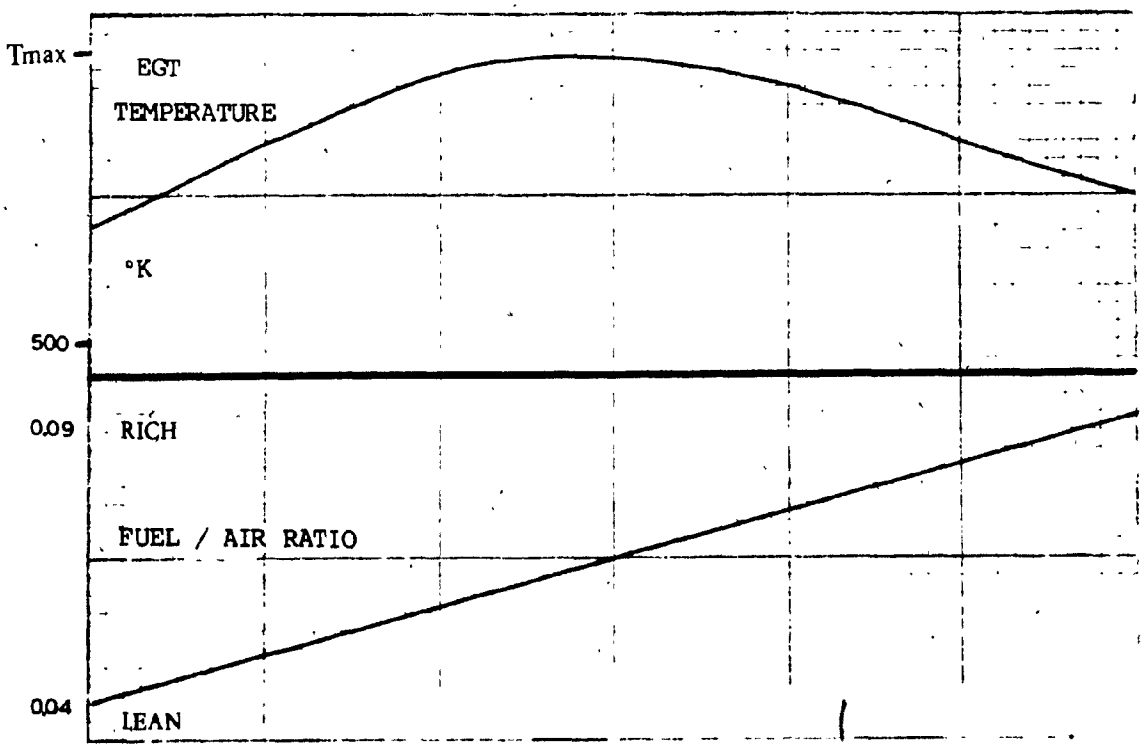


Fig. 6.5b Mixture variation and EGT response.

RECOMMENDED CRUISE POWER

PRESS ALT	ISA −20°C (−36°F) IOAT		MAN PRESS	FUEL FLOW ENGINE		IAS	TAS	STANDARD DAY (ISA) IOAT		MAN. PRESS	FUEL FLOW ENGINE		IAS	TAS
FEET	°C	°F	IN.HG	PPH	GPH	KTS	KTS	°C	°F	IN.HG	PPH	GPH	KTS	KTS
SL	-3	27	24 0	61	10 2	152	147	17	63	24 0	59	9 8	148	148
1000	-5	23	24 0	62	10 3	152	149	15	59	24 0	60	10 0	148	151
2000	-7	19	24 0	63	10 5	153	152	13	55	24 0	61	10 2	148	153
3000	-9	16	24 0	64	10 7	153	154	11	52	24.0	61	10.2	149	155
4000	-11	12	24.0	64	10 7	153	156	9	48	24 0	62	10 3	149	158
5000	-13	9	24 0	65	10 8	153	159	7	45	24 0	63	10 5	149	160
6000	-15	5	23.6	66	11 0	153	161	6	43	23 6	63	10 5	148	162
7000	-17	1	22 7	63	10 5	150	160	4	39	22.7	61	10.2	145	161
8000	-19	-2	21 9	61	10 2	146	159	2	36	21 9	59	9 8	142	160
9000	-21	-6	21.0	59	9 8	143	158	0	32	21.0	57	9.5	139	159
10.000	-23	-9	20.2	57	9 5	140	157	-3	27	20 2	55	9 2	136	158
11.000	-25	-13	19 4	55	9 2	137	156	-5	23	19 4	53	8 8	133	157
12.000	-27	-17	18 7	53	8 8	134	155	-7	19	18 7	51	8.5	130	156
13 000	-29	-20	18 0	51	8 5	131	153	-9	16	18 0	49	8 2	126	154
14.000	-31	-24	17.3	49	8.2	127	152	-11	12	17.3	47	7.8	123	152
15.000	-33	-27	16 6	47	7 8	124	150	-13	9	16 6	45	7 5	120	151
16.000	-35	-31	16 0	45	7.5	121	148	-15	5	16 0	43	7.2	116	148

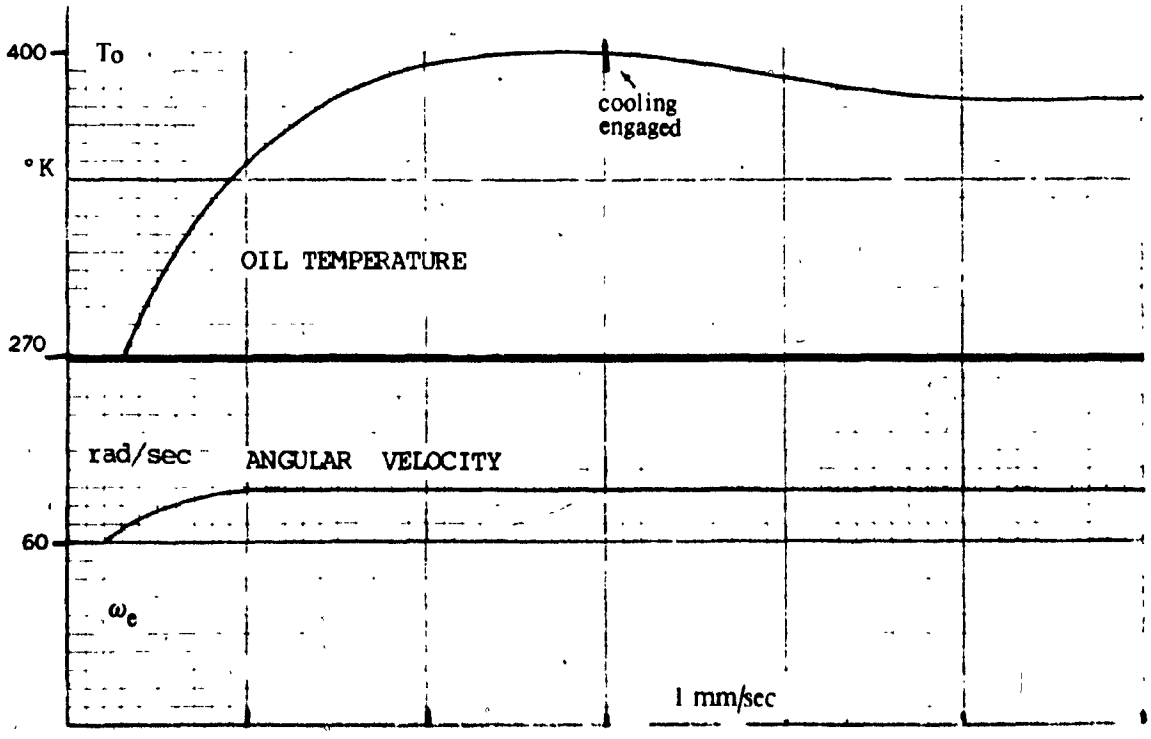


Fig. 6.6 Oil cooling mechanism simulated response.

6.2.5 Sound generation performance.

An examination of the performance of the sound generation hardware can be made. Figure 6.7 illustates the form of the generated sound wave which consists of the two components, that of the engine combustion and that of the propeller. The fidelity of the reproduction was adequate and included the presence of the harmonic beat between the two engines; the power amplification used gave another dimension to the effect and resulted in warnings of possible eviction notice issuance for the research facility. The sample of fig. 6.7 corresponds to idle engine operation.

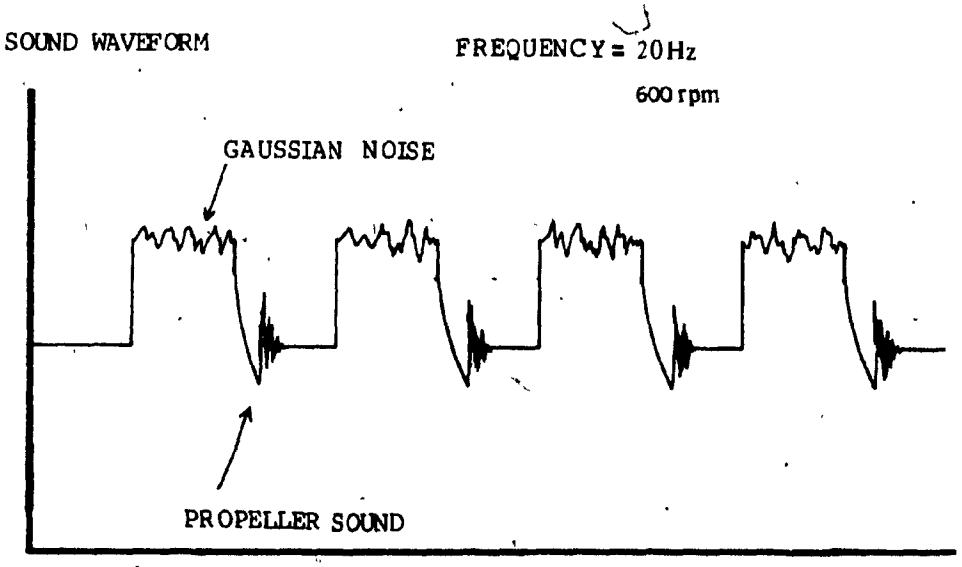


Fig. 6.7 Reproduced sound waveform.

CONCLUSION

It can be concluded from this study that a good quality simulation of an aeroengine powerplant system can be implemented on a sixteen bit microcomputer at low cost and with satisfactory results. An expanded version of this system is installed in the Light Twin Aircraft Simulator at Concordia University. Some modifications were necessary for adaptation and use under the iRMX86 operating system. Usage of microprocessor based computers is on the rise and the capabilities of such computing machines are improving steadily. For example iAPX86 systems can be replaced by the more powerful iAPX286 systems which include data integrity protection built into the hardware and faster execution speeds to the point that two or three such system cards in a multiprocessor configuration would be capable of handling the software of a total system such as a flight simulator.

One of the costly components of such project developments is the composition of machine independant models or control software. High level languages for microcomputer equipment are not currently fully transportable and lower level code is not at all compatible among different computers. Commonality, though, is demanded by users in

order to reduce development cost; Computer manufacturers will have to respond positively and adjust collectively to help the software industry create standard Ada, Fortran, or Pascal language versions.

Flight simulation is an excellent training method among airlines, Airforces and private pilot schools, now and in the years to come. High expenditures though for such training equipment can only be absorbed by military and commercial organisations. The Light Twin Aircraft Simulator research project at Concordia University though, has proven that the lower end of the flight simulation market can be served as well. Lower cost, microprocessor based computers, now make that possible. Integrating automated instruction with a flight simulator is another possibility well within reach at the present level of technology. Hardware cost though is small in comparison to software design and programming. And it is that part of project development where the mind can be left free to improvise; human creativity, ingenuity and artful skill can be unleashed to command the machine and create masterpieces of technology for just about everybody's benefit._

REFERENCES

2.1 FLUID MECHANICS.

D. Schaum, C. Van Der Merwe, New York, McGraw-Hill, 1976.

2.2 THERMODYNAMICS.

K. Wark, McGraw-Hill, New York, 1977.

2.3 PHYSICS, Vols. I,II,III. (Mechanics, Heat, Electricity).

A. Mazis, Estia Publ. Athens, 1972.

2.4 DIFFERENTIAL AND INTEGRAL CALCULUS.

C. Love, E. Rainville, McMillan Co. 1953.

2.5 THEORY OF FLIGHT.

R. Von Mises, Dover Publ. New York, 1959.

2.6 AIRCRAFT POWERPLANTS.

R. Bent, J. McKinley, Northrop Aeronautical Institute, McGraw-Hill, London, 1955.

B I B L I O G R A P H Y

1. CALCULUS AND ANALYTIC GEOMETRY.

   D. Riddle, Wadsworth Publ. Belmont, CA, 1976.

2. VECTOR CALCULUS.

   Marsden, Freeman Co. San Fransisco, 1976.

3. ALGEBRA AND TRIGONOMETRY.

   N. Tzioras, O.E.Δ.B, Athens, 1971.

4. CALCULUS AND SOLID GEOMETRY.

   E. Annikos, O.E.Δ.B, Athens, 1970.

5. STATISTICS.

   W. Collins, Mc Graw-Hill, New York, 1977.

6. DIGITAL INTERFACING WITH AN ANALOG WORLD.

   J. Carr, Tab Book Co. Blue Ridge, PA, 1978.

7. REAL TIME COMPUTING.

   D. Mellichamp, V. Nostrand Co. Melbourne, 1982.

8. INTERNAL COMBUSTION AEROENGINES. Vols. I, II, III, IV.

   Staff, Kronos School of Aeromechanics, Athens, 1970.

9. ASSEMBLY LANGUAGE PROGRAMMING.

   A. Leventhal, Osborne Co. Berkley, CA, 1978.

10. iAPX86,88 USER'S MANUAL.

    Staff, Intel Co, Santa Clara, CA, 1981.

11. iSBC APPLICATIONS HANDBOOK.

    Staff, Intel Co. Santa Clara, CA, 1982.

12. MICROSYSTEMS HANDBOOK. Vol I, II.
Staff, Intel Co. Santa Clara, Ca, 1984.

13. MATHEMATICAL MODEL OF AN INTERNAL COMBUSTION ENGINE.
J. Monk, Measurement and control, pp 93-100, 1970.

14. INTERNAL COMBUSTION ENGINES.
F. Obert, Int'l textbook Co. Scranton, PA, 1968.

15. ANALOG AND DIGITAL MODELLING OF A HYDRAULIC VEHICULAR TRANSMISSION.
J. Svoboda, Proc. I.M.E. Vol 193, pp 227-286.

16. HYDRAULIC HYBRID DRIVE EMPLOYING REGENERATIVE BRAKING.
J. Svoboda, Thesis, Ph.D. Montréal, Nov. 1975.

17. MECHANICAL PRIME MOVERS.
P. Bell, McMillan Press, 1971.

18. INTEL MULTIBUS SPECIFICATION.
Staff, Intel Co. Santa Clara, CA, 1982.

19. COMPUTER SYSTEMS ARCHITECTURE.
J. Baer, Computer Sc. Press, Rockville, MR, 1980.

20. 8086 ASSEMBLY PROGRAMMING MANUAL.
Staff, Intel Co. Santa Clara, CA, 1981.

21. MICROCOMPUTER INTERFACING.
B. Artwick, Prentice-Hall Co. Edgewood NJ, 1980.

22. DIGITAL LOGIC DESIGN.
F. Mowle, Addison Publ. Reading, MA, 1976.

23. MICROPROCESSOR APPLICATIONS HANDBOOK.
D. Stout, McGraw-Hill Co. New York, 1982.

24. 10th IMACS WORLD CONGRESS, PROCEEDINGS.
Modelling and Simulation, Vols. 1-4, Montréal, 1982.

25. COMMUNICATIONS OF THE ACM.
Articles in simulation, Years 1980-84

26. MICROPROCESSOR AND PERIPHERAL HANDBOOK.
Staff, Intel Co. Santa Clara, CA, 1983.

27. THE TTL DATA BOOK.
Staff, Texas Instruments, Dallas, TX, 1982.

28. Private consultations with Wolfgang Blach
and Alex Maroudis on engine simulation (LTAS).

29. Private consultations with CAE Electronics
staff on engine simulation, system configuration etc.

30. Private consultations with faculty of the Computer
Science and Mechanical Engineering departments.

31. ENGINE TEST CELL PROCEDURES (R.I.C.E)
Staff, Hellenic Aerospace, Tanagra, Greece, 1978.

32. RECIPROCATING ENGINES.
Staff, Pratt and Whitney, Montréal, 1955.

33. BEECH DUCHESS PILOT'S MANUAL.
Staff, Beech Acft. Wichita, KS, 1979.

34. BEECH DUCHESS MECHANICAL SPECIFICATIONS MANUAL.
Staff, Beech Acft. Wichita, KS, 1979.

35. BEECH DUCHESS OPERATIONAL PROCEDURES.
Staff, Beech Acft. Wichita, KS, 1979.

36. INTEL 8612A SBC MANUAL.

Staff, Intel Co. Santa Clara, CA, 1980.

37. DATA ACQUISITION HANDBOOK.

Staff, Analog Devices Co, Norwood, MA, 1982.

38. IC-OP AMP HANDBOOK.

W. Jung, Sams Pub. Indianapolis, IA, 1983.

39. TELECOMMUNICATIONS AND COMPUTERS.

F. Martin, Prentice Hall, New York, 1980.

40. FLYING CONDITIONS.

D. Pagen, H.D.Press, Toronto, 1982.

41. INSTRUMENT FLYING.

R. Taylor, McMillan Pr. New York, 1980.

VOLUME II

APPENDIX A

THE PROGRAMMING MODEL.

This appendix presents the programming model of the simulation and system software in pseudocode notation. Variable names are coded in formats that reflect their fuction.

A.1 THE SYSTEM MODELS.

This section presents the various modules that represent the Ancillary and Propulsion systems of the Powerplant; the order of presentation is that of the less variable dependant models to the higher ones.
Function generation is not included in this Appendix.
Absence of units in the variables lists signifies logical values or dimensionless quantities.

A.1.1 The Fuel System Module.

Both fuel systems of the Powerplant are programmed;
they are referenced as Left (L) and Right (R).
Mechanical and electrical variables are computed in the engine and electrical Modules;
the same applies to malfunctions.

FUEL SYSTEM

List of System Variables.

Variable	Type	Description	Unit
FUAVLT	Bool	Fuel available in Left Tank	
FUAVRT	Bool	Fuel available in Right tank.	
FUAVLE	Bool	Fuel available to Left engine.	
FUAVRE	Bool	Fuel available to Right engine.	
CFSLVL	Int	Fuel selector valve Left system.	
CFSLVR	Int	Fuel selector valve Right system.	
FUFLPL	Real	Fuel flow at priming, Left eng.	kg/sec
FUFLPR	Real	Fuel flow at priming, Right eng.	kg/sec
FUPRXL	Real	Fuel pressure, Aux. pump Left	kg/m
FUPRXR	Real	Fuel pressure, Aux. pump Right	kg/m
FUPRML	Real	Fuel pressure, Mech. pump Left	kg/m
FUPRMR	Real	Fuel pressure, Mech. pump Right	kg/m
FUPRIL	Real	Fuel pressure Indicated Left	kg/m
FUPRIR	Real	Fuel pressure Indicated Right	kg/m
FUPRAL	Real	Fuel pressure displayed Left	kg/m
FUPRAR	Real	Fuel pressure displayed, Right	kg/m
FUFLLE	Real	Fuel flow to Left engine	kg/sec
FUFLRE	Real	Fuel flow to Right engine	kg/sec
FUQACL	Real	Fuel consumption Left engine	kg/sec
FUQACR	Real	Fuel consumption Right engine	kg/sec
FUSPCL	Real	Specific fuel consumption Left	kg/kN sec
FUSPCR	Real	Specific fuel consumption Right	kg/kN sec
FUSPCS	Real	System average sfc.	kg/kN sec

Appendix A — The Programming model

List of Malfunction Variables.

Variable	Type	Description	Units
XFINLE	Bool	Fuel flow discontinuance Left system	
XFINRE	Bool	Fuel flow discontinuance Right system	
XFLKLE	Bool	Fuel leak (s) present in Left system	
XFLKLE	Bool	Fuel leak (s) present in Right system	
XFLRLE	Real	Fuel leakage rate Left system	kg/sec
XFLRRE	Real	Fuel leakage rate Right system	kg/sec
XFPAXL	Bool	Auxiliary pump out of order Left	
XFPAXR	Bool	Auxiliary pump out of order Right	
XFQSNL	Bool	Quantity sensor defective Left	
XFQSNR	Bool	Quantity sensor defective Right	
XFSLVL	Bool	Selector valve inoperative Left	
XFSLVR	Bool	Selector valve inoperative Right	
XFCKVL	Bool	Check valve internal leak Left	
XFCKVR	Bool	Check valve internal leak Right	
XFQIVL	Bool	Quantity Indicator defective Left	
XFQIVR	Bool	Quantity Indicator defective Right	
XFPRIL	Bool	Pressure Indicator defective Left	
XFPRIR	Bool	Pressure Indicator defective Right	
XFPRVL	Real	False Pressure Indication Left	kg/m
XFPRVR	Real	False Pressure Indication Right	kg/m

List of System Constants

Constant	Type	Description	Units
FUQAMN	Real	Minimum usable fuel Quantity	litre
FUPREL	Real	Normal Aux pump pressure Left	kg/m
FUPRER	Real	Normal Aux pump pressure Right	kg/m
FUFCEL	Real	Voltage Gain constant	
FUFCMC	Real	Angular Velocity Gain constant	
FUPDFS	Real	Addition Offset constant	
FUPRFL	Real	Pressure friction losses	kg/m
FUFLCS	Real	Fuel flow injection constant	
FUSPGR	Real	Fuel specific gravity	kg/m
FUHTGS	Real	Thermal equivalent constant	Kcal/Nsec

```
  procedure FUELSYSTEM;
{ All variables global. Procedure computes
 fuel flow, consumption, pressure and sfc.}
 begin
 {input is read by I/O control.}
 {fuel available in tanks.}
  if FUQALT - FUQAMN > 0 then
    FUAVLT := true;
 if FUQART - FUQAMN > 0 then
    FUAVRT := true;
 {fuel available to engines.}
 if FUAVLT or FUAVRT then
   begin
    FUAVLE := (FUAVLT and CFSLVL = 'on') or
              (FUAVLT and CFSLVL = 'cross') or
              (FUAVRT and CFSLVR = 'cross') or  not XFINLE;
    FUAVRE := (FUAVRT and CFSLVR = 'on') or
              (FUAVRT and CFSLVR = 'cross') or
              (FUAVLT and CFSLVL = 'cross' ) or  not XFINRE;
 end
 {calculate fuel pressure and fuel flow Left system}
 if FUAVLE and IGONLE then
   begin
   if UFPAXL and not XFPAXL then
      FUPRXL := FUPREL * UVOBSL * FUFCEL
   if not XFPMCL and not XFCKVL then
      begin
       FUPRML := FUPRMC * EANVLL * FUFCMC;
       FUPRAL := FUPRML + FUPRXL * FUPDFS - FUPRFL;
      end
    else
      FUPRAL := 0;
    if not XFPRIL then
      FUPRIL := FUPRAL;
     else
      FUPRIL := XFPRVL;
    FUFLLE := FUFLCS * sqrt (FUSPGR (EPRAMB - EPRVNL));
     if IGPRLE then
       FUFLLE := FUFFLE + FUFLPL;
end  { fuel pressure and flow  Left }
 {calculate fuel pressure and fuel flow Right system}
 if FUAVRE and IGONRE then
   begin
   if UFPAXR and not XFPAXR then
      FUPRXR := FUPREL * UVOBSR * FUFCEL
   if not XFPMCR and not XFCKVR then
      begin
       FUPRMR := FUPRMC * EANVLR * FUFCMC;
       FUPRAR := FUPRMR + FUPRXR * FUPDFS - FUPRFL;
      end
    else
      FUPRAR := 0;
```

```
    if not XFPRIR then
       FUPRIR := FUPRAR;
     else
       FUPRIR := XFPRVR;
    FUFLRE := FUFRCS * sqrt (FUSPGR (EPRAMB - EPRVNR));
     if IGPRRE then
        FUFLRE := FUFFRE + FUFRPR;
  end    {Fuel pressure and flow Right system}
 { calculate fuel quantity left tank }
 if CFSLVL = 'on' then
    FUQALT := FUQALT - FUQACL;
  else
    if CFSLVL = 'cross' then
       FUQALT := FUQALT - FUQACL - FUQACR;
  if not XFQSNL then
     FUQAIL := FUQALT;
  else
     FUQAIL := XFQIVL;
 { calculate fuel quantity right tank }
 if CFSLVL = 'on' then
   FUQART := FUQART - FUQACR;
  else
    if CFSRVR = 'cross' then
    FUQALR := FUQART - FUQACR - FUQACL;
  if not XFQSNR then
    FUQAIR := FUQART;
 else
    FUQAIR := XFQIVR;

{calculate fuel consumption left system}
if not XFINLE then
   FUQACL := FUQACL + (FUFLLE + XFLRLE) * DTIME;
 else
   FUQACL := 0;
{calculate fuel consumption left system}
if not XFINRE then
   FUQACR := FUQACR + (FUFLRE + XFLRRE) * DTIME;
 else
   FUQACR := 0;
{calculate specific fuel consumption}
 FUSPCL := FUHTGS / HHTGAS * EFDFAL * EPERDG;
 FUSPCR := FUHTGS / HHTGAS * EFDFAR * EPERDG;
 FUSPCS := (FUSPCL + FUSPCR) / 2;
end  { procedure FUELSYSTEM }
```

A.1.2 The Electrical System Module.

The electrical system is simulated in this section, both of the left power bus and the right, according to the model derived in Chapter two. The key configuration of the system is the Bus Isolation circuit breakers from the battery bus and the Bus Tie circuit breaker which connects in parallel both power busses; in this mode with the parallel option of the voltage regulators both alternators share the load equally. Of interest are also the battery current limiters to each bus; these allow approximately 40 Amperes maximum current to pass to both busses. That allows the same amount of current to cross from one bus to the other when both Isolation CBs are closed. The valid permutations are represented by a decision tree of degree four, reduced from sixteen possible combinations to only eight significant ones at depth four as depicted in Chapter two. The maximum power rating that the system can deliver is approximately 3 kW.

ELECTRICAL SYSTEM

{ Both systems simulated in one of eight modes }

List of System Variables.

Variable	Type	Description	Unit
CUSXPL	Bool	Switch Auxiliary fuel pump Left	
CUSXPR	Bool	Switch Auxiliary fuel pump Right	
CUSSTL	Bool	Switch Starter Left	
CUSSTR	Bool	Switch Starter Right	
CUSPRL	Bool	Switch Prime Left	
CUSPRR	Bool	Switch Prime Right	
CUSLIL	Bool	Switch Ignition Left subsystem Left	
CUSRIL	Bool	Switch Ignition Right subsystem Left	
CUSLIR	Bool	Switch Ignition Left subsystem Right	
CUSRIR	Bool	Switch Ignition Right subsystem Right	
CUSBTR	Bool	Switch Battery relay	
CUSAFL	Bool	Switch Alternator Field Left	
CUSAFR	Bool	Switch Alternator Field Right	
CBALFL	Bool	Circuit Breaker Alternator Field Left	
CBALFR	Bool	Circuit Breaker Alternator Field Right	
CBSTPL	Bool	C. B. Start, Prime Control Left	
CBSTPR	Bool	C. B. Start, Prime Control Right	
CBECLE	Bool	C. B. Eng. Instr. cluster Left	
CBECRE	Bool	C. B. Eng. Instr. cluster Right	
CBISBL	Bool	C. B. Bus Isolation from battery Left	
CBISBR	Bool	C. B. Bus Isolation from battery Right	
CBBSTI	Bool	C. B. Bus Tie together	
CBAXPL	Bool	C. B. Auxiliary Pump Left	
CBAXPR	Bool	C. B. Auxiliary Pump Right	
UINSTL	Real	Starter Current Intensity Left	Amp
UINSTR	Real	Starter Current Intensity Right	Amp
UPAVBT	Bool	Battery power available	
UPAVAL	Bool	Alternator power available Left	
UPAVAR	Bool	Alternator power available Right	
UEAVML	Bool	Minimum Ang. Vel. available Left	
UEAVMR	Bool	Minimum Ang. Vel. available Right	
UPALWL	Real	Power Available (Alt.) Left	Watt
UPALWR	Real	Power Available (Alt.) Right	Watt
UBCHAV	Real	Battery charge	Amp/hr
UPDBSL	Real	Power Demand Bus Left	Watt
UPDBSR	Real	Power Demand Bus Right	Watt
UPDBST	Real	Power Demand System	Watt
URSBSL	Real	Load Resistance Bus left	Ohm
URSBSR	Real	Load Resistance Bus Right	Ohm
URSBST	Real	Load Resistance System	Ohm
UVOBSL	Real	Bus Voltage (rms) Left	Volt
UVOBSR	Real	Bus Voltage (rms) Right	Volt
UVOBST	Real	Bus Voltage System	Volt
UINBSL	Real	Bus Current Intensity Left	Amp
UINBSR	Real	Bus Current Intensity Right	Amp

UINBST	Real	Bus Current Intensity System	Amp
UVOBAT	Real	Battery Bus Voltage	Volt
UINBAT	Real	Battery Bus Current Intensity	Amp
ULDMTL	Real	Loadmeter indication Left	(%)
ULDMTR	Real	Loadmeter indication Right	(%)
UOVRVL	Bool	Indicator/Status Overvoltage Left	
UOVRVL	Bool	Indicator/Status Overvoltage Right	
UNDRVL	Bool	Indicator/Status Undervoltage Left	
UNDRVR	Bool	Indicator/Status Undervoltage Right	

List of Malfunction Variables.

XUBTRD	Bool	Battery out of order
XUBTRC	Bool	Battery connection intermittent
XUBTRL	Real	Battery charge (%)
XUALFL	Bool	Alternator field or switch out
XUALFR	Bool	Alternator field or switch out
XUALOL	Bool	Alternator or regulator or overvoltage control or fuse open. Left
XUALOR	Bool	Alternator or regulator or overvoltage control or fuse open. Right
XUVRGL	Bool	Shorted voltage regulator Left
XUVRGR	Bool	Shorted voltage regulator Right
XUOVRL	Bool	Overvoltage relay cutoff Left
XUOVRR	Bool	Overvoltage relay cutoff Right
XULDML	Bool	Loadmeter out of order Left
XULDMR	Bool	Loadmeter out of order Right
XUAXPL	Bool	Aux. fuel pump defective Left
XUAXPR	Bool	Aux. fuel pump defective Right
XUSTRL	Bool	Starter defective Left
XUSTRR	Bool	Starter defective Right

List of System Constants

Constant	Type	Description	Unit
UINLMB	Real	Current limiter Battery	Amp
UVBTRY	Real	Nominal Battery Voltage	Volt
UIBTRY	Real	Capacity, Battery	Amp/hr
UCHRTE	Real	Battery, charge rate	Amp/hr
UMNAVL	Real	Minimum Alt. angular velocity	rad/sec
UEQALM	Real	Maximum alt. loading Torque	Nm
UPAMAX	Real	Maximum Alt. continuous power	Watt
URZSTA	Real	Starter motor impedance	Ohm
UEAVKL	Real	Angular velocity constant	
UVOMAX	Real	Maximum Alt./Bus Voltage	Volt
UREGKN	Real	Regulation constant	
ULDPKN	Real	Load percent constant	
CBBSTI	Int	Bus Tie cutoff	Amp
CBISOL	Int	Bus Isolation cutoff	Amp
CBSTRT	Int	Start control cutoff	Amp
CBGAGE	Int	Engine instruments cutoff	Amp
CBALFD	Int	Alternator Field cutoff	Amp
CBAXPM	Int	Auxiliary Pump cutoff	Amp

```
procedure ELECSYSTEM;
{ procedure to simulate the electrical systems;
both,left and right systems are computed.}
 begin
{ Available battery power }
 UPAVBT := not XUBTRD and CUSBTR and (UBCHAV > 0);
  if UPAVBT then
     UVOBTR := UVBTRY;
     UINBTR := XUBTRL * UIBTRY;
     UCHBTR := UINBTR * UCHRTE;
  else
     UVBTRY := 0;
 { power available from left alternator }
  if EANVLL > UMNAVL then
     UEAVML := true;
  else
     UEAVML := false;
  UPAVAL := CBALFL and CUSAFL and not XUALOL and
            not XUALFL and not XUOVRL and UEAVML;
  if UPAVAL then
     UPALWL := UPAMAX;
  else
     begin
      UPALWL := UPAMAX * UEAVKL * EANVLL;
       if UPALWL > UPAMAX then
         OUVRVL := true;
     end
  else
    UPALWL := 0;
 UINALL := UPALWL / URSBSL * URSBSL;
   if XUVRGL then
     begin
     UVOALL := URSBSL * UINALL;
      if UVOALL > UVOMAX then
         UOVRVL := true;
    end
    else
       UVOALL := URSBSL * UINALL * UREGKN;
    if UVOALL < UVBTRY then
        UNDRVL := true;
{ loading torque left }
EQALTL := UINALL * UVOALL * UEQALM / UPAMAX;
  { power available from right alternator }
   if EANVLR = UMNAVL then
     UEAVMR := true;
   ( else
     UEAVMR := false; )

   UPAVAR := CBALFR and CUSAFR and not XUALOR and
             not XUALFR and not XUAVRR and UEAVMR;
   if UPAVAR then
     UPALWR := UPAMAX;
```

```
    else
       begin
        UPALWR := UPAMAX * UEAVKR * EANVLR;
         if UPALWR > UPAMAX then
           UOVRVR := true;
       end
    else
      UPALWR := 0;
    UINALR := UPALWR / URSBSR * URSBSR;
      if XUVRGR then
        begin
       UVOALR := URSBSR * UINALR;
        if UVOALR > UVOMAX then
           UOVRVR := true;
       end
       else
          UVOALR := URSBSR * UINALR * UREGKN;
       if UVOALR < UVBTRY then
          UNDRVR := true;
{ loading torque right }
EQALTR := UINALR * UVOALR * UEQALM / UPAMAX;
 end   { procedure }
```

```
procedure  BUSSTIE;

{ procedure to compute the effect of the
power busses tied together }

begin
 UVOBST := (UVOALL + UVOALR) / 2;  {rms}
  URSBST := URSBSL * URSBSR / ( URSBSL + URSBSR);
   UINBST := UVOBST / URSBST ;  {rms}
     UPDBST := UINBST * UINBST * URSBST;
end
```

```
procedure BATTERY;

{ procedure to calculate the effects of the battery's
 presence in the circuitry.}
 begin
 case
     1. CBISBL and CBISBR and ( UVOBST < UVOBTR );
           UCHBTR := UCHBTR - UINBST * DTIME;

     2. CBISBL and not CBISBR and ( UVOBSL < UVOBTR );
           UCHBTR := UCHBTR - UINBSR * DTIME;

     3. CBISBR and not CBISBL and ( UVOBSR < UVOBTR );
           UCHBTR := UCHBTR + UCHRTE * DTIME;

     else  UCHBTR := UCHBTR + UCHRTE * DTIME;

          if ( UINBSL > UINLMB ) and CBISBL then
              UINBSL := UINLMB;
          if ( UINBSR > UINLMB ) and CBISBR then
              UINBSR := UINLMB;

 end
```

```
 procedure ELECONFA;

{ procedure that establishes the behaviour of the
electrical system with the BusTie and the Battery
connected.}

begin

  BUSSTIE;
  BATTERY;

   ULDMTL := ULDMTR := UINBST * ULDPKN;

end


 procedure ELECONFB;

{ procedure as the A configuration but with
the battery disconnected.}

begin

   BUSSTIE;

    ULDMTL := ULDMTR := UINBST * ULDPKN;

end

 procedure  ELECONFC;

{ procedure to simulate the bus behaviour
in separate terms i.e. no BusTie. The battery
is connected through current limiters. }

begin

   BATTERY;
     ULDMTL := UINBSL * ULDPKN;
     ULDMTR := UINBSR * ULDPKN;

       if ( ULDMTL < UINLMB ) or ( ULDMTR < UINLMB ) then
         ULDMTL := ( ULDMTL + ULDMTR ) / 2;
         ULDMTR := ULDMTL;
end


  procedure ELECONFD;

{ procedure to simulate the system with the battery
connected }
```

```
begin

  BATTERY;
   ULDMTL := UINBSL * ULDPKN;
   ULDMTR := UINBSR * ULDPKN;

end  { procedure }

  procedure CIRCBRKS;

{ subprogramme to 'pop' circuit breakers
  on overload conditions; the same devices
  are also read in input mode since they
  may be reset by the operator }

   begin

     { alternator stator left }

     if XUVRGL and not XUALFL and CUSAFL and
       ( UEAVML or ( UPAVBT and CBISBL )) and not
         XUOVRL and CUSLIL and CBALFL and
       ( UVOBSR > UCBMIN ) then

        BCBAFL := true;

     { auxiliary pump left }

    if XUAXCL and ( UVOBSR > UCBMIN ) and
       CUSXPL and CBAXPL then

      BCBAXL := true;

     { overvoltage left }

    if XUOVRL and UEAVML and
      ( UVOBSL > UVOMAX ) then

       BUOVVL := true;

       { undervoltage left }

    if ( UVOBSL < UVBTRY ) and IGONLE then

       BUNDVL := true;

       { start control }

    if CBSTPL and IGONLE and XUSTRL then

       BCBSTL := true;

       { engine cluster left }
```

```
if CBECLE and XUICSL and IGONLE then

   BCBINL := true;

   { isolation left }

if CBISBL and UMNAVL and
  ( UINBSL > CBISOL ) then

  BCBISL := true;


{ alternator stator right }

 if XUVRGR and not XUALFR and CUSAFR and
   ( UEAVMR or ( UPAVBT and CBISBR )) and not
     XUOVRR and CUSLIR and CBALFR and
   ( UVOBSR > UCBMIN ) then

    BCBAFR := true;

 { auxiliary pump right }

if XUAXCR and ( UVOBSR > UCBMIN ) and
   CUSXPR and CBAXPR then

   BCBAXR := true;

  { overvoltage right }

if XUOVRR and UEAVMR and
  ( UVOBSR > UVOMAX ) then

   BUOVVR := true;

   { undervoltage right }

if ( UVOBSR < UVBTRY ) and IGONRE then

   BUNDVR := true;

   { start control }

if CBSTPR and IGONRE and XUSTRR then

   BCBSTR := true;

   { engine cluster right }

if CBECRE and XUICSR and IGONRE then

   BCBINR := true;
```

```
      { isolation right }

    if CBISBR and UMNAVR and
      (UINBSR > CBISOL ) then

      BCBISR := true;

    { bus  tie }

    if CBBSTI and ( UINBST > CBISOL ) or
      ( UINBSR > CBBSTI ) then

      BCBTIE := true;
end     { procedure }

  { selection of the electrical system
    configuration is based on the decision tree  as
    it is presented in Chapter 3  }

  case       {electrical system configuration}

     1.  UPAVBT and CBBSTI and CBISBL or CBISBR;
                    ELECONFA;
      { battery in, bus  tie in, one isolation in}

     2.  UPAVBT and CBBSTI and CBISBL and CBISBR;
                    ELECONFB;
     { battery in, bus  tie in, both isolation in }
     3.  CBBSTI and not CBISBL and not CBISBR;
                ELECONFB;
     { bus  tie in only }

     4.  UPAVBT and not CBBSTI and CBISBL and CBISBR;
                ELECONFC;
     { battery in, no tie, one to battery}

     5.  UPAVBT and not CBBSTI and not CBISBL and CBISBR;
                ELECONFD;
     { battery in, no tie, one to battery}

     6.  UPAVBT and not CBBSTI and CBISBL and not CBISBR;
                ELECONFD;
     { battery in, no tie, one to battery}

     7.  not CBBSTI and not CBISBL and not CBISBR;
                ELECONFD;
     {all independant}

     8.  not UPAVBT and not CBBSTI and CBISBL and CBISBR;
                ELECONFC;
     {no battery, no tie, both to battery}
```

A.1.3 The Thermal Module.

The displayable thermal quantities are computed in this section; these are the cylinder head temperature and the exhaust gas temperature. The temperature of the oil is calculated in the oil system model; inputs to the model are the position of the cowl flaps, the engine cooling characteristics (by function generation) and variables from the engine programmes such air temperature etc.

THERMAL MODEL

List of model variables.

Variable	Type	Description	Unit
HFLINL	Real	Cylinder heat flow rate Left	Kcal/sec
HFLINR	Real	Cylinder heat flow rate Right	Kcal/sec
HTCHLE	Real	Cylinder Head temperature Left	K
HTCHRE	Real	Cylinder Head temperature Right	K
HFCHLE	Real	Heat loss flow rate Left engine	Kcal/sec
HFCHRE	Real	Heat loss flow rate Right engine	Kcal/sec
HFCPRL	Real	Cooling rate factor Left nacelle	
HFCPRR	Real	Cooling rate factor Right nacelle	
HFEXLE	Real	Exhaust gas Heat flow rate Left	Kcal/sec
HFEXRE	Real	Exhaust gas Heat flow rate Right	Kcal/sec
HTEGTL	Real	Exhaust temperature rise Left	K
HTEGTR	Real	Exhaust temperature rise Right	K
HTEGTL	Real	Exhaust gas temperature Left	K
HTEGTR	Real	Exhaust gas temperature Right	K
HTDCHL	Real	Displayable CHT Left engine	K
HTDCHR	Real	Displayable CHT Right engine	K
HTDEGL	Real	Displayable EGT Left engine	K
HTDEGR	Real	Displayable EGT Right engine	K
HGCFLK	Real	Cowling effect (FUNGEN)	
HTEFLE	Real	Left engine thermal efficiency	
HTEFRE	Real	Right engine thermal efficiency	

List of Malfunction Variables.

Variable	Type	Description	Unit
XCFLPL	Bool	Inoperative cowl flaps Left	
XCFLPR	Bool	Inoperative cowl flaps Right	
XCFLSL	Int	Seized Cowl position Left	
XCFLSR	Int	Seized Cowl position Right	
XCHDTL	Bool	CHT ind. out of order Left	
XCHDTR	Bool	CHT ind. out of order Right	
XCHDRL	Real	CHT false indication Left	K
XCHDRR	Real	CHT false indication Right	K
XEGSRL	Bool	EGT ind. out of order Left	
XEGSRR	Bool	EGT ind. out order Right	
XEGSLE	Real	EGT false indication Left	K
XEGSRE	Real	EGT false indication Right	K

List of model constants.

Constant	Type	Description	Unit
HECVEX	Real	Exhaust gases specific heat (V)	Kcal/kgK
HGASHT	Real	Fuel heat product	Kcal/kg
HFINKN	Real	Heat inflow factor	
HCHTKF	Real	Heat transfer factor (cylinders)	
HRCOMA	Real	Compr. ratio to adiab. exp (k-1)	
HRCOAM	Real	Compr. ratio to adiab. exp (1-k)	

```
 procedure  THERMAL;

{ procedure to calculate the cylinder head
 temperature and the exhaust gas temperature;
 input to this subprogramme are the cowl
 flap lever positions and variables from
 the engine procedures }


 begin


{ left CHT calculation }

 HFLINL := EFARLE * HGASHT * EANVLE * HFINKN;

  if XCFLPL then
    CECFLL := XCFLSL;

 HFCPRL := CECFLL * HGCFKL;
 HFCHLE := HFCPRL * ( HTCHLE - ETAAMB );

 HTCHLE := HTCHLE + HCHTKF ( HFLINL - HFCHLE ) * DTIME;

    if not XCHDTL then
       HTDCHL := HTCHLE;
    else
       HTDCHL := XCHDRL;
{ right CHT calculation }

 HFLINR := EFARRE * HGASHT * EANVRE * HFINKN;

  if XCFLPR then
    CECFLR := XCFLSL;

 HFCPRR := CECFLR * HGCFKL;
 HFCHRE := HFCPRR * ( HTCHRE - ETAAMB );

 HTCHRE := HTCHRE + HCHTKF ( HFLINR - HFCHRE ) * DTIME;

    if not XCHDTR then
       HTDCHR := HTCHRE;
    else
       HTDCHR := XCHDRL;

{ left EGT calculation }

 HTEXLE := EFARLE * FUFLLE * HGASHT / HECVEX;

 HFEXLE := HECVEX (( ETAMFL * HRCOMA + HTEXLE ) *
```

```
            HRCOAM - ETAMFL ) - HFCHLE;

  HTEGTL := HTEGTL + HFEXLE * DTIME;

    if XEGSRL then
       HTDEGL := XEGSLE;
    else
       HTDEGL := HTEGTL;
  { right EGT calculation }

  HTEXRE := EFARRE * FUFLRE * HGASHT / HECVEX;

  HFEXRE := HECVEX (( ETAMFR * HRCOMA + HTEXRE ) *
            HRCOAM - ETAMFR ) - HFCHRE;

  HTEGTR := HTEGTR + HFEXRE * DTIME;

    if XEGSRR then
       HTDEGR := XEGSRE;
    else
       HTDEGR := HTEGTL;

  { thermal efficiency left }

  HTEFLE := EWORLE / HFLINL;

  { thermal efficiency right }

  HTEFRE := EWORRE / HFLINR;

  end
```

A.1.4 The Ignition System Module.

The functions of the ignition systems of both engines are simulated in the following programme sections; input originates from the ignition switches of the aircraft panel and engine system variables.

IGNITION SYSTEM

List of System Variables.

Variable	Type	Description	Unit
IGONLE	Bool	Left engine on and running	
IGONRE	Bool	Right engine on and running	

List of System Malfunctions.

Variable	Type	Description	Unit
XIGMLL	Bool	Magneto/Distr. defective Left Left	
XIGMRL	Bool	Magneto/Distr. defective Right Left	
XIGMLR	Bool	Magneto/Distr. defective Left Right	
XIGMRR	Bool	Magneto/Distr. defective Right Right	
XEIGML	Bool	Spark plugs out of order Left	
XEIGMR	Bool	Spark plugs out of order Right	
XEISLL	Int	Number Left ignition Left	
XEISRL	Int	Number Right ignition Left	
XEISLR	Int	Number Left ignition Right	
XEISRR	Int	Number Right ignition Right	

List of System Constants.

Constant	Type	Description	Unit
IGSTTQ	Real	Starter torque normal	Nm
IGSTCT	Int	Ignition time delay	sec
IGDLNP	Int	Ignition delay no priming	sec

```
procedure STARTLEFT;

{ procedure to enable the left engine into
 running mode; input is read by I/O control }

begin

 if (CUSLIL and not XIGMLL) or
    (CUSRIL and not XIGMRL) then
   begin
     if CUSPRL then
     begin
       IGONLE := true;
       IGSTML := 0;
     end;
    else
    begin
     if IGSTML > IGDLNP then
     begin
       IGONLE := true;
       IGSTML := 0;
     end;
    else          { failed start }
      if IGSTML > RELMAX then
        IGSTML := 0;
    end;
    end;
 end    { procedure }
```

```
  procedure STARTRIGHT;

 { procedure to enable the right engine into
  running mode; input is read by I/O control }

begin

  if (CUSLIR and not XIGMLR) or
     (CUSRIR and not XIGMRR) then
    begin
      if CUSPRR then
      begin
        IGONLE := true;
        IGSTMR := 0;
      end;
     else
     begin
      if IGSTMR > IGDLNP then
      begin
        IGONLE := true;
        IGSTMR := 0;
      end;
     else          { failed start }
       if IGSTMR > RELMAX then
          IGSTMR := 0;
      end;
     end;
   end    { procedure }

  procedure IGNITION;

 { procedure to simulate the ignition circuitry;
  there are two magnetos ang two ignition circuits
  on each engine and all separate }

  begin

  { determine whether dead start left engine }

   if not UEAVML and not IGONLE then
  begin
    if CUSSTL and (not CBISBL or (CBBSTI and
       CBISBR)) and not CBSTPL and CUSXPL and
       CBAXPL and not XUAXPL and not XUSTRL and
       UPAVBT then
     begin
      EQSTRL := IGSTTQ * UVOBAT / UVBTRY;
      if XUBTRC then
      EQSTRL := EQSTRL * RANDOM (0,1);

      UINSTL := UVOBAT / URZSTA;
      UBCHAV := UBCHAV - UINSTL * DTIME;
```

```
        { ignition timing loop }

       IGSTML := IGSTML + DTIME;
       if IGSTML > IGSTCT then

            STARTLEFT;
      end;
     else
      if not (CUSLIL and XIGMLL) or
         not (CUSRIL and XIGMRL) and UEAVML then
              IGONLE := false;
           { stop running }
      end;
  end
  { determine whether dead start right engine }

   if not UEAVMR and not IGONRE then
  begin
    if CUSSTR and (not CBISBR or (CBBSTI and
       CBISBR)) and not CBSTPR and CUSXPR and
       CBAXPR and not XUAXPR and not XUSTRR and
       UPAVBT then
     begin
      EQSTRR := IGSTTQ * UVOBAT / UVBTRY;
      if XUBTRC then
      EQSTRR := EQSTRR * RANDOM (0,1);

      UINSTR := UVOBAT / URZSTA;
      UBCHAV := UBCHAV - UINSTL * DTIME;

        { ignition timing loop }

       IGSTMR := IGSTMR + DTIME;
       if IGSTMR > IGSTCT then

            STARTREFT;
      end;
     else
      if not (CUSLIR and XIGMLL) or
         not (CUSRIR and XIGMRL) and UEAVMR then
              IGONRE := false;
           { stop running }
      end;
  end
```

A.1.5 The Oil System Module.

As discussed in Chapter two, oil temperature and pressure are two important parameters that enable the pilot to monitor engine status. These quantities are computed in this section along with the viscous friction that the oil imposes on the revolving crankshaft and the reciprocating piston rods; oil temperature rises due to the that friction and heat transferred by coming in contact with components of higher temperature such as the rods or the piston and cylinder walls. The engine is assumed to have an automatic mechanism that when oil temperature reaches a certain value it routes the fluid through a heat exchanger where oil heat is dissipated in the atmosphere.

OIL SYSTEM

List of System Variables.

Variable	Type	Description	Unit
OTENLE	Real	Oil temperature Left engine	K
OTENRE	Real	Oil temperature Right engine	K
OPRNLE	Real	Oil pressure Left engine	kg/m
OPRNRE	Real	Oil pressure Right engine	kg/m
OHFLGL	Real	Oil heat gain flow rate Left	Kcal/sec
OHFLGR	Real	Oil heat gain flow rate Right	Kcal/sec
OHFLLL	Real	Oil heat loss flow rate Left	Kcal/sec
OHFLLR	Real	Oil heat loss flow rate Right	Kcal/sec
OGVSFT	Real	Viscous friction factor (FUNGEN)	Nmsec
OGTMPR	Real	Oil Temp. rise - mech. (FUNGEN)	K
OTDNLE	Real	Displayed oil Temperature Left	K
OTDNRE	Real	Displayed oil Temperature Right	K
OPRDLE	Real	Displayed oil Pressure Left	kg/m
OPRDRE	Real	Displayed oil Pressure Right	kg/m

List of Malfunction Variables.

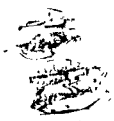
Variable	Type	Description	Unit
XOHEDL	Bool	Heat exchanger/valve stuck closed Left	
XOHEDR	Bool	Heat exchanger/valve stuck closed Right	
XOHEOL	Bool	H/E valve stuck open Left	
XOHEOR	Bool	H/E valve stuck open Right	
XTDINL	Bool	Oil temp. defective Left	
XTDINR	Bool	Oil temp. defective Right	
XTOIFL	Real	False temp. indication Left	K
XTOIFR	Real	False temp. indication Right	K
XPOINL	Bool	Oil press. ind. defective Left	
XPOINR	Bool	Oil press. ind. defective Right	
XPOIFL	Real	False press. Indication left	kg/m
XPOIFR	Real	False press. Indication Right	Kg/m
XOLNBL	Bool	Oil circuit open Left	
XOLNBR	Bool	Oil circuit open Right	

Appendix A — The Programming model

List of System Constants.

Constant	Type	Description	Unit
OTEMAX	Real	Maximum oil temperature	K
OPRNRM	Real	Normal oil pressure	kg/m
OAVLCN	Real	Angular Velocity factor	
OTGLCN	Real	Heat gain factor	
OTLSCN	Real	Heat loss factor	
OTFVLV	Real	Valve trigger temperature	K
OPRCHE	Real	Heat exchanger pressure drop	kg/m
OCHTFR	Real	Heat tranfer factor	

```
procedure  OILSYSTEM;

{ procedure to simulate the effects of the
 lubrication subsystem of the powerplant and
 the displayable oil parameters }

begin

    { oil heat gains and losses left }

 OHFLGL := OTGLCN * ( HTCHLE / OCHTFR +
           OGTMPR ( EXTMLE, OTENLE ));

 if ( OTENLE > OTFVLV ) or XOHEDL then
      OHFLLL := OTLSCN * ( OTENLE - ETAAMB );

  if XOHEDL then
     OHFLLL := 0;

  { oil temperature left }

  OTENLE := OTENLE + ( OHFLGL - OHFLLL ) * DTIME;

  if OTENLE > OTEMAX then
     OTENLE := OTEMAX;
    if not XTOINL then
       OTDNLE := OTENLE;
    else
       OTDNLE := XTOIFL;

    { oil heat gains and losses right }

 OHFLGR := OTGLCN * ( HTCHRE / OCHTFR +
           OGTMPR ( EXTMRE, OTENRE ));

 if ( OTENRE > OTFVLV ) or XOHEDR then
      OHFLLR := OTLSCN * ( OTENRE - ETAAMB );

  if XOHEDR then
     OHFLLR := 0;

  { oil temperature right }

  OTENRE := OTENRE + ( OHFLGR - OHFLLR ) * DTIME;

  if OTENRE > OTEMAX then
     OTENRE := OTEMAX;
    if not XTOINR then
       OTDNRE := OTENRE;
    else
       OTDNRE := XTOIFR;
```



```
{ oil pressure left engine }

OPRNLE := OPRNRM ( OAVLCN * EAVLLE -
          OGVSFT ( EXTMLE, OHFLGL ));

if XOHEOL and not XOHEDL then
   OPRNLE := OPRNLE * OPRCHE;

if XOLNBL then
   OPRNLE := 0;

if not XPOINL then
   OPRDLE := OPRNLE;
else
   OPRDLE := XPOINL;


{ oil pressure right engine }

OPRNRE := OPRNRM ( OAVLCN * EAVLRE -
          OGVSFT ( EXTMRE, OHFLGR ));

if XOHEOR and not XOHEDR then
   OPRNRE := OPRNRE * OPRCHE;

if XOLNBR then
   OPRNRE := 0;

if not XPOINR then
   OPRDRE := OPRNRE;
else
   OPRDRE := XPOINL;

end   { procedure }
```

A.1.6 The Propulsion System Module.

The propulsion system consists of the propellers, the propeller governors and the blade pitch control levers. It is considered as a disturbance input to the engine model along with the propeller governor in terms of torque loading. The propeller torque is calculated from data pertinent to the propeller by function generation; since the chart is expressed in terms of propeller power, torque is derived from that quantity. Input to the model are the blade pitch settings and variables from the engine and external models.

PROPULSION SYSTEM

List of System Variables.

Variable	Type	Description	Unit
JAVRLP	Real	Propeller Advance ratio Left	
JAVRRP	Real	Propeller Advance ratio Right	
JTHRLP	Real	Propeller Thrust Left	kg
JTHRRP	Real	Propeller Thrust Right	kg
JTRQLP	Real	Propeller Torque Left	Nm
JTRQRP	Real	Propeller Torque Right	Nm
JPWRLP	Real	Propeller Power Left	Joule/sec
JPWRRP	Real	Propeller Power Right	Joule/sec
JCFPWR	Real	Coefficient of Power	
JBANLP	Real	Blade angle left propeller	deg.
JBANRP	Real	Blade angle right propeller	deg.
JBPTLP	Real	Blade pitch left propeller	m
JBPTRP	Real	Blade pitch right angle	m

List of Malfunction Variables.

Variable	Type	Description	Unit
XJBPDL	Bool	Uncontrollable pitch Left	
XJBPDR	Bool	Uncontrollable pitch Right	
XJGVLL	Bool	Pitch linkage error Left	
XJGVLR	Bool	Pitch linkage error Right	
XJGVRL	Real	False linkage factor Left	
XJGVRR	Real	False linkage factor Right	
XJBPFL	Bool	Pitch stuck at feather Left	
XJBPFR	Bool	Pitch stuck at feather Right	
XJBPHL	Bool	Pitch stuck at high Left	
XJBPHR	Bool	Pitch stuck at high Right	
XJBPNL	Bool	Not feathering Left	
XJBPNR	Bool	Not feathering Right	
XJBPLL	Bool	Pitch stuck at low Left	
XJBPLR	Bool	Pitch stuck at low Right	

List of System Constants.

Constant	Type	Description	Unit
JBLRAD	Real	Propeller blade radius	m
JAVRFC		Pi over radius	
JBPCNL		Blade pitch const. ovr 3 * Pi * radius	
JAVCBP		Power constant	
JBPFAN		Feather blade angle	
JQPGMX	Real	Maximum Governor torque	Nm
JGCNST	Real	Governor constant	
JBPLPA		Low blade angle	
JBPHPA		High blade angle	

```
{ procedure to simulate the propulsion
  system of a light twin; the propellers are
  counterotating, two blade, of 60 cm radius;}
begin
    { left propeller parameters }
   case
    1.  XJBPFL;
    2.  XJBPHL;
        JBANNP := JBPHPA;
    3.  XJBPLL;
        JBANNP := JBPLPA;
    else if XJBPDL then
            JBPTLP := random (0.1,2);
         JBANLP := arctan ( JBPTLP * JBPCNL );
         if XJBPNL and ( JBANLP > JBPLPA ) then
           JBANLP := JBPLPA;
        if XJGVRL then
           JBANLP := ( JBANLP + XJGVRL ) mod JBPFAN;
             if JBANLP < 1 then
                JBANLP := 1;
     {  propeller advance ratio left }
     JAVRLP := AFWVEL * JAVRFC / EANVLL;
     {  coefficient of power left }
     JCFPWL := JGPWPR ( JAVRLP, JBANLP );
     {  propeller power left }
     JPWRLP := EDAAMB * JAVCBP * JCFRWL;
     {  propeller torque left }
     JTRQLP := JPWRLP / EANVLL;
   { governor torque left }
   EQPGVL := JQPGMX * JBPTLP * AFWVEL * JGCNST;
    { right propeller parameters }
   case
    1.  XJBPFR;
        JBANRP := JBPFAN;
    2.  XJBPHR;
        JBANLP := JBPHPA;
    3.  XJBPLR;
        JBANLP := JBPRPA;
    else if XJBPDR then
            JBPTRP := random (0.1,2);
         JBANRP := arctan ( JBPTRP * JBPCNL );
         if XJBPNR and ( JBANRP > JBPLPA ) then
           JBANRP := JBPLPA;
        if XJGVRR then
           JBANRP := ( JBANRP + XJGVRR) mod JBPFAN;
             if JBANRP < 1 then
                JBANRP := 1;
     {  propeller advance ratio right }
     JAVRRP := AFWVER * JAVRFC / EANVLR;
     {  coefficient of power right }
     JCFPWR:= JGPWPR ( JAVRRP, JBANRP );
     {  propeller power right }
```

```
    JPWRRP := EDAAMB * JAVCBP * JCFRWR;
    { propeller torque right }
    JTRQRP := JPWRRP / EANVLR;
  { governor torque right }
  EQPGVR := JQPGMX * JBPTLP * AFWVEL * JGCNST;
end.   { procedure }
```

A.1.7 The Aeroengine System Module.

Both aeroengines are represented in this section; all the external parameters are derived to this point and they can be included in the engine models. Angular velocity is calculated by integrating angular acceleration . After computing each engine's brake torque all the disturbance inputs in terms of loading torque are subtracted so the system becomes balanced. Frictional losses, dry and viscous are introduced in the gross torque calculations. Carburator icing is inluded in both engine systems.

AERO ENGINE SYSTEM.

List of System Variables.

Variable	Type	Description	Unit
EANVLL	Real	Angular Velocity Left engine	rad/sec
EANVLR	Real	Angular Velocity Right engine	rad/sec
EDANVL	Real	Displayable angular velocity Left	rad/sec
EDANVR	Real	Displayable angular velocity Right	rad/sec
EQGRSL	Real	Gross generated torque Left	Nm
EQGRSR	Real	Gross generated torque Right	Nm
EQBRKL	Real	Brake torque Left engine	Nm
EQBRKR	Real	Brake torque Right engine	Nm
EADNCL	Real	Carburator inlet air density Left	kg/m
EADNCR	Real	Carburator inlet air density Right	kg/m
EATMCL	Real	Carburator inlet air temperature L	K
EATMCR	Real	Carburator inlet air temperature R	K
EAVLCL	Real	Carburator inlet air velocity Left	m/sec
EAVLCR	Real	Carburator inlet air velocity Right	m/sec
EATMVL	Real	Venturi tube air temperature Left	K
EATMVR	Real	Venturi tube air temperature Right	K
EADNVL	Real	Venturi tube air density Left	kg/m
EADNVR	Real	Venturi tube air density Right	kg/m
EAVLVL	Real	Venturi tube air velocity Left	m/sec
EAVLVR	Real	Venturi tube air velocity Right	m/sec
EAPRML	Real	Manifold air pressure Left	kg/m
EAPRMR	Real	Manifold air pressure Right	kg/m
EADNML	Real	Manifold air density Left	kg/m
EADNMR	Real	Manifold air density Right	kg/m
EAVLML	Real	Manifold air velocity Left	m/sec
EAVLMR	Real	Manifold air velocity Right	m/sec
ENRFNA	Real	Bernoulli's equation for air density	
ENRDVA	Real	First derivative of the Bernoulli equation	
EPRAMB	Real	Ambient air pressure	Pa
ETAAMB	Real	Ambient air temperature	K
ETDPNT	Real	Air Dew point	K
EDAAMB	Real	Ambient air density	kg/m
ECTHRL	Real	Throttle lever position Left	
ECTHRR	Real	Throttle lever position Right	
ECMIXL	Real	Mixture lever position Left	
ECMIXR	Real	Mixture lever position Right	
ECCHTL	Real	Carburator Heat lever position Left	
ECCHHTR	Real	Carburator Heat lever position Right	

List of System Malfunctions

Variable	Type	Description	Unit
XERPML	Bool	Inoperative RPM meter Left	
XERPMR	Bool	Inoperative RPM meter Right	
XERPAL	Real	Invalid RPM indication Left	rpm
XERPAL	Real	Invalid RPM indication Right	rpm
XERPIL	Bool	Intermittent RPM meter Left	
XERPIR	Bool	Intermittent RPM meter Right	
XEMFIL	Bool	Inoperative Manifold pressure meter Left	
XEMFIR	Bool	Inoperative Manifold pressure meter Right	
XEMFIL	Real	Invalid manifold pressure indication L	
XEMFIR	Real	Invalid manifold pressure indication R	
XEMFNL	Bool	Intermittent manifold pressure Left	
XEMFNR	Bool	Intermittent manifold pressure Right	
XEICEL	Bool	Carburator icing Left	
XEICER	Bool	Carburator icing Right	
XEICPL	Real	Icing congestion rate Left	m/sec
XEICPR	Real	Icing congestion rate Right	m/sec
XECBHL	Bool	Inoperative Carburator Heat Left	
XECBHR	Bool	Inoperative Carburator Heat Right	
XCCBSL	Bool	Stuck Carb. Heat lever Left	
XCCBSR	Bool	Stuck Carb. Heat lever Right	
XCCBRL	Real	Stuck lever position Left	
XCCBRR	Real	Stuck lever position Right	
XCTHLL	Real	Throttle linkage error Left	
XCTHLR	Real	Throttle linkage error Right	
XEAHUL	Bool	RPM hang up Left engine	
XEAHUR	Bool	RPM hang up Right engine	
XEAHIL	Real	Hang up crossover point Left	
XEAHIR	Real	Hang up crossover point Right	
XETMDL	Bool	Time counter inoperative Left	
XETMDR	Bool	Time counter inoperative Right	
XETMIL	Real	Invalid counter indication Left	hr
XETMIR	Real	Invalid counter indication Right	hr

List of System Constants

Constant	Type	Description	Unit
ECRINT	Real	Inlet cross sectional area	m
ECRVNT	Real	Venturi cross sectionsl area	m
ENRADX	Real	Factor k/k-1	
ENREXP	Real	Adiabatic exponent k-1	
ENRBDX	Real	Factor k+1	
ENMRTO	Real	Area ratio (carburator sections)	
ENCYLD	Real	Cylinder capacity for 2 rads	
ENADEX	Real	Adiabatic exponent C_p/ C_v	
ETQCOE	Real	Torque coefficient	Nm/kg
EDFSCO	Real	Dry frictional loss coefficient	Nm
EINRTM	Real	Total moment of inertia	kgm

```
procedure AEROENG;
{ procedure to compute the engine system
parameters displayable and internal, as
derived in Chapter two }

begin
 if GSTATE then
 begin
  if IGONLE then
 begin
  case
    1. XECBHL;
       ECCHTL := 0;
    2. XECCBSL;
       ECCHTL := XCCBRL;
   { inlet air temperature left }
   EATMCL := ETAAMB + ( HTCHLE - ETAAMB ) * ECCHTL;
   { inlet air density left }
   EADNCL := EDAAMB * ETAAMB / EATMCL;
   EAVLML := ENCYLD * EANVLL;
   {inlet air velocity left }
   EAVLCL := EADNML * EAVLML / EADNCL;
   { air velocity at Venturi orifice }
   EAVLVL := EAVLCL * ENMRTO;
   { air density at Venturi left }
   EADNVL := EADNVL - NWTPRX ( ven := true; );
   { air temperature at Venturi } left }
   EATMVL := ETAAMB * ( EADNVL / EADNCL ) ** ENREXP;
   { carburator icing }
   if EATMVL < ETDPNT then
      CARBICE;
   { manifold air density left }
   if XEAHUL then
      if ( ECTHRL - ECTHXL > XEAHIL ) then
           ECTHRL := ECTHXL;
   EADNML := EADNML - NWTPRX ( ven := false; );
   { manifold pressure left }
   EAPRML := EPRAMB * ( EADNML / EADNCL ) ** ENADEX;
   ECTHXL := ECTHRL;
   if not XEMFPL then
      EDRMFL := EAPRML;
   else
      EDRMFL := XEMFAL;
  if XEMFIL then
     EDRMFL := EDRMFL * random (0,1);
   { Gross engine torque }
   EQGRSL := ETQCOE * EADNML * EGEFTQ ( EANVLL ) *
             EGFAMX;
   { Brake engine torque - net quantity }
   EQBRKL := EQGRSL - OGVSFT ( EXTMLE, EANVLL ) -
             EDFSCO;
   { Integration of angular acceleration }
```

```
    EANVLL := EANVLL + ( EQBRKL - JTRQLP +
              EQSTRL - EQALTL - EQPGVL - EQFUPL -
              EQOLPL ) * DTIME;
    { Displayable angular velocity }
    EDANVL := EANVLL;
    case
       1. XERPML;
          EDANVL := XERPAL;
       2. XERPIL;
          EDANVL := random (0,1);
     { time counter left }
     EXTMLE := EXTMLE + DTIME / 60;
     if not XETMDL then
        EDTMLE := EXTMLE;
    else
        EDTMLE := XETMIL;
    if XEAHUL then
       if ( ECTHRL - ECTHXL > XEAHIL ) then
          ECTHRL := ECTHXL;
end    { left system }

  { right engine }
 if IGONRE then
 begin
  case
   1. XECBHR;
      ECCHTR := 0;
   2. XECCBSR;
      ECCHTR := XCCBRR;
   { inlet air temperature right }
   EATMCR := ETAAMB + ( HTCHRE - ETAAMB ) * ECCHTR;
   { inlet air density right }
   EADNCR := EDAAMB * ETAAMB / EATMCR;
   { manifold suction air velocity right }
   EAVLMR := ENCYLD * EANVLR;
   {inlet air velocity right }
   EAVLCR := EADNMR * EAVLMR / EADNCR;
   { air velocity at Venturi orifice }
   EAVLVR := EAVLCR * ENMRTO;
   { air density at Venturi right }
   EADNVR := EADNVR - NWTPRX ( ven := true; );
   { air temperature at Venturi  right }
   EATMVR := ETAAMB * ( EADNVR / EADNCR ) ** ENREXP;
   { carburator icing }
   if EATMVR < ETDPNT then
      CARBICE;
   { manifold air density right }
   if XEAHUR then
      if ( ECTHRR - ECTHXR > XEAHIR ) then
         ECTHRR := ECTHXR;
   EADNMR := EADNMR - NWTPRX ( ven := false; );
   { manifold pressure right }
   EAPRMR := EPRAMB * ( EADNMR / EADNCR ) ** ENADEX;
   ECTHXR := ECTHRR;
```

```
    if not XEMFPR then
       EDRMFR := EAPRMR;
    else
       EDRMFR := XEMFAR;
  if XEMFIR then
     EDRMFR := EDRMFR * random (0,1);
   { Gross engine torque }
    EQGRSR := ETQCOE * EADNMR * EGEFTQ ( EANVLR ) *
              EGFAMX;
    { Brake engine torque - net quantity }
    EQBRKR := EQGRSR - OGVSFT ( EXTMRE, EANVLR ) -
              EDFSCO;
    { Integration of angular acceleration }
    EANVLR := EANVLR + ( EQBRKR - JTRQLP +
              EQSTRR - EQALTR - EQPGVR - EQFUPR -
              EQOLPR ) * DTIME;
    { Displayable angular velocity }
    EDANVR := EANVLR;
    case
       1.  XERPMR;
           EDANVR := XERPAR;
       2.  XERPIR;
           EDANVR := random (0,1);
     { time counter right }
     EXTMRE := EXTMRE + DTIME / 60;
     if not XETMDR then
        EDTMRE := EXTMRE;
    else
        EDTMRE := XETMIR;
    if XEAHUR then
       if ( ECTHRR - ECTHXR > XEAHIR ) then
          ECTHRR := ECTHXR;
 end { right system }

end  { procedure }
```

```
function NWTPRX ( leng, ven : Bool; ): Real;

{ function to calculate air density at the
  Venturi orifice or at the manifold, by the
  Newton approximation method since
  the expressions are too complicated to
  compute }
var
  thr, densin, densout, velin, velout,
   areaout : Real;
 leng : Bool;
begin
if leng then
 begin
  densin := EADNCL;
  velin := EAVLCL;
  if ven then
  begin
   densout := EADNVL;
   velout  := EAVLVL;
   thr := 0;
   areaout := EARVNL;
   end
  else
   begin
   densout := EADNML;
   velout := EAVLML;
   thr := ECTHRL;
   areaout := EARTHL;
  end
 end
else
 begin
  densin := EADNCR;
  velin := EAVLCR;
  if ven then
  begin
   densout := EADNVR;
   velout  := EAVLVR;
   thr := 0;
   areaout := EARVNR;
   end
  else
   begin
   densout := EADNMR;
   velout := EAVLMR;
   thr := ECTHRR;
   areaout := EARTHR;
   end
 end
{ cross sectional area }
areasqp := areaout / ECRINT;
```

```
{ intermediate calculations }
comprs := EPRAMB * ENRADX / densin;
densrat := densout / densin;
dynpres := densrat * densrat * areasqr *
           velin * velin;
conrat := densrat ** ENREXP;
velsqr := velout * velout;
areaft := areasqr / densin;
densfun := comprs * ( 1 - conrat ) + dynpres -
           velsqr * ( 1 + thr ) / 2;
funderiv := conrat * comprs * ENREXP / densout -
           areaft * densout * areaft * velsqr;
NWTPRX := densfun / funderiv;
end   { function }
```

A.2 THE DATA CONTROL SYSTEM.

The next sections show the flow of control regarding data acquisition, conversion and transfer. Record data structures are used for easy manipulation of both kinds of data.

A.2.1 The Data Transfer System Module.

The displayable data is processed by this subprogram in order that it be converted, scaled and limited in an a suitable format for output; these functions take place for analog values that are to drive the panel indicators. Discrete variables are grouped in byte and/or nibble formats for faster channelling through the parallel ports where after they are broadcast singly to the driver circuitry.

DATA TRANSFER SYSTEM

List of System Variables.

Variable	Type	Description			Unit
BXFVAQ	record :	VALUE	Real	Analogue value	
	:	PORT	Char	Analogue port name	
	:	SLEW	Int	Slew rate code	
	:	NEXT	Int	Pointer	
BXDSCQ	record :	BITE	Int	Group.	
	:	PORT	Char	Discrete port name.	
	:	NEXT	Int	Pointer	
BERPML	Real	RPM Left engine			rpm
BERPMR	Real	RPM Right engine			rpm
BETHEL	Real	Running time hours Left			hr
BETHER	Real	Running time hours Right			hr
BETMEL	Real	Running time minutes Left			min
BETMER	Real	Running time minutes Right			min
BEMFPL	Real	Manifold pressure Left			in Hg
BEMFPR	Real	Manifold pressure Right			in Hg
BEEGTL	Real	EGT Left			F
BEEGTR	Real	EGT Right			F
BFQUAL	Real	Fuel quantity Left tank			US Gal
BFQUAR	Real	Fuel quantity Right tank			US Gal
BFPRSL	Real	Fuel pressure Left			psi
BFPRSR	Real	Fuel pressure Right			psi
BFFLOL	Real	Fuel flow Left			Gal/hr
BFFLOR	Real	Fuel flow Right			Gal/hr
BOTEML	Real	Oil temperature Left			F
BOTEMR	Real	Oil temperature Right			F
BOPRSL	Real	Oil pressure Left			psi
BOPRSR	Real	Oil pressure Right			psi
BECHTL	Real	Cylinder head temperature Left			F
BECHTR	Real	Cylinder head temperature Right			F
BULODL	Real	Alternator load Left			%
BULODR	Real	Alternator load Right			%
BUCINL	Real	Bus Current Left			Amp
BUCINR	Real	Bus Current Right			Amp
BUOVVL	Bool	Overvoltage Indicator Left			
BUOVVR	Bool	Overvoltage Indicator Right			
BUNDVL	Bool	Undervoltage Indicator Left			
BUNDVR	Bool	Undervoltage Indicator Right			
BCBAXL	Bool	Circuit Breaker (CB) Auxiliary pump Left			
BCBAXR	Bool	CB Auxiliary pump Right			
BCBSTL	Bool	CB Start and prime Left			
BCBSTR	Bool	CB Start and prime Right			
BCBINL	Bool	CB Engine instrument cluster Left			
BCBINR	Bool	CB Engine instrument cluster Right			
BCBISL	Bool	CB Bus Isolation Left			
BCBISR	Bool	CB Bus Isolation Right			
BCBAFL	Bool	CB Alternator stator Left			

BCBAFR	Bool	CB Alternator stator Right
BCBTIE	Bool	CB Bus Tie

List of System Constants.

Constant	Type	Description	Unit
BKGIHG	Real	Conv. Factor (kg/m to in HG to mtr.)	
BLTIGT	Real	Conv. Factor (lt to US Gal to mtr.)	
BRDIRV	Real	Conv. Factor (rad/sec to rpm to mtr.)	
BKGIPS	Real	Conv. Factor (kg/m to psi to mtr.)	
BKVIFA	Real	Conv. Factor (K to F to mtr.)	
BPRILD	Real	Conv. Factor (Load to Pct to mtr.)	
BRPMLU	Real	Upper limit RPM	rpm
BRPMLC	Real	Lower limit RPM	rpm
BMFPRU	Real	Upper limit Manifold pressure	in Hg
BMFPRC	Real	Lower limit Manifold pressure	in Hg
BEGTLU	Real	Upper limit EGT	F
BEGTLC	Real	Lower limit EGT	F
BFQULU	Real	Upper limit Fuel quantity	Us Gal
BFQULC	Real	Lower limit Fuel quantity	US Gal
BFPRLU	Real	Upper limit Fuel pressure	psi
BFPRLU	Real	Lower limit Fuel pressure	psi
BFFLLU	Real	Upper limit Fuel flow	US Gal/hr
BFFLLC	Real	Lower limit Fuel flow	US Gal/hr
BOTMLU	Real	Upper limit Oil Temperature	F
BOTMLC	Real	Lower limit Oil Temperature	F
BOPRLU	Real	Upper limit Oil pressure	psi
BORRLC	Real	Lower limit Oil pressure	psi
BCHTLU	Real	Upper limit Cylinder Temperature	F
BCHTLC	Real	Lower limit Cylinder Temperature	F
BLODLU	Real	Upper limit Electrical load	%
BLODLC	Real	Lower limit Electrical load	%
BTWOPI	Real	Two Pi	
BTIMNE	Int	60	min
BTIMTH	Int	10	min
BTIMAX	Real	Time counter zero	hr
BPOINT	Int	Starting Analogue Address	

```
procedure DISPOUT;

{ subprogram to convert, scale and limit
 within specific bounds the values of the
 displayable analogue quantities; discrete
 quantities are packed in byte/nibble format }

begin

  while GSTATE do

   begin

     { analogue variables }

       { engine RPM left }

      n := BPOINT;

       BERPML := ( EDANVL * BRDIRV );
    if BERPML > BRPMLU then
       BERPML := BRPMLU;
  else
       if BERPML < BRPMLC then
          BERPML := BRPMLC;
          BXFVAQ [n] . VALUE := BERPML;
                   n := BXFVAQ [n] . NEXT;

       { time counter }

       BETHEL := EDTMLE div BTIMNE;
       BETMEL := (( EDTMLE - BETHEL ) mod BTIMNE ) div
                    BTIMTH;

     if BETHEL > BTIMAX then
       BETHEL := BTIMAX;
     if BETMEL > BTMMAX then
       BETMEL := BTMMAX;

       BXFVAQ [n] . VALUE := BETHEL;
                n := BXFVAQ [n] . NEXT;

       BXFVAQ [n] . VALUE := BETMEL;
                n := BXFVAQ [n] . NEXT;


       { manifold pressure left }

       BEMFPL := EDRMFL * BKGIHG;

     if BEMFPL > BMFPRU then
       BEMFPL := BMFPRU;
```

```
if BEMFPL < BMFPRC then
   BEMFPL := BMFPRC;

   BXFVAQ [n] . VALUE := BEMFPL;
          n := BXFVAQ [n] . NEXT;

   { EGT left }

   BEEGTL := HTEGTL * BKVIFA;

if BEEGTL > BEGTLU then
   BEEGTL := BEGTLU;
else
   if BEEGTL < BEGTLC then
      BEEGTL := BEGTLC;

   BXFVAQ [n] . VALUE := BEEGTL;
          n := BXFVAQ [n] . NEXT;

   { fuel quantity left wing tank }

   BFQUAL := FUQAIL * BLTIGT;

if BFQUAL > BFQULU then
   BFQUAL := BFQULU ;
else
if BFQUAL < BFQULC then
   BFQUAL := BFQULC;

   BXFVAQ [n] . VALUE := BFQUAL;
          n := BXFVAQ [n] . NEXT;

   { fuel pressure left system }

   BFPRSL := FUPRIL * BKGIPS;

if BFPRSL > BFPRLU then
   BFPRSL := BFPRLU;
else
if BFPRSL < BFQULC then
   BFPRSL := BFQULC;

   BXFVAQ [n] . VALUE := BFPRSL;
          n := BXFVAQ [n] . NEXT;

   { oil temperature left engine }

   BOTEML := OTDNLE * BKVIFA;

if BOTEML > BOTMLU then
   BOTEML := BOTMLU;
else
if BOTEML < BOTMLC then
```

```
      BOTEML := BOTMLC;

      BXFVAQ [n] . VALUE := BOTEML;
             n := BXFVAQ [n] . NEXT;

      { oil pressure }

      BOPRSL := OPRNLE * BKGIPS;

   if BOPRSL > BFPRLU then
      BOPRSL := BFPRLU;
  else
   if BOPRSL < BFPRLC then
      BOPRSL := BFPRLC;

      BXFVAQ [n] . VALUE := BOPRSL;
             n := BXFVAQ [n] . NEXT;

      { CHT left engine }

      BECHTL := HBDCHL * BKVIFA;

   if BECHTL > BCHTLU then
      BECHTL := BCHTLU;
   else
   if BECHTL < BCHTLC then
      BECHTL := BCHTLC;

      BXFVAQ [n] . VALUE := BECHTL;
             n := BXFVAQ [n] . NEXT;

      { electrical load }

      BULODL := ULDMTL * BPRILD;

   if BULODL > BLODLU then
      BULODL := BLODLU;
  else
   if BULODL < BLODLC then
      BULODL := BLODLC;

      BXFVAQ [n] . VALUE := BULODL;
             n := BXFVAQ [n] . NEXT;


      { engine RPM right }

      BERPMR := ( EDANVR * BRDIRV );
   if BERPMR > BRPMLU then
      BERPMR := BRPMLU;
  else
      if BERPMR < BRPMLC then
         BERPMR := BRPMLC;
         BXFVAQ [n] . VALUE := BERPML;
```

```
                n := BXFVAQ [n] . NEXT;

      { time counter }

      BETHER := EDTMLE div BTIMNE;
      BETMER := (( EDTMLE - BETHER ) mod BTIMNE ) div
                BTIMTH;

   if BETHER > BTIMAX then
      BETHER := BTIMAX;
   if BETMER > BTMMAX then
      BETMER := BTMMAX;

      BXFVAQ [n] . VALUE := BETHEL;
             n := BXFVAQ [n] . NEXT;

      BXFVAQ [n] . VALUE := BETMEL;
             n := BXFVAQ [n] . NEXT;


      { manifold pressure right }

      BEMFPR := EDRMFR * BKGIHG;

   if BEMFPR > BMFPRU then
      BEMFPR := BMFPRU;

   if BEMFPR < BMFPRC then
      BEMFPR := BMFPRC;

      BXFVAQ [n] . VALUE := BEMFPL;
             n := BXFVAQ [n] . NEXT;

      { EGT right }

      BEEGTR := HTEGTR * BKVIFA;

   if BEEGTR > BEGTLU then
      BEEGTR := BEGTLU;
   else
      if BEEGTR < BEGTLC then
         BEEGTR := BEGTLC;

      BXFVAQ [n] . VALUE := BEEGTL;
             n := BXFVAQ [n] . NEXT;

      { fuel quantity right wing tank }

      BFQUAR := FUQAIR * BLTIGT;

   if BFQUAR > BFQULU then
      BFQUAR := BFQULU ;
   else
   if BFQUAR < BFQULC then
```

```
      BFQUAR := BFQULC;

      BXFVAQ [n] . VALUE := BFQUAL;
             n := BXFVAQ [n] . NEXT;

      { fuel pressure right system }

      BFPRSR := FUPRIR * BKGIPS;

    if BFPRSR > BFPRLU then
      BFPRSR := BFPRLU;
    else
    if BFPRSR < BFQULC then
      BFPRSR := BFQULC;

      BXFVAQ [n] . VALUE := BFPRSL;
             n := BXFVAQ [n] . NEXT;

      { oil temperature right engine }

      BOTEMR := OTDNLE * BKVIFA;

    if BOTEMR > BOTMLU then
      BOTEMR := BOTMLU;
    else
    if BOTEMR < BOTMLC then
      BOTEMR := BOTMLC;

      BXFVAQ [n] . VALUE := BOTEML;
             n := BXFVAQ [n] . NEXT;

      { oil pressure }

      BOPRSR := OPRNLE * BKGIPS;

    if BOPRSR > BFPRLU then
      BOPRSR := BFPRLU;
   else
    if BOPRSR < BFPRLC then
      BOPRSR := BFPRLC;

      BXFVAQ [n] . VALUE := BOPRSL;
             n := BXFVAQ [n] . NEXT;

      { CHT right engine }

      BECHTR := HBDCHR * BKVIFA;

   if BECHTR > BCHTLU then
      BECHTR := BCHTLU;
    else
    if BECHTR < BCHTLC then
      BECHTR := BCHTLC;
```

```
      BXFVAQ [n] . VALUE := BECHTL;
             n := BXFVAQ [n] . NEXT;

      { electrical load }

      BULODR := ULDMTR * BPRILD;

   if BULODR > BLODLU then
      BULODR := BLODLU;
  else
   if BULODR < BLODLC then
      BULODR := BLODLC;

      BXFVAQ [n] . VALUE := BULODL;
             n := BXFVAQ [n] . NEXT;

      { discrete variables }
   m := 0;

      { overvoltage left }

   if BUOVVL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 1;

      { undervoltage left }

   if BUNDVL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 2;

      { auxiliary pump left }

   if BCBAXL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 4;

      { start control left }

   if BCBSTL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 8;

      { instrument cluster left }

   if BCBINL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 16;

      { isolation left }

   if BCBISL then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 32;

      { alternator stator left }

   if BCBAFL then
      BXDSCQ [m] . BITE := BXDSCQ [m] .BITE + 64;
```

```
      { bus  tie }

   if BCBTIE then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 128;

      { overvoltage right },

   if BUOVVR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 1;

      { undervoltage right }

   if BUNDVR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 2;

      { auxiliary pump right }

   if BCBAXR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 4;

      { start control right }

   if BCBSTR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 8;

      { instrument cluster right }

   if BCBINR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 16;

      { isolation right }

   if BCBISR then
      BXDSCQ [m] . BITE := BXDSCQ [m] . BITE + 32;

      { alternator stator right }

   if BCBAFR then
      BXDSCQ [m] . BITE := BXDSCQ [m] .BITE + 64;


  end  { while }

end  { procedure }
```

```
procedure WRITEOUT;

{ procedure to invoke an assembler
 routine to transfer data to the
 output ports of the system in integer
 format }

const
     alast : last analogue port  address.
     dlast : last discrete port address.

var
    { passed parameters }
   datum, port, slew, n, m : integer (hex);

begin

 n := m := o;

while GSTATE and ( n  < > LAST ) do

 begin


      datum := round ( BXFVAQ [n] . VALUE ) Hx;
      port := BXFVAQ [n] . PORT Hx;
      slew := BXFVAL [n] . SLEW Hx;
         n := BXFVAL [n] . NEXT;

      WRITEF (( port, datum, slew); ASSEMB; );

  end  { while }

   { discrete output }

    while GSTATE and ( m < > dlast ) do
     begin

      datum := BXDSCQ [m] . VALUE Hx;
      port  := BXDSCQ [m] . PORT Hx;
      slew  := 0;
          m := BXDSCQ [m] . NEXT;

      WRITEF (( datum, port, slew ); ASSEMB;);

    end   { while }

end  { procedure }
```

A.2.2 The Data Aquisition System Module.

Both analogue and discrete data types are handled in this section; the analogue data is entered in integer format and is stored in a record data structure (expandable). The same applies to the digital input which is afterwards masked and decomposed into the respective Boolean variables; analogue variables are all of type Real.

DATA ACQUISITION SYSTEM.

List of System Variables.

Variable	Type	Description			Unit
CBXAIQ	Record :	VALUE	Int	Analogue value	
		PORT	Char	Port name	
		NEXT	Int	Pointer	
CBXDIQ	Record :	BITE	Int	Group value	
		PORT	Char	Port name	
		NEXT	Int	Pointer	

List of System Constants

Constant	Type	Description	Unit
IETHCN	Real	Conversion factor Throttle	
IEPICN	Real	Conversion factor Pitch	
IEMXCN	Real	Conversion factor Mixture	
IECHCN	Real	Conversion factor Carb. Heat	
IECFCN	Real	Conversion factor Cowl Flaps	

```
procedure DISINDAT;

{procedure to convert and scale analogue
input and decompose groupped data into
usable Boolean variables }

begin

 n := m := 0;


  while GSTATE and ( n < alast ) do
  begin

  { throttle left }

  ECTHRL := CBXAIQ [n] . VALUE * IETHCN;
       n := CBXAIQ [n] . NEXT;

  { pitch left }

  ECPTCL := CBXAIQ [n] . VALUE * IEPICN;
       n := CBXAIQ [n] . NEXT;

  { mixture left }

  ECMIXL := CBXAIQ [n] . VALUE * IEMXCN;
       n := CBXAIQ [n] . NEXT;

  { carburator heat left }

  ECCHTL := CBXAIQ [n] . VALUE * IECHCN;
       n := CBXAIQ [n] . NEXT;

  { cowl flaps left }

  ECFLPL := CBXAIQ [n] . VALUE * IECFCN;
       n := CBXAIQ [n] . NEXT;

  { throttle right }

  ECTHRR := CBXAIQ [n] . VALUE * IETHCN;
       n := CBXAIQ [n] . NEXT;

  { pitch right }

  ECPTCR := CBXAIQ [n] . VALUE * IEPICN;
       n := CBXAIQ [n] . NEXT;

  { mixture right }

  ECMIXR := CBXAIQ [n] . VALUE * IEMXCN;
```

```
        n := CBXAIQ [n] . NEXT;

    { carburator heat right }

    ECCHTR := CBXAIQ [n] . VALUE * IECHCN;
        n := CBXAIQ [n] . NEXT;

    { cowl flaps right }

    ECFLPR := CBXAIQ [n] . VALUE * IECFCN;
        n := CBXAIQ [n] . NEXT;


 end   { while }

     { discrete data }


    while GSTATE and ( m < > dlast ) do
    begin

    bite := CBXDIQ [m] . BITE;
    { auxiliary pump switch left }

    CUSXPL := bite and 1;

    { prime switch left }

    CUSPRL := bite and 2;

    { start switch left }

    CUSSTL := bite and 4;

    { alternator field switch left }

    CUSAFL := bite and 8;

    { Ignition switch left left }

    CUSLIL := bite and 16;

    { Ignition switch left right }

    CUSRIL := bite and 32;

    { battery switch }

    CUSBTR := bite and 64;

    { fuel selector valve left }

    CFSLVL := bite and 128;    { on }
```

```
        m : = CBXDIQ [m] . PORT;
        { next group }
    bite := CBXIDQ [m] . BITE;
    { fuel selector valve left }
    CFSVCL := bite and 1;
    { CB auxiliary fuel pump left }
    CBAXPL := bite and 2;
    { CB start control left }
    CBSTPL := bite and 4;
    { CB engine cluster left }
    CBECLE := bite and 8;
    { CB isolation left }
    CBISBL := bite and 16;
    { CB bus  tie }
    CBBSTI := bite and 32;
    bite := CBXDIQ [m] . BITE;
    { auxiliary pump switch right }
    CUSXPR := bite and 1;
    { prime switch right }
    CUSPRR := bite and 2;
    { start switch right }
    CUSSTR := bite and 4;
    { alternator field switch right }
    CUSAFR := bite and 8;
    { Ignition switch right left }
    CUSLIR := bite and 16;
    { Ignition switch right right }
```

```
    CUSRIR := bite and 32;

    CUSBTR := bite and 64;
    { fuel selector valve right }

    CFSLVR := bite and 128; { on }

        m := CBXDIQ [m] . PORT;

        { next group }

    bite := CBXIDQ [m] . BITE;

    { fuel selctor valve right }

    CFSVCR := bite and 1;

    { CB auxiliary fuel pump right }

    CBAXPR := bite and 2;

    { CB start control right }

    CBSTPR := bite and 4;

    { CB engine cluster right }

    CBECLE := bite and 8;

    { CB isolation right }

    CBISBR := bite and 16;


end     { procedure }
```

```
procedure READIN;

{ procedure to read in analogue and digital
 data by invoking an Assembler routine }

var
  m, n : integer;
  datum, port : integer;

begin

   m := n := 0;
 while GSTATE and ( n < alast ) do

  begin

   READEF ( datum, port; ASSEMB; );

   CBXAIQ [n] . VALUE := datum;
   CBXAIQ [n] . PORT  := port;
                  n := CBXAIQ [n] . NEXT;

  end   { while }

   { discrete data }

 while GSTATE and ( m < last ) do

   begin

      READEF ( bite, group; ASSEMB; );

      CBXDIQ [m] . bite  := bite;
      CBXDIQ [m] . group := group;
                    m := CBXAIQ [m] . NEXT;

    end     { while }

end  { procedure }
```

APPENDIX B

ANALOGUE COMPUTER MODEL.

Before proceeding into programming model derivation, an EAI 640 analogue computer was used to evaluate the critical component of the powerplant model, based on the equations derived in Chapter two. Although limited, the output displayed a satisfactory level of stability and coherence. This test is presented briefly in this Appendix.

B.1 The analogue model.

The analogue computer model is limited by the operational status of the machine at the time it were used; however it can clearly be seen by the generated output that its behaviour is satisfactory and thus further development was warranted. The diode function generator was used to establish function generation for only one parameter. Notable is the behaviour of the carburator model, approximating that that of the actual component. The oil viscous friction was also represented in a realistic form; finally the throttling process behaved as expected.

B.2 The analogue computer circuit.

The analogue model is shown with the computer circuit in Fig. B.1. The diode fuction generator was programmed to produce the fuel/air ratio efficiency factor.

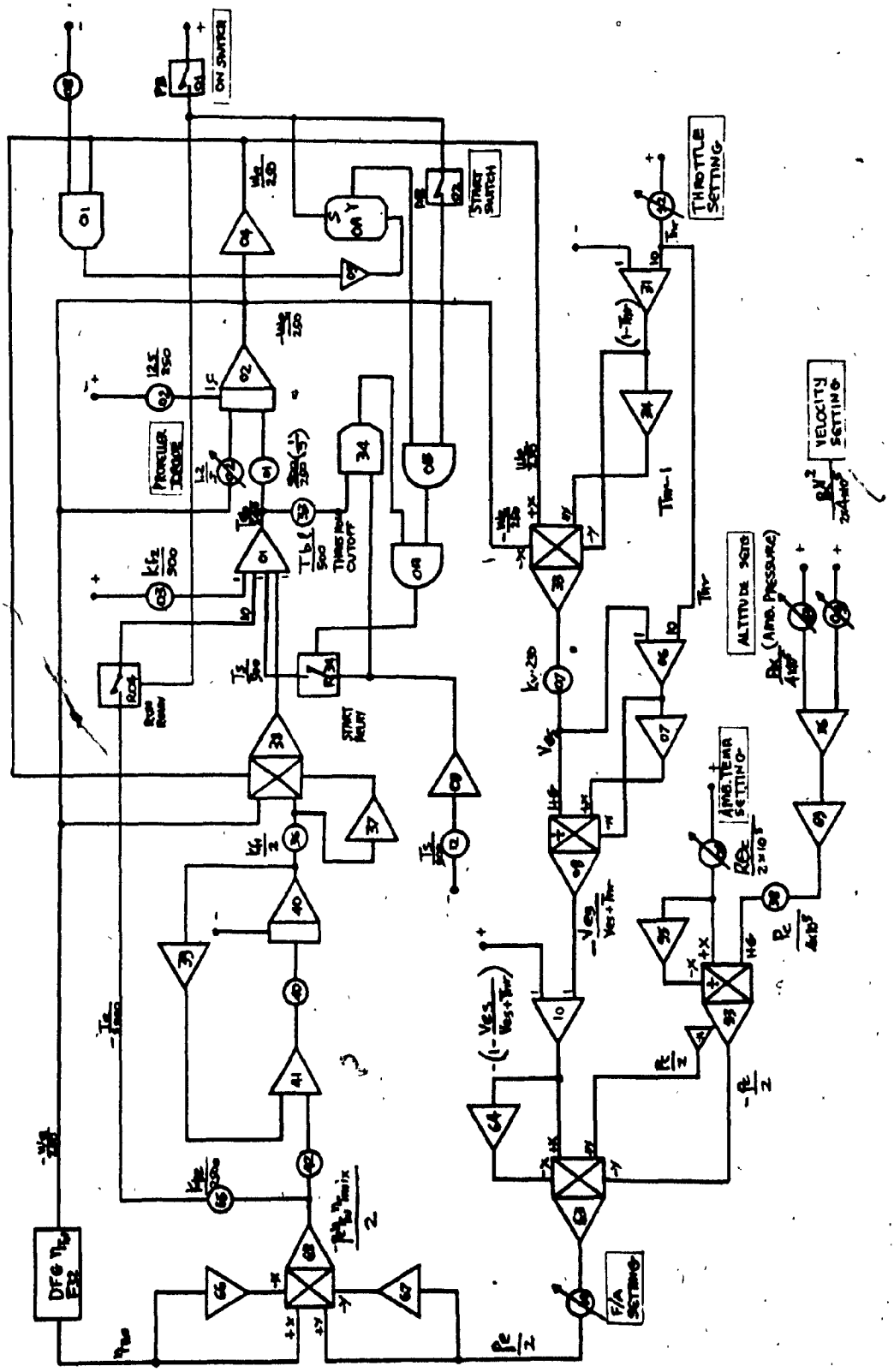


Fig. B.1. The analogue computer model.

2.2.3 Model behaviour.

The output generated by the analogue computer is presented in the next pages for comparison purposes to those of the digital simulation in Chapter six. Inputs to the circuit were the engine controls, the operational altitude and the ambient air temperature; the output represents angular velocity.

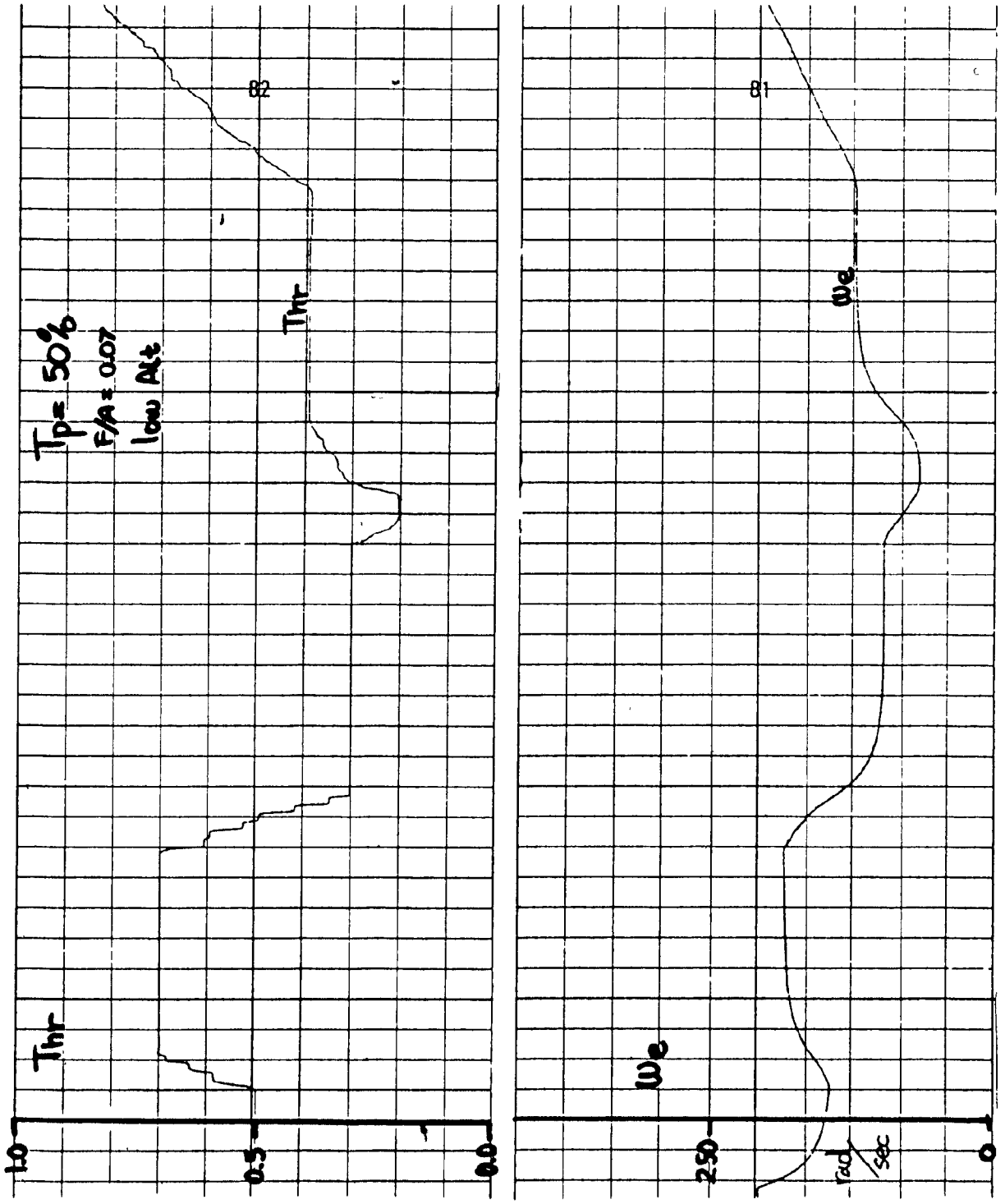


Fig. B.2. Response to manual Throttling.

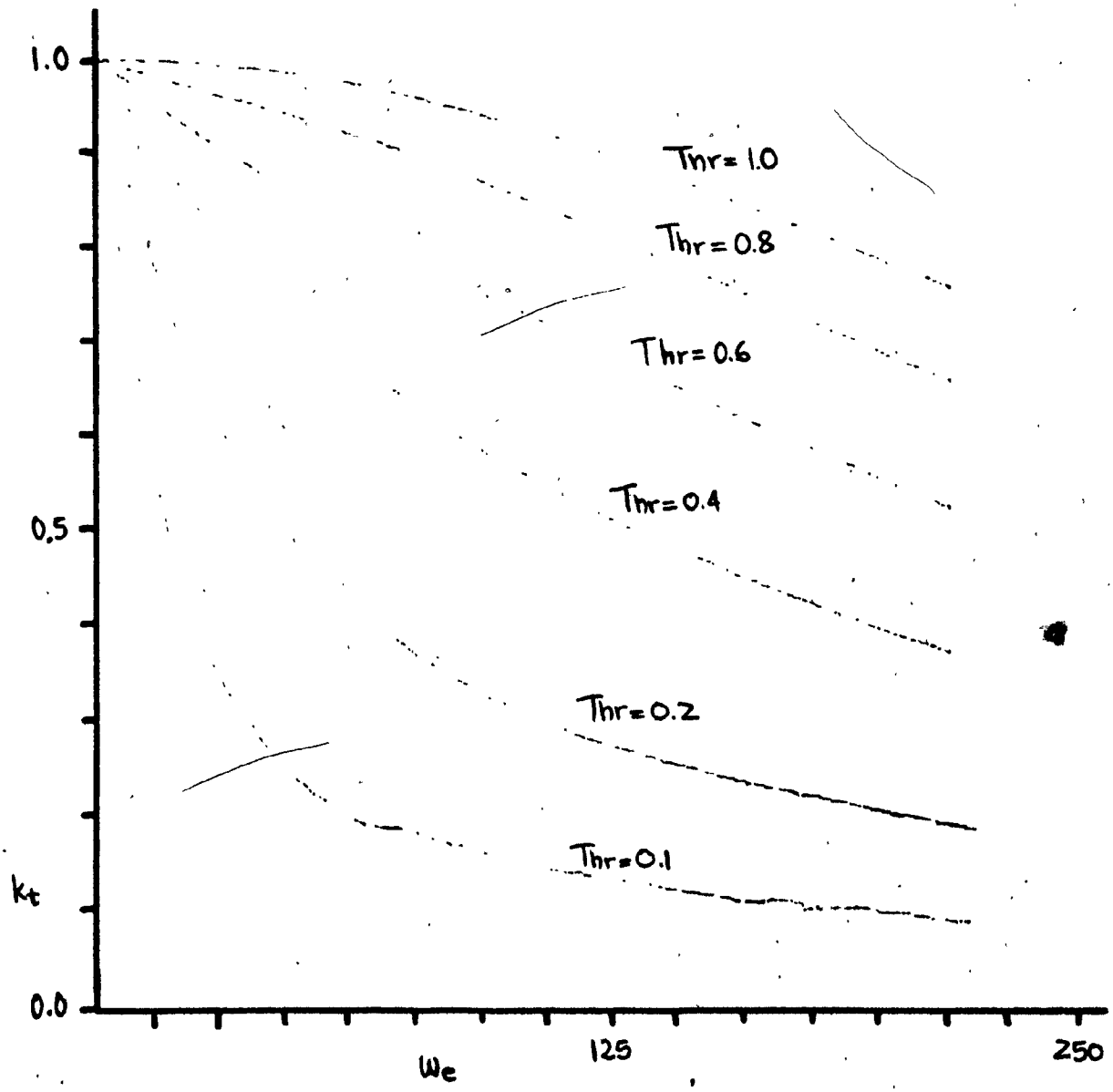


Fig. B.3. Carburator behaviour.

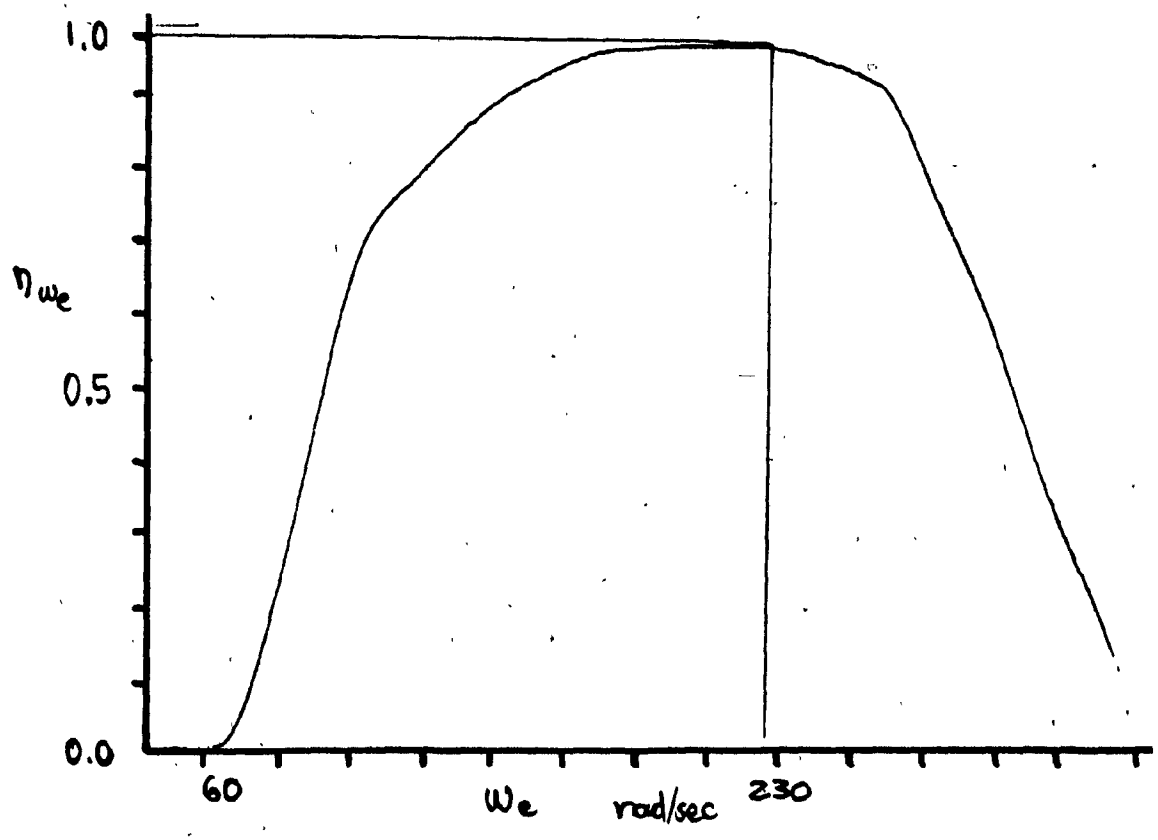


Fig. B.4. Diode function generator output.

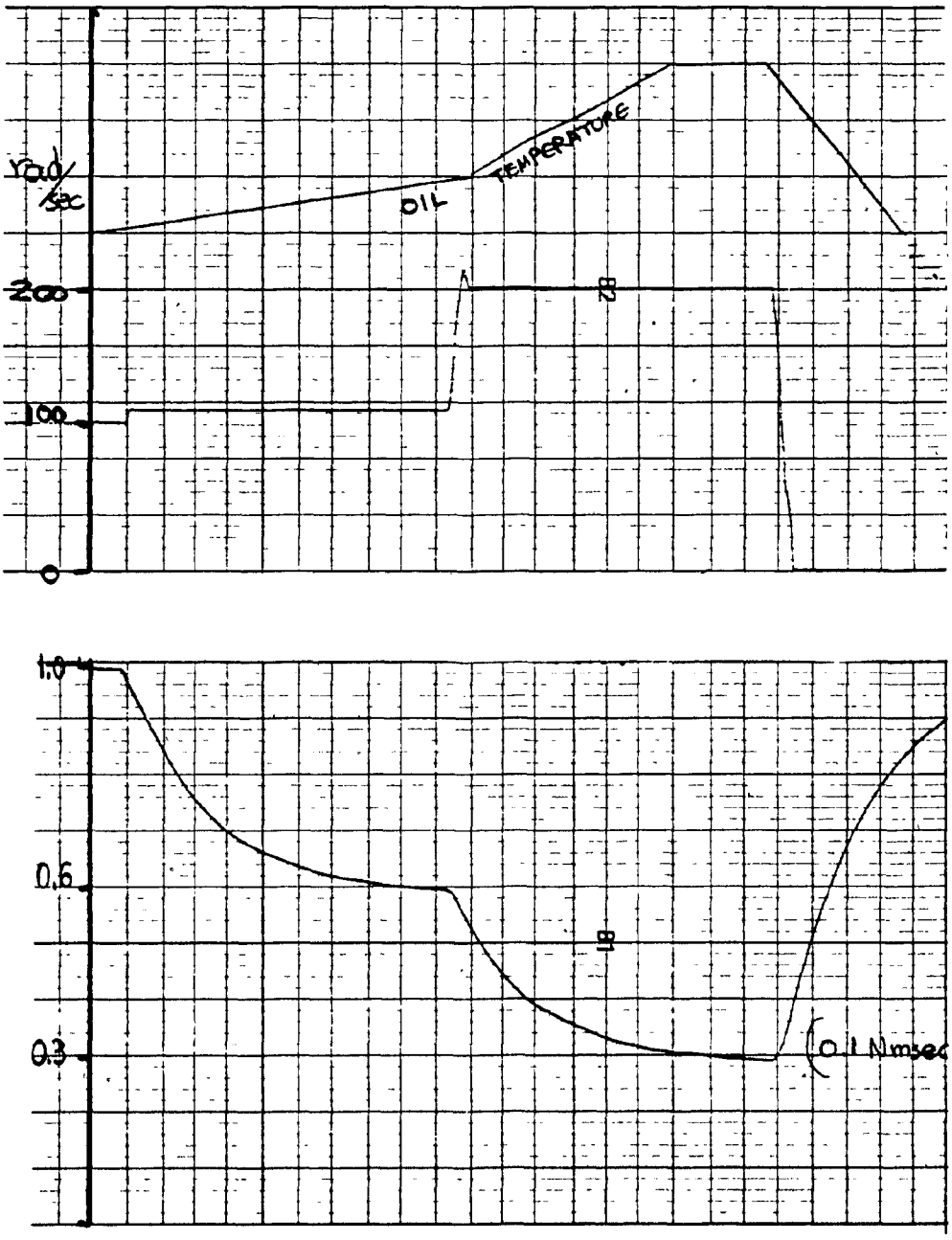


Fig. B.5. Reproduced oil system characteristics.

APPENDIX C

AEROENGINE SPECIFICATIONS.

Specification data for the O 360 AVCO-Lycoming aeroengine are listed in the next pages for comparison purposes.

AVCO LYCOMING SPECIFICATION

NO. 2525-B

DETAIL SPECIFICATION FOR

ENGINE, AIRCRAFT, MODEL

O-360-E1A6D

180 HORSEPOWER

DIRECT DRIVE

AVCO LYCOMING WILLIAMSPORT DIVISION

AVCO CORPORATION

WILLIAMSPORT, PA.

SUMMARY OF SPECIFICATIONS

Model	O-360-E1A6D
FAA Type Certificate	286
Rated Max. Cont. HP @ BSFC	180 @ .49
Rated Speed, RPM	2700
Cruise Speed (Economy), RPM	2350
Fuel Consumption, Cruise, gph @ 75% Rated Power	10.8
65% Rated Power	8.8
60% Rated Power	8.2
Propeller Drive Ratio	1:1
Propeller Shaft Rotation	Clockwise
Bore, Inches	5.125
Stroke, Inches	4.375
Displacement, Cubic Inches	361
Compression Ratio	9.00:1
Dry Weight, Pounds	300
Dimensions: Height, Inches	21.08
Width, Inches	33.81
Length, Inches	32.26
Oil, SAE Number, Above 60°F	50
30°F to 90°F	40
0°F to 70°F	30
Below 10°F	20
Oil Sump Capacity, Quarts	6
Fuel Aviation Grade, Octane	100/130
Carburetor, Marvel Schebler	HA-6
Magneto, Bendix	D4RN-2021
Tachometer Drive, Ratio to C'Shaft & Rotation	0.5:1 - Clockwise
Starter Drive, Ratio to C'Shaft & Rotation	16.55:1 - C'Clockwise
Starter, Prestolite, Geared, 3.38:1) 12 Volt	Standard
With Switch) 24 Volt	Optional
Alternator Drive, Ratio to C'Shaft & Rotation	3.25:1 - Clockwise
Alternator, Prestolite, (12 Volt, 60 Amp)	Standard
(12 Volt, 70 Amp)	Optional
(24 Volt, 70 Amp)	Optional
AN Vacuum Pump Drive, Ratio to C'Shaft & Rotation	1.313:1 - C'Clockwise
Optional Dual Drive Mounting on Vacuum Pump Drive Pad	
Vacuum Pump-Hyd. Pump Drive, Ratio & Rotation	1.313:1 - C'Clockwise
Prop. Gov. Drive, AN20010, Type XX, Ratio & Rotation	1.000:1 - Clockwise
Fuel Pump Drive, Diaphragm Type	Optional
Fuel Pump, AC	Optional

I. GENERAL SPECIFICATIONS

The following Avco Lycoming* drawings and engine power curves form a part of this specification:

No. 13324 - Fuel Flow Curve
No. 13325 - Sea Level and Altitude Performance Curve
No. 13326 - Cooling Air Requirements
No. 63482 - Installation - O-360-E Engine Series
-- - Schematic Wiring Diagram-Self Rectifying Alternator

II. TYPE

This specification covers the following type of Avco Lycoming aircraft engine:

1. Avco Lycoming aircraft Model O-360-E1A6D is a direct drive, four cylinder, horizontally opposed, air cooled engine. This engine is supplied at the factory with automotive type alternator and starter. Available as optional equipment are an AN propeller governor drive, a diaphragm type fuel pump drive, a diaphragm type fuel pump, and an AN vacuum pump drive and dual vacuum-hydraulic drive.

2. FAA Type Certificate No. 286 has been issued for this engine.

III. DETAIL REQUIREMENTS

1. Rating. This engine has a rated maximum continuous power of 180 HP at 2700 RPM at standard sea level conditions using 100/130 octane aviation grade fuel. Power output of production engines may vary within +5% and -2% of the rating.

2. Bore, Stroke and Displacement. This engine has a bore of 5.125 inches, a stroke of 4.375 inches and a piston displacement of 361.0 cubic inches.

3. Compression Ratio. This engine has a compression ratio of 9.00:1.

4. Weight. The average dry weight of this engine, including shielded ignition and spark plugs, tachometer drive, magneto, carburetor, starter and alternator, is 300 pounds. For detail weights see Section V.

5. Center of Gravity Position. The center of gravity of this engine, including starter and alternator, is located .95 inches below the centerline of the crankshaft, on the left to right crankshaft centerline and 13.90 inches from the front face of the propeller mounting flange.

6. Moment of Inertia. The moments of inertia of this engine, including starter and alternator, but less installation parts, and weighing 300 lb., are as follows: about the axis parallel to the crankshaft centerline 44.55 in. lb. $sec.^2$, about the vertical axis 62.60 in. lb. $sec.^2$, about the axis parallel to centerline of cylinders 41.35 in. lb. $sec.^2$. All axes pass through the center of gravity.

*NOTE: Wherever reference is made to "Avco Lycoming" the Avco Lycoming Williamsport Division, is implied.

7. Overall Dimensions. The overall dimensions of this engine, including the carburetor and magnetos, are as follows:

Height - 21.08 Inches
Width - 33.81 Inches
Length - 32.26 Inches *

* Measured from front face of propeller flange.

8. Mounting. A rear type 1 (30°) dynafocal mounting is provided by four mounting brackets, two at the top rear and two at the bottom rear of the crankcase, as shown on Avco Lycoming Drawing No. 63482. Provision is made for the use of four .4375 inch mounting bolts with bonded rubber mounts. The location of the mounting bolt holes shall be as shown on the Avco Lycoming Drawing No. 63482. The bonded rubber engine mounts are not furnished as part of the engine. A rear type 2 (18°) dynafocal mounting may be provided as optional equipment.

9. Propeller Drive. The propeller shaft, an integral part of the crankshaft, is the flange type and conforms to Specification AS127, Type 2, modified as shown on Avco Lycoming Drawing No. 63482. The engine provides for supplying oil through the propeller shaft for a single acting controllable pitch propeller. Rotation of the propeller shaft is clockwise viewed from the anti propeller end of the engine. Attaching holes for a propeller spinner are supplied on the starter ring gear carrier, as shown on Avco Lycoming Drawing No. 63482.

10. Propeller Governor. A propeller governor drive AN20010, Type XX, can be furnished as optional equipment, as shown on Avco Lycoming Drawing No. 63482.

11. Magneto. The engine is equipped with one Bendix Type D4RN-2021 (impulse coupling) magneto. The magneto incorporates integral feed through capacitors and require no external noise filter in the ground lead. The direction of rotation of the magneto shaft, viewed from the anti propeller end of the engine, is counterclockwise. The engine firing order is 1-3-2-4. The spark advance shall be 25° BTC.

12. Spark Plugs. This engine is equipped with radio shielded plugs at the option of Avco Lycoming as standard equipment, in accordance with the latest issue of Avco Lycoming Service Instruction No. 1042.

13. Radio Shielding. All weather shielded harness braid on wire type is provided as an integral part of the ignition system.

14. Carburetor. This engine is equipped with a Marvel Schebler Type HA-6 manual mixture control, horizontal carburetor, as standard equipment. The carburetor is provided with a .250-18 NPT hole for fuel inlet connection.

15. Fuel. The use of 100/130 octane aviation grade fuel is recommended for this engine. The fuel pressure (lb./sq. inch above carburetor air entrance pressure) at the inlet to the carburetor shall be as follows:

Operating Conditions	Fuel Pressure
Desired	3
Maximum	8
Minimum	.5

16. Fuel Consumption. The attached fuel flow curve No. 13324 shows lean limit fuel flow at part throttle with manual mixture control.

17. Fuel Pump. The engine may be equipped with a diaphragm type fuel pump and drive, as shown on Avco Lycoming Drawing No. 63482.

18. Priming System. A .125 pipe thread primer boss is provided on all cylinder heads for the installation of a priming system. The priming system can be furnished as special equipment.

19. Oil. This engine is provided with a wet sump pressure oil system having a capacity of six (6) quarts. Safe minimum oil in sump is two (2) quarts. It is recommended that single or multi viscosity aviation grade oils in accordance with latest issue of Avco Lycoming Service Instruction 1014 be used. The following seasonal aviation oil grades and seasonal ambient temperature ranges are recommended:

Average Ambient Temperature	MIL-L-6082-B SAE Grade	MIL-L-22851 Ashless Dispersant SAE Grades	Oil-In Temp. °F Desired	Max.
Above 60°F	50	40 or 50	180	245
30°F to 90°F	40	40	180	245
0°F to 70°F	30	40 or 30	170	225
Below 10°F	20	30	160	210

20. Oil Consumption. The maximum oil consumption for this engine is .008 lb/bhp/hr for power above 75% of rated maximum continuous power and .006 lb/bhp/hr for 75% and lower.

21. Oil Filter. This engine is provided with a full flow oil filter assembly shown on Avco Lycoming Drawing No. 63482 as standard equipment.

22. Oil Temperature Measurement. A thermometer well with a .625-18 NF-3 thread is provided in the oil filter adapter for the installation of an MS28034-1 or equivalent standard type thermometer bulb for measurement of oil temperature.

23. Oil Cooler. Provision is made for the use of a full flow oil cooler with this engine. Oil flow through the cooler system will be approximately 7 gallons per minute and heat rejection will not exceed 475 Btu per minute. NOTE: The oil cooler must be capable of withstanding continuous pressure of 150 lb/in.^{2}. A thermostatic bypass and pressure relief valve is supplied as standard equipment. It limits pressure drop between cooler connections to 75 $\pm$ 15 psi and closes at 185^{o}F oil temperature to put all engine oil flow thru the cooler.

24. Oil Pressure Gage Connection. A .125 pipe tap hole is provided for the installation of an oil pressure gage connection. The oil pressure gage connection must provide for a restricted fitting.

25. Oil Level Gage and Breather. This engine is equipped with a breather connection for an outside line located on the upper rear of the crankcase, as shown on Avco Lycoming Drawing No. 63482. A combination filler cap and oil level gage is installed in the crankcase.

26. Oil Drain Plug. A 1.00-20 thread oil plug with .63" hex head is provided at the left rear of the oil sump for oil drain and removal of suction screen. Two (2) .50 inch pipe plugs located in the bottom rear of the sump are provided as alternate drains.

27. Tachometer. A single SAE standard tachometer drive, Specification AS-54, is provided with this engine as standard equipment in accordance with Avco Lycoming Drawing No. 63482. Rotation is clockwise viewed from the anti-propeller end of the engine.

28. Starter. This engine is equipped with a Prestolite 12 volt, 3.38:1 geared starter, a solenoid operated switch, and a Bendix drive pinion ring gear ratio of 16.556:1, in accordance with the Avco Lycoming Drawing No. 63482, as standard equipment. The direction of rotation of the starter pinion viewed from the anti propeller end of the engine is counter-clockwise. Prestolite 24 volt starter is available as optional equipment. Cranking requirements recommended for engines equipped with these starters are based on the following batteries: one 12 volt, 35 ampere hour aircraft battery for the 12 volt system or two 12 volt, 25 ampere hour battery for the 24 volt system.

29. Alternator. A Prestolite 12 volt, 60 ampere, self rectifying alternator, equipped with a transistor voltage regulator and over voltage relay, as shown on Avco Lycoming Drawing No. 63482, is supplied as standard equipment for this engine. Prestolite 12 volt, 70 ampere and 24 volt, 70 ampere alternators are available as optional equipment. Alternators are supplied with a standard drive ratio of 3.25:1. The alternator blast tube must be connected to a source of cooling air sufficient to prevent alternator temperatures from exceeding maximum limits. The engine may be provided with mounting brackets for installation of a 12 volt, 60 ampere Ford alternator as optional equipment.

30. Vacuum Pump. A vacuum pump drive, in accordance with Avco Lycoming Drawing No. 63482, is available as special equipment. The direction of rotation of the vacuum pump drive, viewed from the anti propeller end of the engine, is counter-clockwise. A dual vacuum pump-hydraulic pump drive mounting on this drive, as shown on Avco Lycoming Drawing No. 63482, is available as optional equipment.

31. Cylinder and Head Cooling Baffles. Intercylinder cooling baffles are supplied with this engine as standard equipment.

32. Cooling Air. The attached Curve No. 13326 shows the cooling air flow and pressure differential required for cooling this engine and its lubricating oil.

33. Exhaust Collector. Provision is made for the installation of an exhaust collector. Exhaust flanges, either stainless or low carbon steel type, are available as optional equipment.

34. Accessory Drives. The direction of rotation, gear ratio, torque rating and permissible overhang moment for the pad of each accessory drive are as follows:

	Type of Drive	Direction of Rotation	Drive Ratio	Maximum Torque (Pound - Inch) Continuous	Maximum Torque (Pound - Inch) Static	Maximum Overhang Moment (Pound - Inch)
Starter	SAE (Auto.)	C-C	16.556:1	--	450	150
Alternator	SAE (Auto.)	C	3.250:1	60	120	175
Vacuum Pump	AND20000**	C-C	1.313:1	70	450	25
Tachometer	SAE	C	.500:1	7	50	5
Prop. Gov.	AND20010*	C	1.000:1	125	1200	40
Fuel Pump	Plunger Op.	--	.500:1	--	--	10
Optional	Dual Drive Mounting on Vacuum Pump Drive Pad					
Vacuum Pump	AND20000**	C-C	1.313:1	70	450	6
Hyd. Pump	AND20000**	C-C	1.313:1	Total	Total	10

* Except for torque limitations.
** Except for rotation.

35. Crankshaft Counterweights. The engine is provided with one 6.3 order and one 8th order pendulum type counterweights as standard equipment.

36. Thermocouples. Provision is made in all cylinder heads for insertion of fittings, Air Force-Navy Drawing No. AN-4076, to take bayonet type thermocouple Air Force-Navy Drawing No. AN5541. Embedded type cylinder base thermocouples may be supplied as special equipment.

37. Installation. Installation of this engine is subject to the approval of the engine manufacturer.

IV. INSPECTION AND TESTS

All engine parts and assemblies used in the construction of this engine are thoroughly inspected for conformity to specifications and drawings. Upon completion of assembly, the engine is thoroughly tested before leaving the factory. After the engine is given a final run to determine fuel consumption, it is prepared for shipment.

The following values, as determined by flight testing, shall not be exceeded under any conditions:

Oil Pressure:	Minimum Idling	-	25 pounds per square inch
	Normal	-	60 to 90 pounds per square inch
	Maximum Starting & Warmup	-	100 pounds per square inch

Maximum Permissible Oil Temperature. The maximum permissible oil inlet temperature is 245°F. See Paragraph 19 for recommendations on operating conditions with light oils.

Exhaust Back Pressures: The exhaust back pressures shall not exceed 2 inches of mercury at any cylinder unless the exhaust system has been evaluated and approved by Avco Lycoming.

Cylinder Temperature. Maximum permissible cylinder head temperature, measured with bayonet type AN5541 thermocouple or bayonet type AN5546 resistance temperature bulb, shall be 500°F. Thermocouples, when supplied by Avco Lycoming, may be furnished with either -Y or -J calibration.

Maximum permissible cylinder base temperature measured at the lowest point on the leeward side of the cylinder in the fillet of the base flange is 325°F. Engines intended for prototype installation can be ordered with cylinder base thermocouples installed.

Magneto Temperature. Maximum permissible magneto temperature measured at a coil core shall not exceed 225°F. Engines intended for prototype installation can be ordered with magneto thermocouple installed.

Ignition Harness Temperature. Maximum permissible ignition harness temperature limits vary with the type of harness on the engine. Ignition harness temperature data shall be submitted to Avco Lycoming Installation for approval.

Alternator Temperature. Maximum permissible alternator temperature limits shall be as follows:

1.	Alternator Stator Slot	- 360°F
2.	Stator End Turns	- 360°F
3.	Drive End Bearing	- 248°F
4.	Positive Heat Sink	- 305°F

Each iron constantan thermocouple will be tagged with a number 1 through 4, corresponding to the locations listed above.

Engines intended for prototype installation should be ordered with thermocouples installed.

Temperature Correction. All temperatures shall be corrected in accordance with FAA factor.

Carburetor Air Intake. Carburetor air intake system shall be approved by the engine manufacturer. Provision shall be made for the use of hot air to the carburetor. When using hot air, a temperature rise of at least 90°F above an outside ambient air temperature of 30°F is required when the engine is developing 75% normal rated power.

Air Cleaner. The engine induction system shall incorporate a suitable air cleaner which must be approved by Avco Lycoming Division. Attention is drawn to the fact that operation under severe dust conditions will require additional air cleaner capacity and efficiency effective on both the hot and cold air supply to provide satisfactory engine life. These conditions may be encountered in aircraft operating from dusty fields or in dust storm areas.

Air cleaner air flow at sea level rated power is 1150 pounds of air per hour.

V. DETAIL WEIGHTS

1. ENGINE (STANDARD)

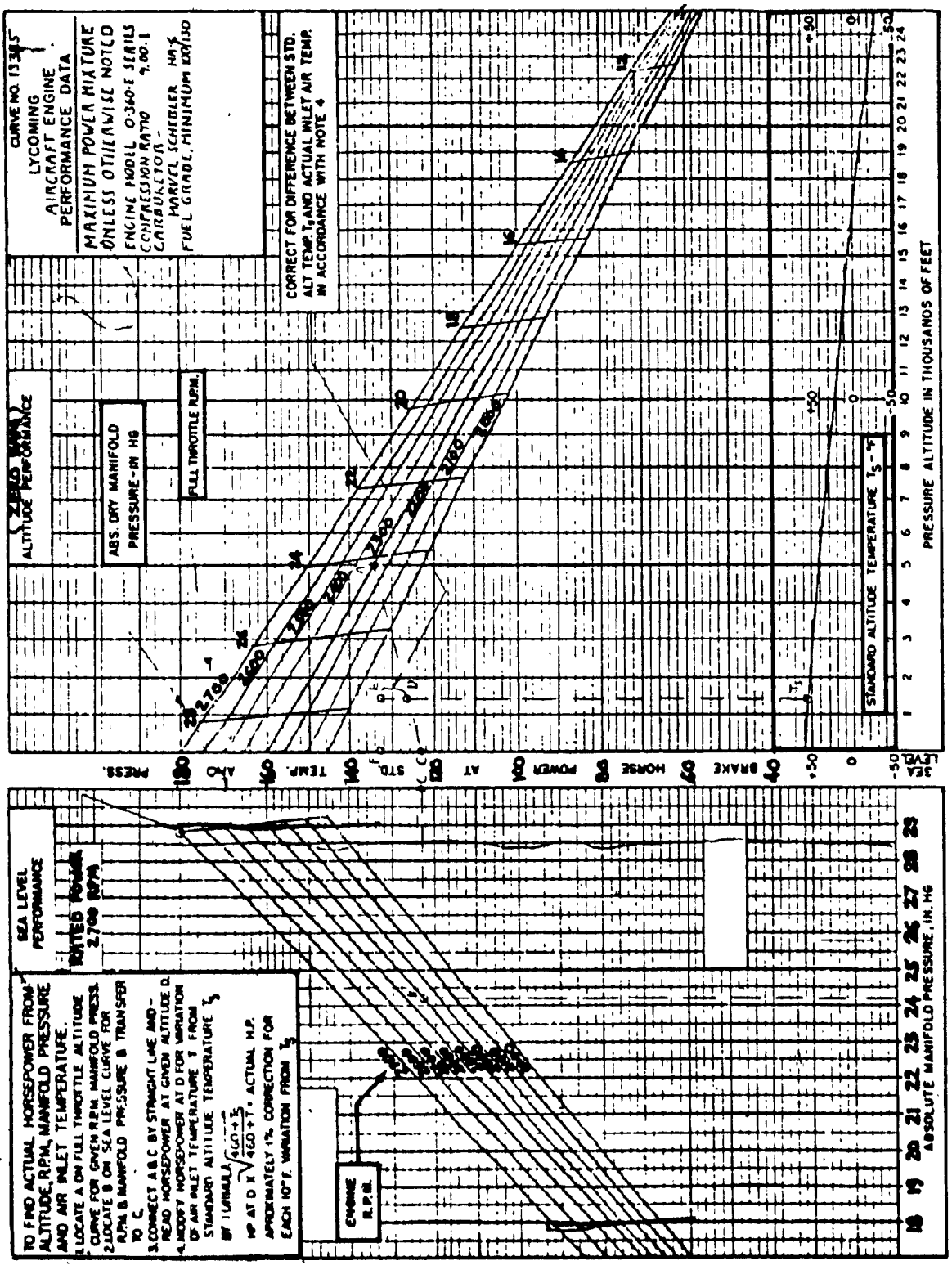
Basic Engine, Avco Lycoming O-360-E1A6D	239.28
Carburetor, Marvel Schebler Type HA-6	5.13
Ignition System, Bendix Magneto & Harness D4RN-2021 (Impulse Coupling)	11.35
Spark Plugs, Shielded	1.86
Oil Filter	1.60
Intercylinder Baffles	.67
Tachometer Drive	0.08
Starter and Alternator Drive	6.53
Starter, Prestolite, Geared, With Bendix Unit, 3.38:1	18.00
Alternator, Prestolite, 12 Volt, 60 Amp With Mounting Bracket	13.00
Dynafocal Mounting Bracket (Type 1)	2.50
STANDARD ENGINE DRY WEIGHT	300.00

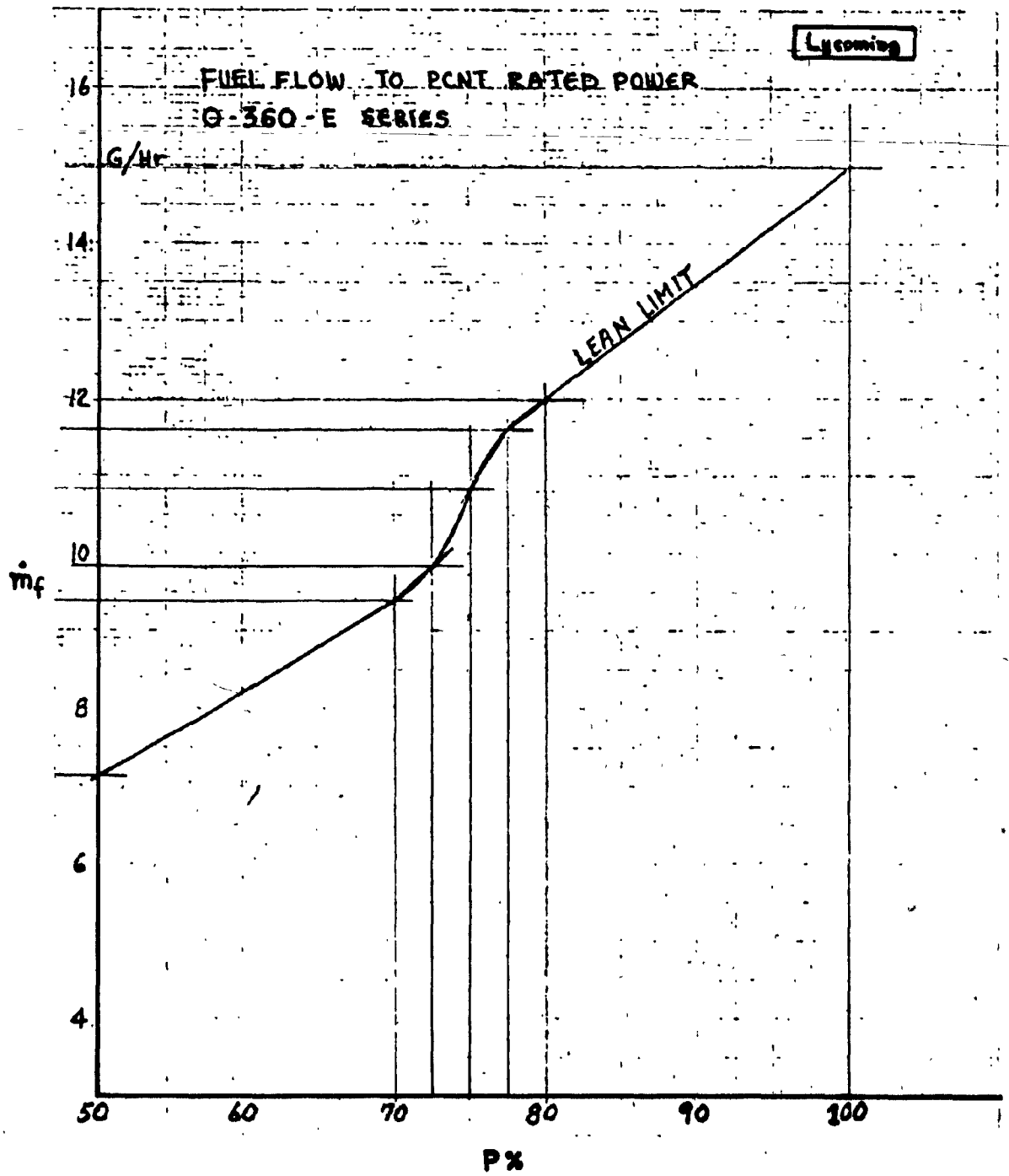
2. ENGINE INSTALLATION PARTS (STANDARD)

Starter Switch, Magnetic	1.16
Transistor Voltage Regulator	1.25
Overvoltage Relay	.85
WEIGHT OF INSTALLATION PARTS	3.26

3. ACCESSORIES, DRIVES AND PARTS (OPTIONAL)

Fuel Pump, AC Type	1.84
AN Propeller Governor Drive	1.50
Alternator, Prestolite, 12 Volt, 70 Amp With Bracket	13.00
24 Volt, 70 Amp With Bracket	13.00
Starter, Prestolite, Geared, 3.38:1, 24 Volt	18.00
Primer System	.40
Exhaust Flanges	.66
Cylinder Base Thermocouples With AN Terminals	.11
Cylinder Head Thermocouples, Bayonet Type	.39
Magneto Thermocouple	.03
Dual Drive, Vacuum Pump-Hydraulic Pump	4.50
Alternator Brackets for Ford Alternator	2.00
Dynafocal Mounting Brackets (Type 2)	2.50


CURVE NO. 1345
LYCOMING
AIRCRAFT ENGINE
PERFORMANCE DATA
MAXIMUM POWER MIXTURE
UNLESS OTHERWISE NOTED
ENGINE MODEL O-360-F SERIES
COMPRESSION RATIO 8.60:1
CARBURETOR -
MARVEL SCHEBLER MA-5
FUEL GRADE, MINIMUM 100/130
CORRECT FOR DIFFERENCE BETWEEN STD. ALT TEMP. T, AND ACTUAL INLET AIR TEMP. IN ACCORDANCE WITH NOTE 4
ZERO RAM
ALTITUDE PERFORMANCE
ABS. DRY MANIFOLD PRESSURE - IN HG
FULL THROTTLE R.P.M.
SEA LEVEL PERFORMANCE
RATED POWER 2700 R.P.M.
TO FIND ACTUAL HORSEPOWER FROM ALTITUDE, R.P.M., MANIFOLD PRESSURE AND AIR INLET TEMPERATURE.
1. LOCATE A ON FULL THROTTLE ALTITUDE CURVE FOR GIVEN R.P.M. MANIFOLD PRESS.
2. LOCATE B ON SEA LEVEL CURVE FOR R.P.M. & MANIFOLD PRESSURE & TRANSFER TO C.
3. CONNECT A & C BY STRAIGHT LINE AND READ HORSEPOWER AT GIVEN ALTITUDE D.
4. MODIFY HORSEPOWER AT D FOR VARIATION OF AIR INLET TEMPERATURE T FROM STANDARD ALTITUDE TEMPERATURE T_S BY FORMULA $\frac{460+T_S}{460+T}$
HP AT D X $\frac{460+T_S}{460+T}$ = ACTUAL H.P.
APPROXIMATELY 1% CORRECTION FOR EACH 10°F. VARIATION FROM T_S
ENGINE R.P.M.
ABSOLUTE MANIFOLD PRESSURE, IN. HG
PRESS. ALT.
STD. AIR TEMP.
BRAKE HORSE POWER AT
STANDARD ALTITUDE TEMPERATURE T_S - °F
PRESSURE ALTITUDE IN THOUSANDS OF FEET

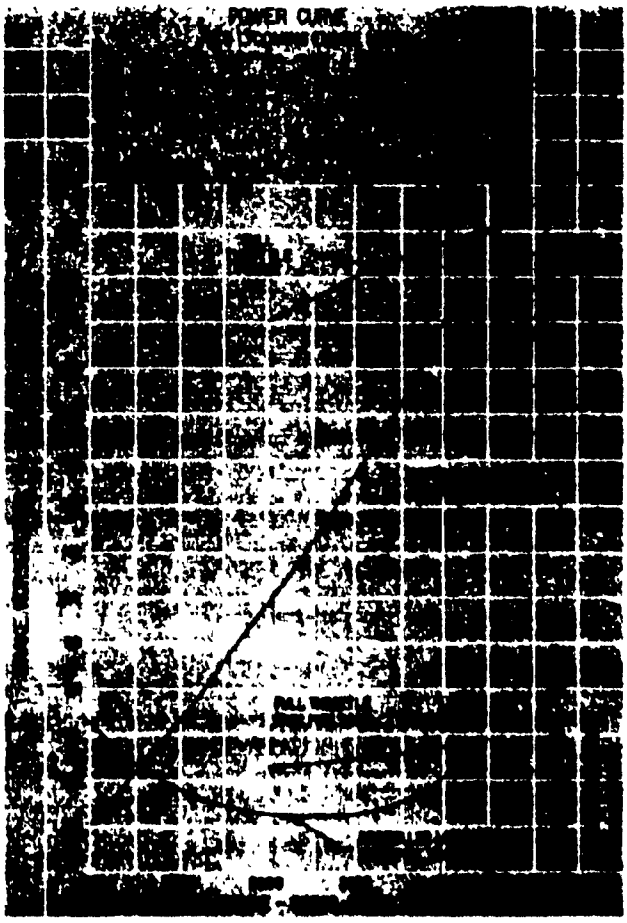

Lycoming
FUEL FLOW TO PCNT RATED POWER
O-360-E SERIES
G/Hr
16
14
12
10
8
6
4
LEAN LIMIT
$\dot{m}_f$
50
60
70
80
90
100
P%

AVCO LYCOMING

O-360 and ASSOCIATED MODELS

OPERATING CONDITIONS

Operation	RPM	HP	Fuel Cons. Gal./Hr.	Max. Oil Cons. Qts./Hr.	*Max. Cyl. Head Temp.
			O-360-A,-C** Series		
Normal Rated	2700	180	—	.80	500°F. (260°C.)
Performance Cruise (75% Rated)	2450	135	10.5	.45	500°F. (260°C.)
Economy Cruise (65% Rated)	2350	117	9.0	.39	500°F. (260°C.)

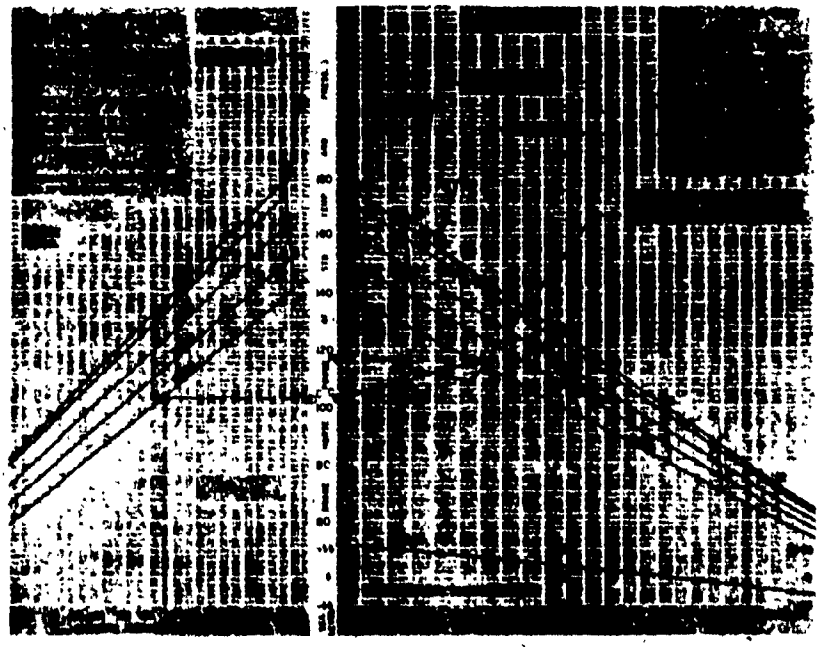


AVCO LYCOMING

O-360 and ASSOCIATED MODELS

FULL THROTTLE HP AT ALTITUDE
(Normally Aspirated Engines)

Altitude Ft.	% S. L. H. P.	Altitude Ft.	% S. L. H. P.	Altitude Ft.	% S. L. H. P.
0	100	10,000	70.8	19,500	49.1
500	98.5	11,000	68.3	20,000	48.0
1,000	96.8	12,000	65.8	20,500	47.6
2,000	93.6	13,000	63.4	21,000	46.0
2,500	92.0	14,000	61.0	21,500	45.2
3,000	90.5	15,000	58.7	22,000	44.0
4,000	87.5	16,000	56.5	22,500	43.3
5,000	84.6	17,000	54.3	23,000	42.2
6,000	81.7	17,500	53.1	23,500	41.4
7,000	78.9	18,000	52.1	24,000	40.3
8,000	76.2	18,500	51.4	24,500	39.5
9,000	73.5	19,000	50.0	15,000	38.5



Appendix C

Aeroengine Specifications

Keenan, J. H., F. G. Keyes, P. G. Hill, and J. G. Moore, "Steam Tables," Wiley, New York,

Temp. °C T	Press. bars P	Specific volume: Sat. liquid v_f	Specific volume: Sat. vapor v_g	Internal energy: Sat liquid u_f	Internal energy: Sat. vapor u_g	Enthalpy: Sat. liquid h_f	Enthalpy: Evap. h_{fg}	Enthalpy: Sat. vapor h_g	Entropy: Sat. liquid s_f	Entropy: Sat vapor s_g
0	.00611	1.0002	206278	−.03	2375.4	−.02	2501.4	2501.3	−.0001	9.1565
5	.00872	1.0001	147120	20.97	2382.3	20.98	2489.6	2510.6	.0761	9.0257
10	.01228	1.0004	106379	42.00	2389.2	42.01	2477.7	2519.8	.1510	8.9008
15	.01705	1.0009	77926	62.99	2396.1	62.99	2465.9	2528.9	.2245	8.7814
20	.02339	1.0018	57791	83.95	2402.9	83.96	2454.1	2538.1	.2966	8.6672
25	.03169	1.0029	43360	104.88	2409.8	104.89	2442.3	2547.2	.3674	8.5580
30	.04246	1.0043	32894	125.78	2416.6	125.79	2430.5	2556.3	.4369	8.4533
35	.05628	1.0060	25216	146.67	2423.4	146.68	2418.6	2565.3	.5053	8.3531
40	.07384	1.0078	19523	167.56	2430.1	167.57	2406.7	2574.3	.5725	8.2570
45	.09593	1.0099	15258	188.44	2436.8	188.45	2394.8	2583.2	.6387	8.1648
50	.1235	1.0121	12032	209.32	2443.5	209.33	2382.7	2592.1	.7038	8.0763
55	.1576	1.0146	9568	230.21	2450.1	230.23	2370.7	2600.9	.7679	7.9913
60	.1994	1.0172	7671	251.11	2456.6	251.13	2358.5	2609.6	.8312	7.9096
65	.2503	1.0199	6197	272.02	2463.1	272.06	2346.2	2618.3	.8935	7.8310
70	.3119	1.0228	5042	292.95	2469.6	292.98	2333.8	2626.8	.9549	7.7553
75	.3858	1.0259	4131	313.90	2475.9	313.93	2321.4	2635.3	1.0155	7.6824
80	.4739	1.0291	3407	334.86	2482.2	334.91	2308.8	2643.7	1.0753	7.6122
85	.5783	1.0325	2828	355.84	2488.4	355.90	2296.0	2651.9	1.1343	7.5445
90	.7014	1.0360	2361	376.85	2494.5	376.92	2283.2	2660.1	1.1925	7.4791
95	.8455	1.0397	1982	397.88	2500.6	397.96	2270.2	2668.1	1.2500	7.4159
100	1.014	1.0435	1673.	418.94	2506.5	419.04	2257.0	2676.1	1.3069	7.3549
110	1.433	1.0516	1210.	461.14	2518.1	461.30	2230.2	2691.5	1.4185	7.2387
120	1.985	1.0603	891.9	503.50	2529.3	503.71	2202.6	2706.3	1.5276	7.1296
130	2.701	1.0697	668.5	546.02	2539.9	546.31	2174.2	2720.5	1.6344	7.0269
140	3.613	1.0797	508.9	588.74	2550.0	589.13	2144.7	2733.9	1.7391	6.9299

v, cm^3/g; u, kJ kg; h, kJ kg; s, kJ (kg)(K)

Fig. C.1 Icing steam table function data.

APPENDIX D

THE 8086 MICROPROCESSOR.

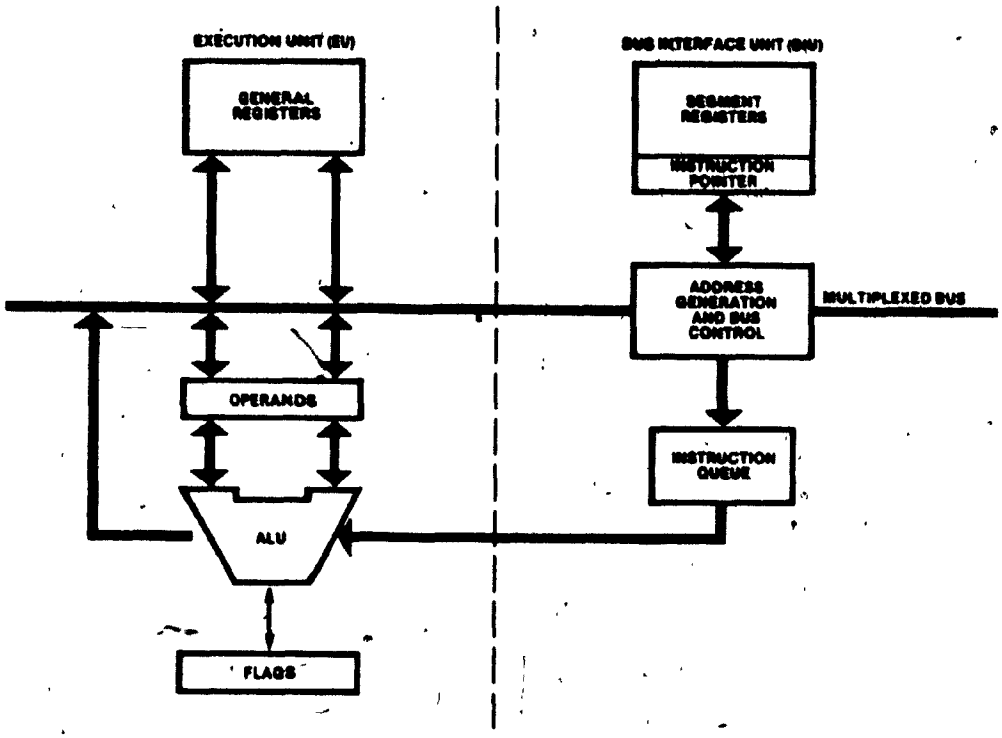
The 8086 is a third generation, 16 bit microprocessor housed in a 40 pin DIP; it is suitable for a very wide range of applications from single processor minimal-memory designs, to multiprocessor systems of one megabyte of memory. Pipelined architecture is combined with a 16 bit internal data path in this μPU to realise a high level of performance since instructions can be prefetched during spare bus cycles, more of them due to their compact format. It is designed to provide direct hardware support for software written in high level languages therefore, assembly programming is not a must. Its symmetrical instruction set supports direct operation on memory operands, including operands on the stack. The hardware addressing modes provide efficient, straightforward implementations of based variables, arrays, arrays of structures and other high level language data constructs. Memory to memory string operations are available for efficient character format data operations.

The 8086 microprocessor consists of two basic building blocks found in most Intel designs; the Execution unit and the Bus Interface unit. (Fig. D.1) The first one executes instructions; the other, fetches instructions, reads operands and writes results; the two units can operate concurrently and independant of each other, hence instruction fetch and execution cycles overlap extensively. (Fig. D.1). The execution unit of the 8086 processor has no connections to the system bus, it simply obtains instructions from a queue maintained by the Bus Interface unit; similarly it may request access to memory or peripheral devices. All addresses manipulated are 16 bit wide but the Bus Interface unit performs address relocation giving the Execution unit access to one megabyte of memory space.

The Bus Interface unit of the 8086 μPU, 'looks ahead' when the Execution unit is busy and fetches more instructions from programme memory; these are stored in the stream queue until they are requested for execution. A set of eight 16 bit data, index and pointer registers with no particular accumulator, four segment registers, an instruction pointer, and nine flags constitute the main programming facilities of the microprocessor. Dynamically relocatable code, segmentation and memory implemented stacks are other important features present in this family of μPUs. A hardware specification view of the computer system is presented in the next pages.

Appendix D

The iAPX 86 μPU system.



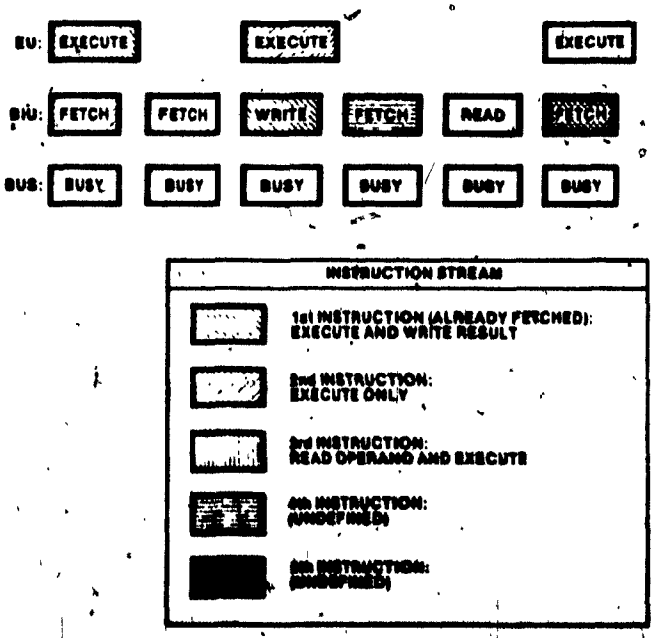


Fig. D.1. Architecture of the 8086 μPU.

iSBC 86/12A

SPECIFICATIONS

WORD SIZE	
Instruction	8, 16, or 32 bits
Data	8/16 bits
INSTRUCTION CYCLE TIME	400 nanoseconds for fastest executable instruction (assumes instruction is in the queue) 1.0 microseconds for fastest executable instruction (assumes instruction is not in the queue)
MEMORY CAPACITY	
On-board ROM/EPROM	Up to 16K bytes; user installed in 2K, 4K or 8K byte increments or up to 32K bytes if iSBC 340 Multimodule EPROM option installed.
On-board Dynamic RAM	32K bytes or 64K bytes if iSBC 300 Multimodule RAM option installed. Integrity maintained during power failure with user-furnished batteries.
Off-board Expansion	Up to 1 megabyte of user-specified combination of RAM, ROM, and EPROM.
MEMORY ADDRESSING	
On-board ROM/EPROM	FF000–FFFFF$_H$ (using 2758 EPROM's). FE000–FFFFF$_H$ (using 2316E ROM's or 2716 EPROM's). FC000–FFFFF$_H$ (using 2332A ROM's or 2732 EPROM's), and F8000–FFFFF$_H$ (if iSBC 340 Multimodule EPROM option installed)
On-board RAM (CPU Access)	00000–07FFF$_H$. 00000–0FFFF$_H$ (if iSBC 300 Multimodule RAM option installed).
On-Board RAM (Multibus Interface Access)	Jumpers and switches allow board to act as slave RAM device for access by another bus master. Addresses may be set within any 8K boundary of any 128K segment of the 1-megabyte system address space. Access is selectable for 8K, 16K, 24K, or 32K bytes.
Slave RAM Access	Average; 1.5 microseconds
SERIAL COMMUNICATIONS	
Synchronous:	5-, 6-, 7-, or 8-bit characters. Internal; 1 or 2 sync characters. Automatic sync insertion.
Asynchronous:	5-, 6-, 7-, or 8-bit characters. Break character generation. 1, 1½, or 2 stop bits. False start bit detection.

Sample Baud Rate:

Frequency[1] (kHz, Software Selectable)	Baud Rate (Hz)[2]		
	Synchronous	Asynchronous	
		÷16	÷64
153.6	—	9600	2400
76.8	—	4800	1200
38.4	38400	2400	600
19.2	19200	1200	300
9.6	9600	600	150
4.8	4800	300	75
2.4	2400	150	—
1.76	1760	110	—

Notes

1 Frequency selected by I/O writes of appropriate 16-bit frequency factor to Baud Rate Register

2 Baud rates shown here are only a sample subset of possible software-programmable rates available Any frequency from 18.75 Hz to 613.5 kHz may be generated utilizing on-board crystal oscillator and 16-bit Programmable Interval Timer (used here as frequency divider)

INTERVAL TIMER AND BAUD RATE GENERATOR

Input Frequency (selectable)

2.46 MHz ±0.1% (0.41 μsec period nominal),
1.23 MHz ±0.1% (0.82 μsec period nominal), and
153.6 kHz ±0.1% (6.5 μsec period nominal)

Output Frequencies

Function	Single Timer		Dual Timers (Two Timers Cascaded)	
	Min	Max	Min.	Max
Real-Time Interrupt, Interval	1.63 μsec	427.1 msec	3.26 μsec	466.5 minutes
Rate Generator (Frequency)	2.342 Hz	613.5 kHz	0.000036 Hz	306.8 kHz

POWER REQUIREMENTS (MAXIMUM).

CONFIGURATION	V_{CC} = +5V±5%	V_{DD} = +12V±5%	V_{BB} = -5V±5%	V_{AA} = -12V±5%
Without EPROM[1]	5.2A	350 mA	—	40 mA
RAM Only[2]	390 mA	40 mA	1.0 mA	—
With iSBC 530[3]	5.2A	450 mA	—	40 mA
With 4K EPROM[4] (Using 2758)	5.5A	350 mA	—	40 mA

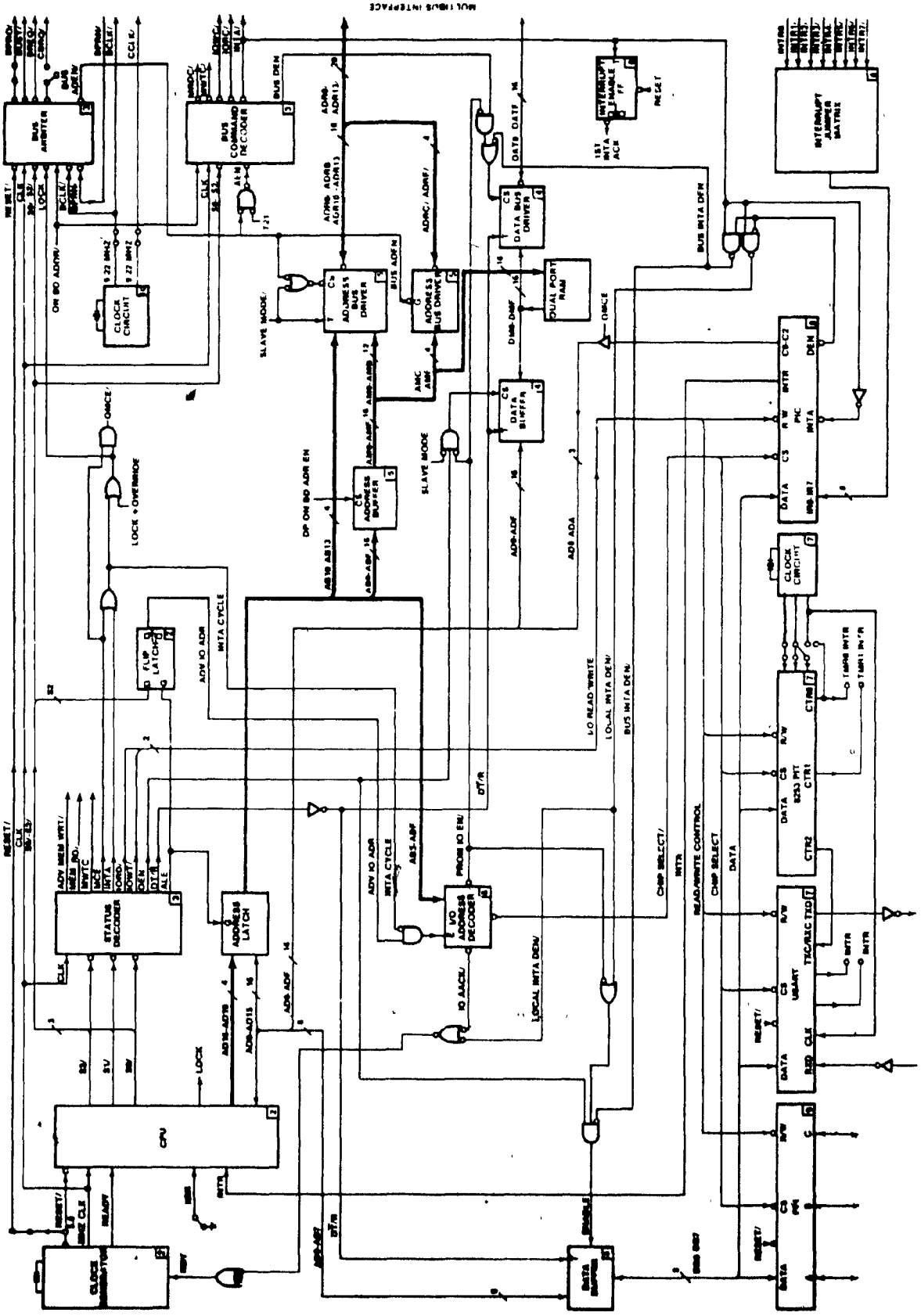
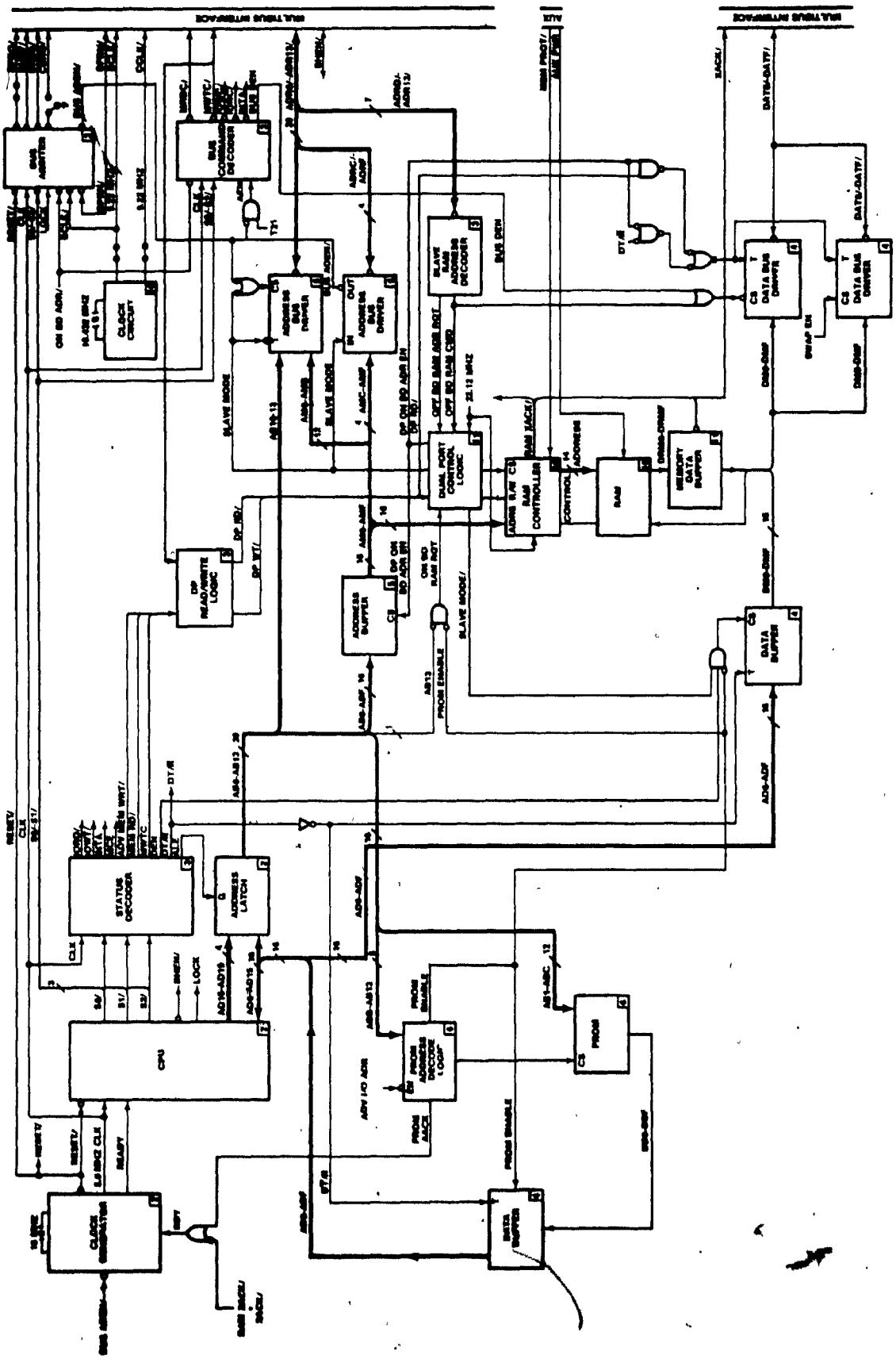


Fig. D.2. The 8612A SBC system design.

Appendix D

The iAPX 86 PU system.



APPENDIX E.

ANALOG DEVICES

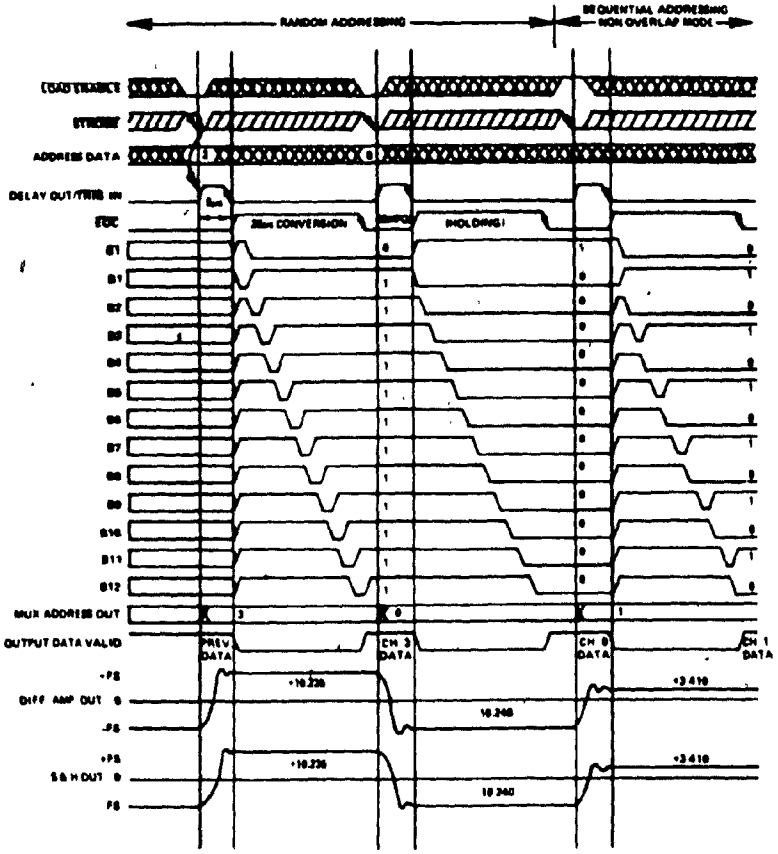
Low-Cost, High Speed Data Acquisition Module

DAS1128

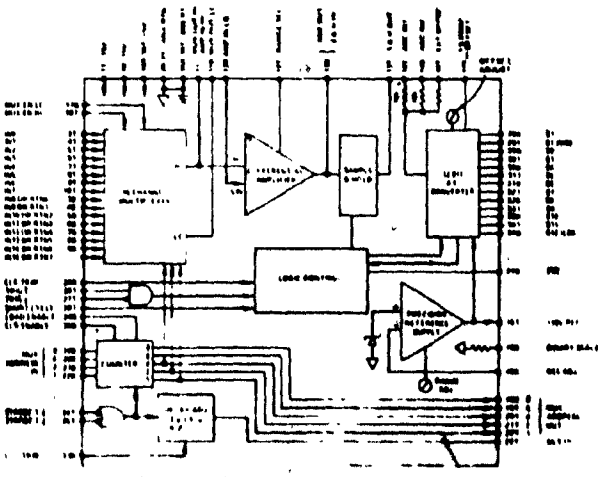
SPECIFICATIONS

ANALOG INPUTS	
Number of Inputs to Multiplexer	16 Single Ended, 8 True Differential 16 Pseudo Differential
Input Voltage (Full Scale Range)	±10V to +10V, 0V to +10V, 5V to +5V, 0V to +5V, ±0.24V to +10.24V, 0V to +10.24V, ±2.12V to +5.12V, or 0V to +5.12V
Maximum Input Voltage	±15V
Input Current (per channel)	5nA max
Input Impedance	$>10^{10}$ ohms
Input Capacitance	10pF for OFF channel 100pF for ON channel
Input Fault Current (power off or MUX failure)	Internally limited to 20mA
Direct ADC Input Impedance	10kΩ for each input line
ACCURACY[1]	
Resolution	12 Bits
Error Relative to F.S.	±½LSB
Quantization Error	±½LSB
Differential Nonlinearity Error	
@ 33kHz throughput rate	±½LSB, 1LSB max
@ 50kHz throughput rate	±1LSB
Noise Error	±½LSB
FS to +FS Error Between Successive Channel Transitions	±1LSB
TEMP. COEFFICIENTS	
Gain	8ppm/°C, 20ppm/°C max
Offset	5ppm/°C, 15ppm/°C max
Differential Nonlinearity	±.5ppm/°C, 6ppm/°C max
SIGNAL DYNAMICS	
Throughput Rate (12 Bits)	50kHz (max) (includes 5µs for MUX and SHA settling time plus 15µs for ADC)
MUX Crosstalk ("OFF" channels to "ON" channel)	>80dB down @ 1kHz
Differential Amplifier CMRR	70dB to 1kHz
SHA Acquisition Time to 0.01%	4.5µs max
SHA Aperture Uncertainty	10ns
SHA Feedthrough	70dB down @ 1kHz
DIGITAL INPUT SIGNALS	
Compatibility	Standard DTL/TTL logic levels, 1 unit load/line
MUX Address Inputs (8, 4, 2, 1, Pins 19B through 22B)	Positive true natural binary coding selects channel for random addressing mode. Must be stable for 100ns after STROBE
MUX ENABLE HI (Pin 18T)	High (Logic "1") input enables MUX "HI" output (for inputs 0 through 7)
MUX ENABLE LO (Pin 17B)	High (Logic "1") input enables MUX "LO" output (for inputs 8 through 15)
STROBE (Pin 24T or 25T)	Negative going transition (Logic "1" to Logic "0") updates MUX address register. STROBE 1 must be a Logic "1" to enable STROBE 2. STROBE 2 must be at Logic "1" to enable STROBE 1.
LOAD ENABLE (Pin 24B)	High (Logic "1") input allows next STROBE command to sequentially advance MUX address register. Low (Logic "0") input allows next STROBE command to update MUX address register according to external address inputs.
CLEAR ENABLE (Pin 25B)	Low (Logic "0") input allows next STROBE command to reset MUX address to channel "0" overriding LOAD ENABLE
TRIGGER (Pin 26T)	Positive going transition (Logic "0" to Logic "1") initiates A/D conversion (even during conversion). TRIGGER (Pin 27T) must be at Logic "0" to allow TRIGGER function.
TRIGGER (Pin 27T)	Negative going transition (Logic "1" to Logic "0") initiates A/D conversion. Pin 26T (TRIGGER) must be at Logic "1" to allow TRIGGER function.
DIGITAL OUTPUT SIGNALS	
Compatibility	Standard DTL/TTL logic levels, 5 unit loads/line
Parallel Outputs	BT, B1 through B12
Coding	Natural binary, two's complement, offset binary, or one's complement. Pin selectable
MUX Address Outputs (8, 4, 2, 1, pins 18B, 19T through 22T)	Positive true natural binary coding indicates channel selected
DELAY OUT (Pin 23T)	Negative going transition (Logic "1" to Logic "0") occurring nominally 5µs (adjustable from 3.0µs to 20µs) after STROBE command initiates A/D conversion automatically when connected to the TRIGGER
EOC (Pin 27B)	High (Logic "1") output during A/D conversion
ADJUSTMENTS & TRIMS	
Offset Adjust	
Internal Adjustment (Externally Accessible)	±1/2LSBs (min)
Remote External Adjustment (Pin 16T)	±1/2LSBs (min)
Range Adjust	
Internal Adjustment (Externally Accessible)	±1/2LSBs (min)
Remote External Adjustment (Pin 14B)	±1/2LSBs (min)
Clock Trim (Pin 26B)	
Factory Setting (Pin 26B OPEN)	1.25µs/Bit
External Adjustment Range	1.25µs/Bit to 2.00µs/Bit
Delay Trim (Pin 23B)	
Factory Setting (Pin 23B OPEN)	3.0µs
External Adjustment Range	3.0µs to 20µs
CONTROLS	
SHORT CYCLE (Pin 28T)	Connect to ground for full 12 bit resolution. Connect to B_8 output for resolution to B_{8-1} bits
Channel Selection Mode (MUX Address Loading Mode)	Random, sequential continuous, and sequential triggered. Pin selectable
A/D Conversion/Channel Select Sequences	Normal (input channel remains selected during its A/D conversion) and overlap (next channel selected during A/D conversion). Pin selectable
Range Select (Pin 12T)	Differential Amplifier gain control: connect to ANA RTN (Pin 2T) for X1 gain, connect to AMP OUT (Pin 13B) for X2 gain. This control is used in FSR selection procedure
BINARY SCALE (Pin 15B)	Connect to REF ADJ (Pin 16B) to set reference to 10.24V. This control is used in FSR selection procedure, see Table II
OUTPUT CODING (Pin 17T)	Ground for 1's complement output code, connect to -15V dc for other available codes
POWER REQUIREMENTS	
+15V ±3%	40mA, 50mA max
-15V ±3%	70mA, 100mA max
+5V ±5%	250mA, 300mA max
Power Supply Sensitivity[1]	
Gain	±2.0mV/V
Offset	±4.0mV/V
Ref	±0.5mV/V
ENVIRONMENT & PHYSICAL	
Operating Temperature	0 to +70°C
Storage Temperature	-25°C to +85°C
Relative Humidity	Up to 95% noncondensing
Electrical Shielding	RFI & EMI 6 sides (except connector area)
Packaging	Insulated steel coated module 3.00" x 4.00" x 0.375"

Timing Diagrams



Timing for Non-Overlap Operation in Both Random and Sequential Addressing Modes. For Status Keys and Signal Condition Data, Refer to Box Below.



APPENDIX F.

ENLODE INCORPORATED

8-CHANNEL DAC

SPECIFICATIONS:

RATING	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Logic Current	Icc	Vcc = 5.0		50	75	mA
Analog Supply	Vanlg	+ and - supply	12.0	15.0	15.5	Volts
+ Analog Current	I+	No outputs loaded		80	100	mA
-Analog Current	I-	No outputs loaded		65	80	mA
Output Load	Iout				10	mA

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	- 55°C to + 85°C
Ambient Temperature Under Bias	- 40°C to + 80°C
Logic Power Voltage	- 0.5 to + 7.0 Volts
Analog Power Voltage	± 16 Volts
Analog Output Current	± 50 mA

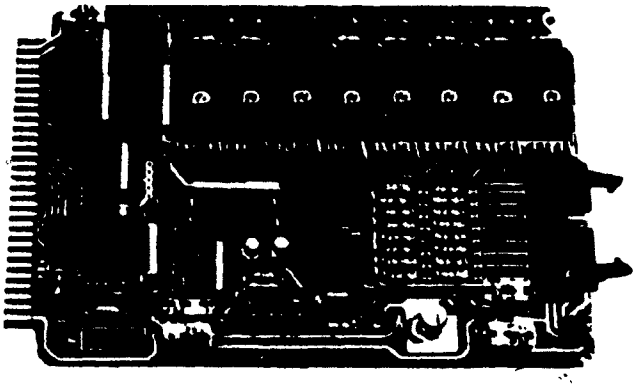
SPECIFIED OPERATING LIMITS

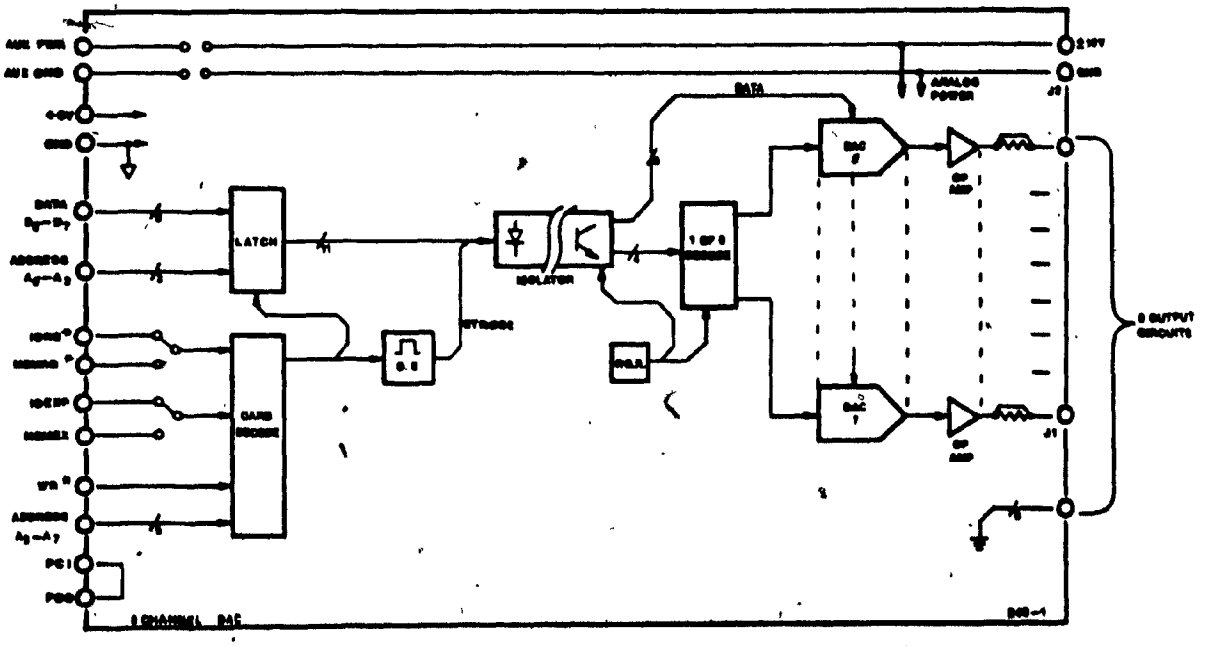
Ambient Temperature	0 to + 70°C
Logic Power Voltage	+ 4.5 to + 5.5 Volts
Analog Power Voltage	± 14.5 to ± 15.5 Volts*

The analog outputs are monotonic over the operating ambient temperature range. Output signal accuracy is within 3/4 LSB (0.3% full scale). Slew rate plus settling time to within 0.1% of final value of the analog outputs are four (4) microseconds per volt of commanded change plus 1.5 microseconds.

Appendix F

Digital to Analogue Converter





APPENDIX G.

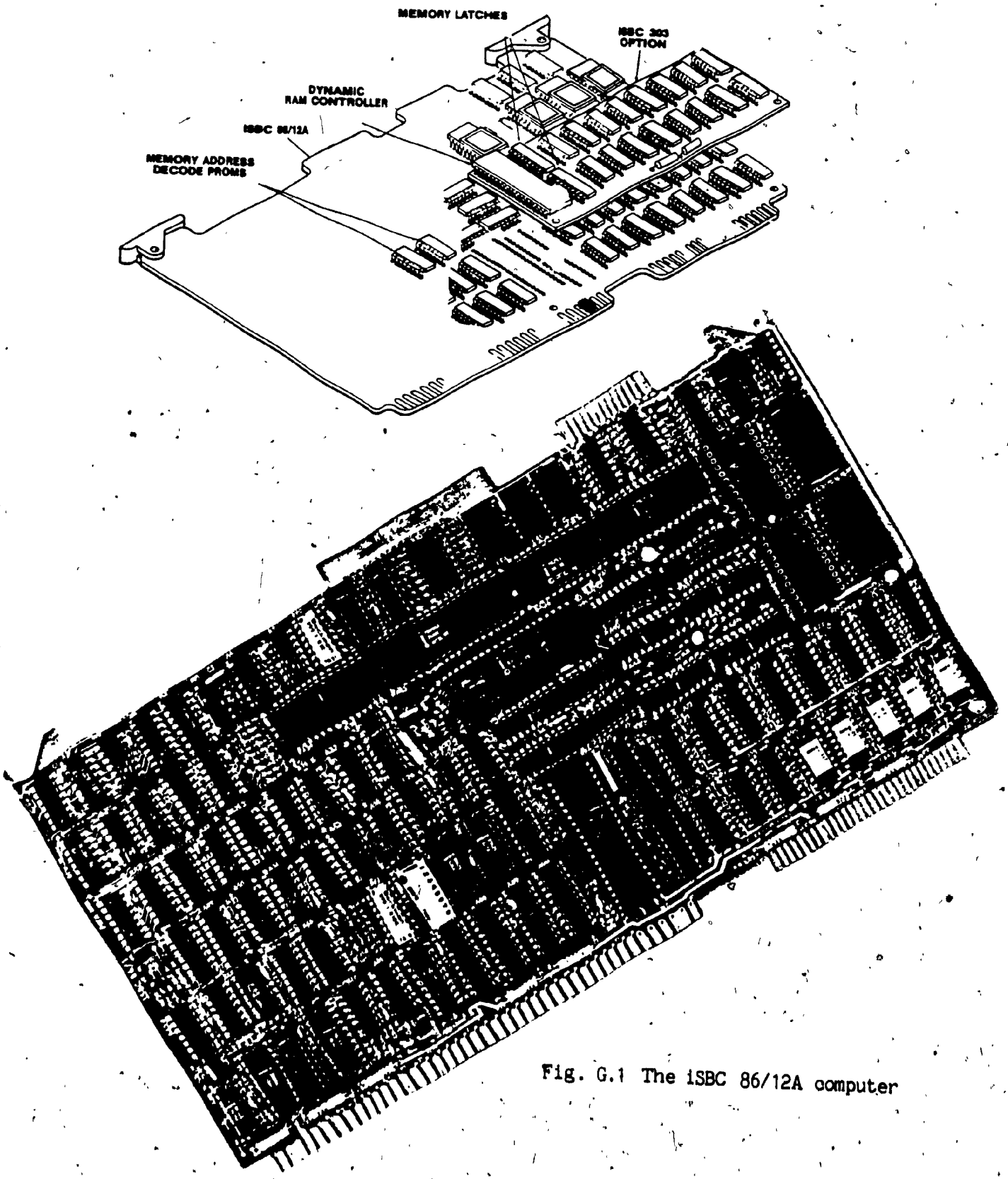


Fig. G.1 The iSBC 86/12A computer

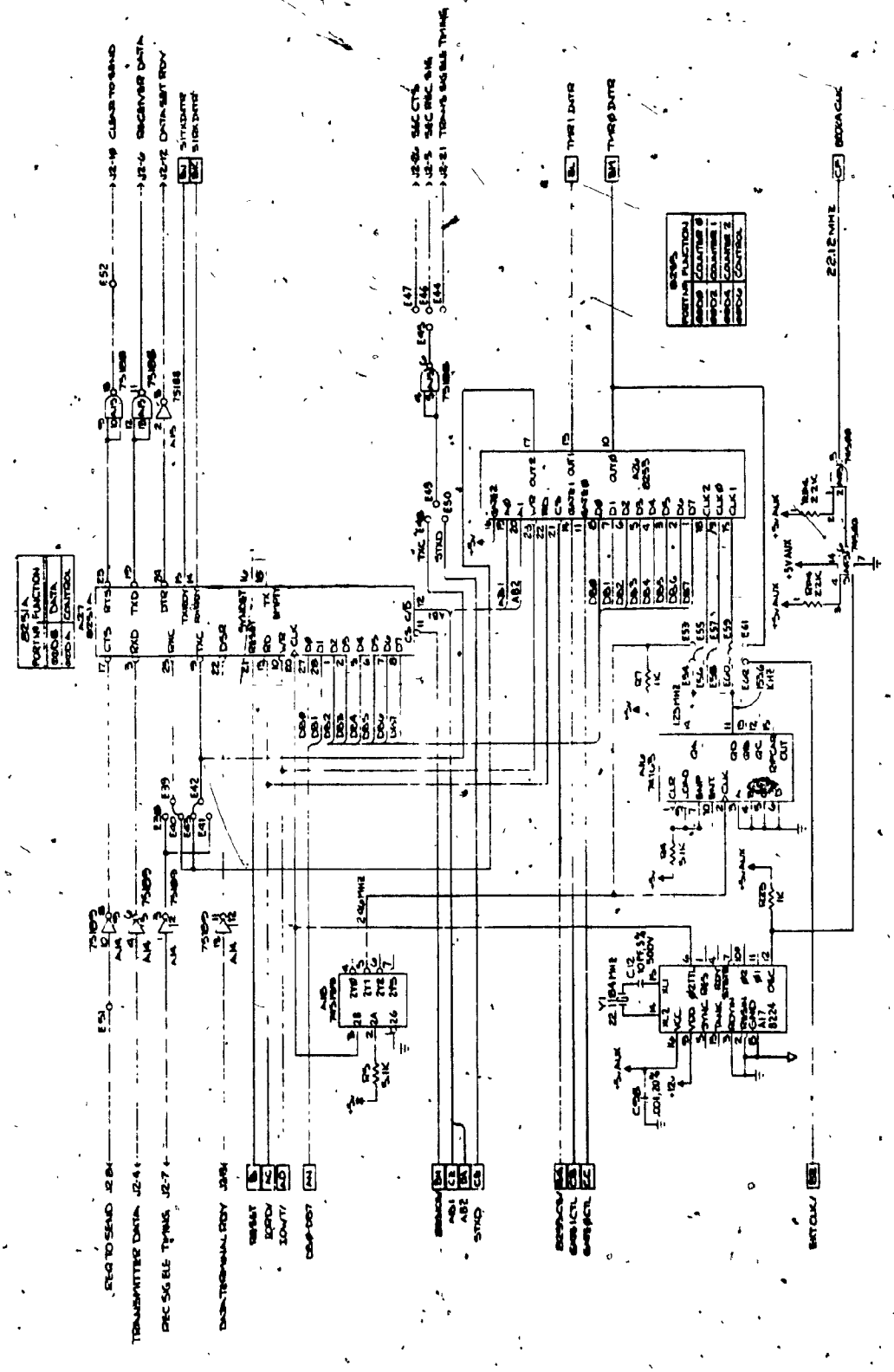


Fig. G.2 Serial Communications system.

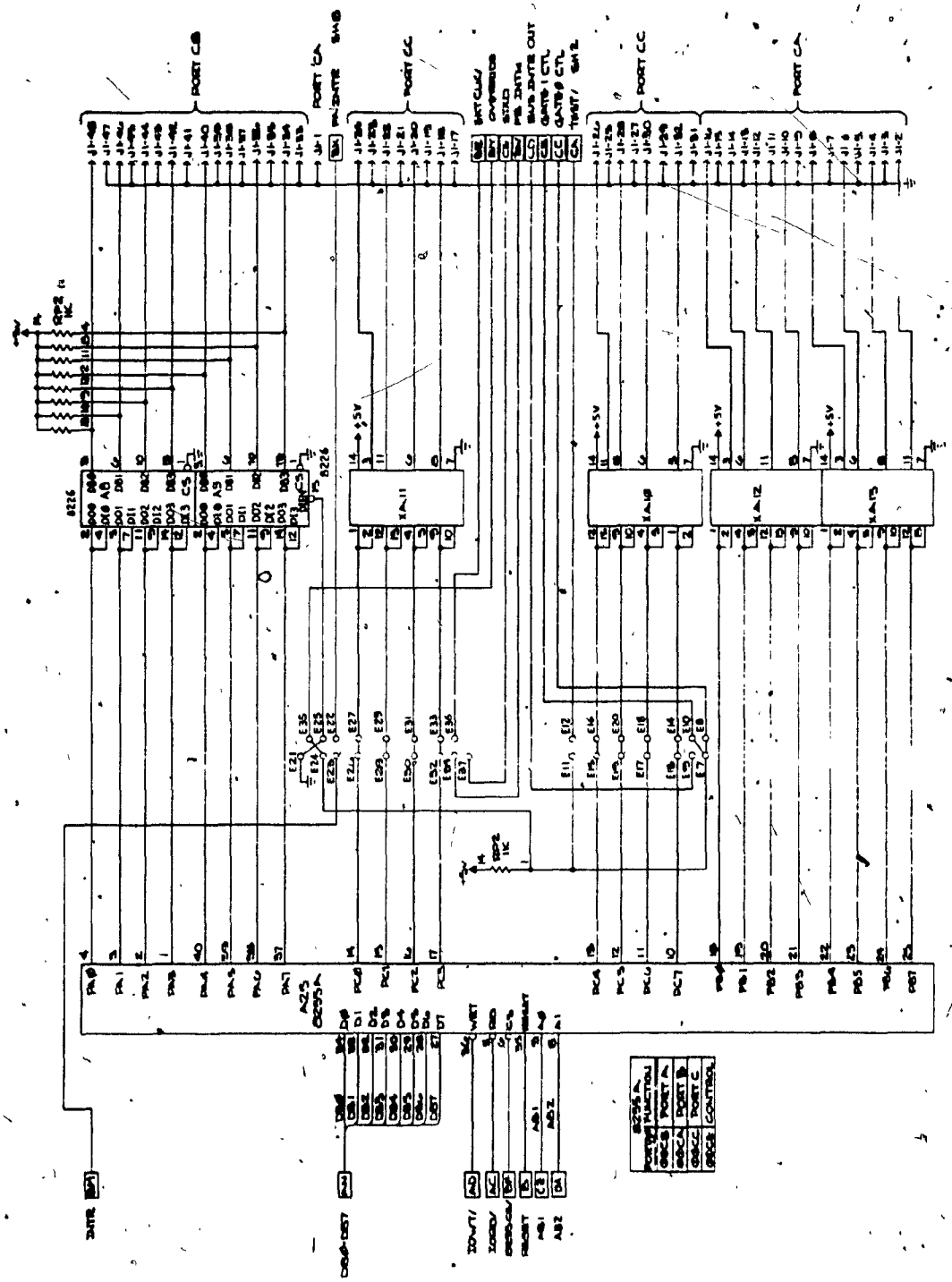


Fig. G.3 Parallel Communications system.

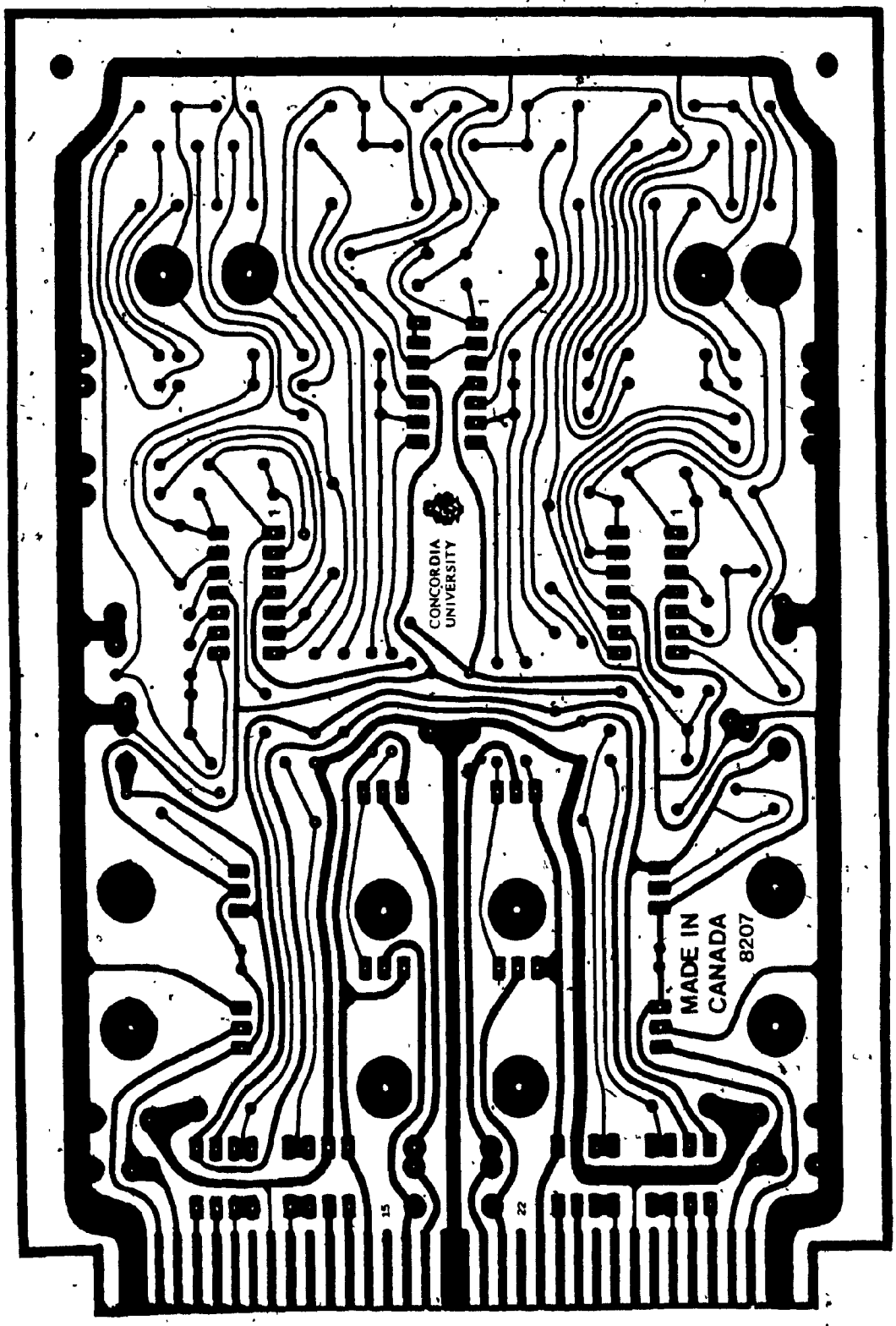


Fig. G.4 Servocontroller printed circuit.

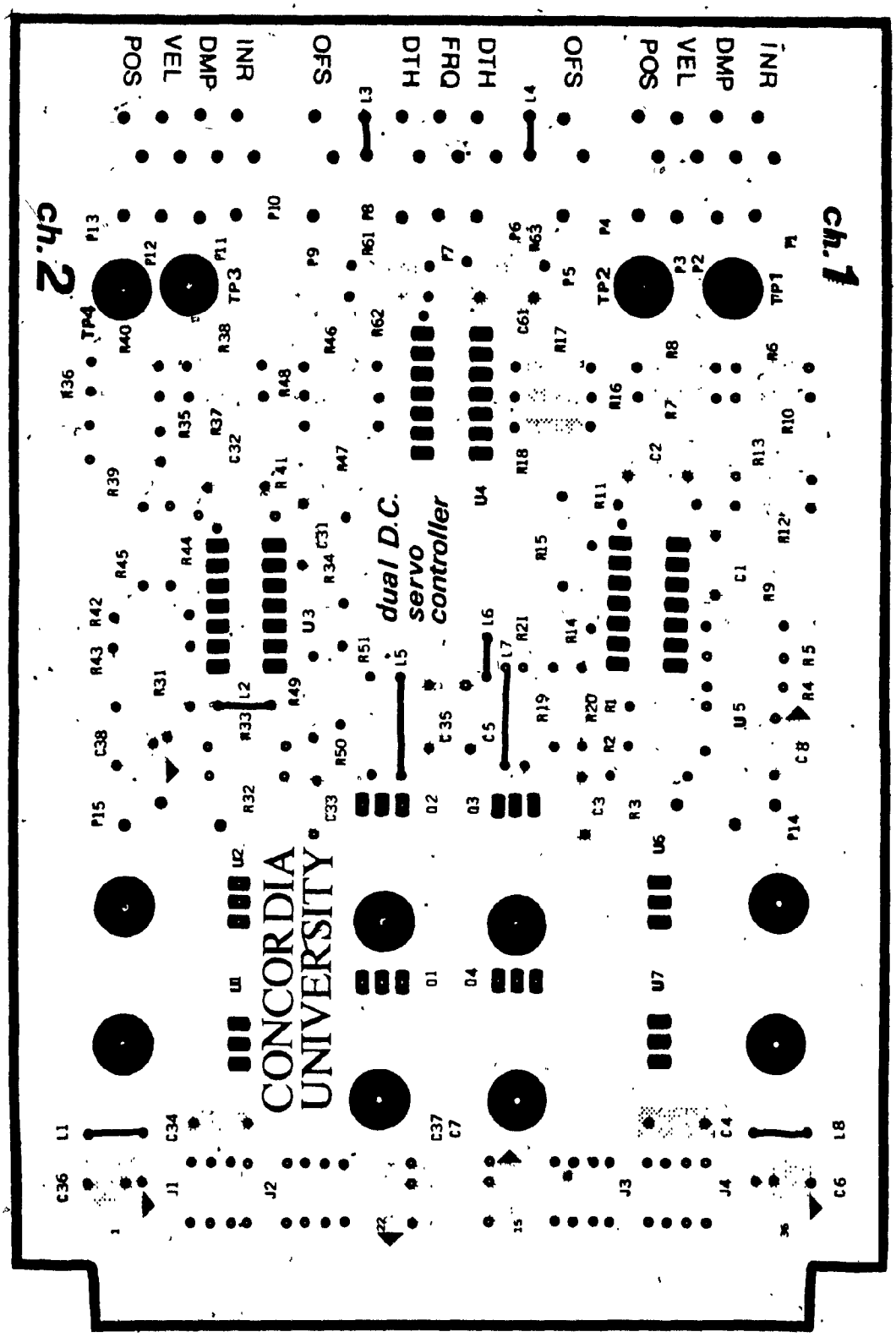


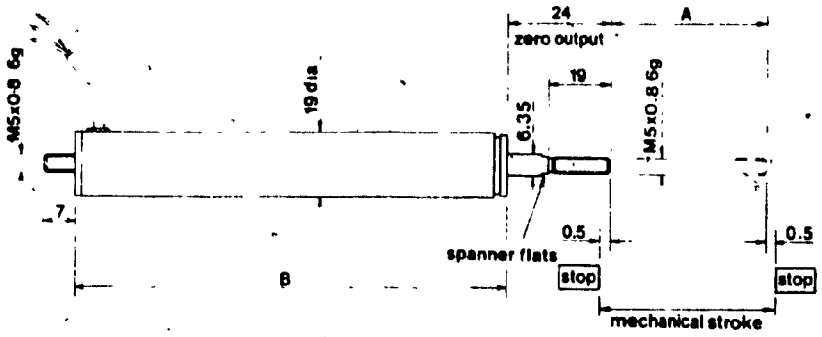
Fig. G.5 Servocontroller board component side.

Penny & Giles Controls Plc Ltd
Blackwood South Wales UK

Bendix
Aviation
Electric Ltd

rectilinear potentiometers specification

potentiometer model number LCPU



all dimensions in millimetres

electrical stroke length	A	25mm	50mm	100mm	150mm	200mm
body length	B	106.5	131.5	206.5	256.5	306.5
weight approx		190gm	200gm	220gm	230gm	240gm

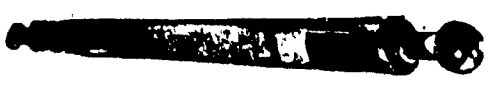


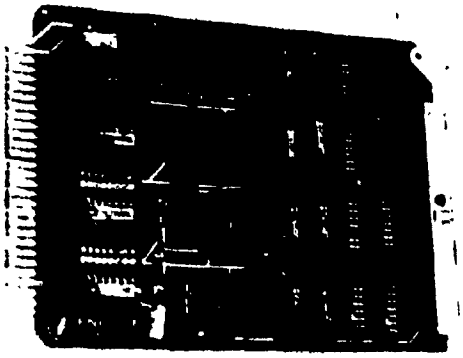
Fig. G.6 Transducer data.

ENLODE INCORPORATED

LATCHED OUTPUT DRIVER

SPECIFICATIONS:

PARAMETER	CONDITION	MIN	TYP	MAX	UNIT
V_{cc}		4.5	5.0	5.5	Volts
I_{cc}	V_{cc} = 5.0		40	64	mA
V_{out}	Output off			50	Volts
I_{out} Leakage	V_o = 50, T_A = 25°C			50	uA
I_{out} Leakage	V_o = 50, T_A = 70°C			100	uA
I_{out}	5 mS surge			600	mA
I_{out}*	1 or 2 outputs			400	mA
(derating)	3 outputs			300	mA
	4 outputs			240	mA
	5 outputs			210	mA
	6 outputs			175	mA
	7 outputs			150	mA
Response Time **	Resistive Load			2.5	uS



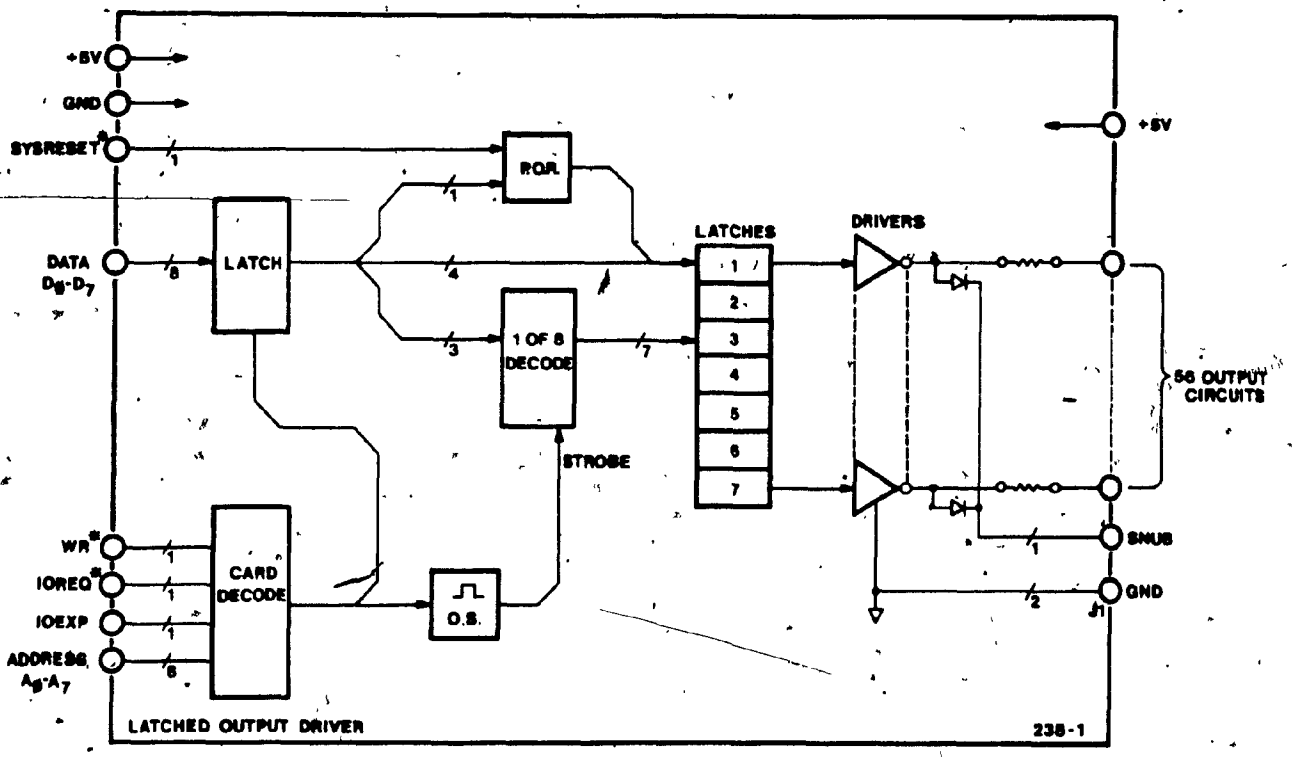


Fig. G.7 The Digital Data Output Circuit.