THE "TURBOCODE" SCHEME FOR THE PROGRAMMING OF THERMODYNAMIC CYCLE CALCULATIONS ON AN ELECTRONIC DIGITAL COMPUTER

by

J.R. Palmer
The "Turbocode" Scheme for the Programming of Thermodynamic Cycle Calculations on an Electronic Digital Computer

by


SUMMARY

The "Turbocode" Scheme for programming thermodynamic cycle calculations on an electronic computer is described in terms of the Ferranti "Pegasus" version. The structure and general mode of operation of the scheme is described in the main body of the Report, while details of the major program segments and their specifications, the method of programming and the method of running a Turbocode program are given in Appendices.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2. Basic Concepts</td>
<td>1</td>
</tr>
<tr>
<td>2.1 Requirements</td>
<td>1</td>
</tr>
<tr>
<td>2.2 Bricks and Station Vectors</td>
<td>2</td>
</tr>
<tr>
<td>2.3 Brick Data</td>
<td>4</td>
</tr>
<tr>
<td>2.4 Engine Vector</td>
<td>4</td>
</tr>
<tr>
<td>2.5 Other Types of Brick</td>
<td>4</td>
</tr>
<tr>
<td>3. The Master Program and Codewords</td>
<td>5</td>
</tr>
<tr>
<td>4. Assembly and Execution of Program</td>
<td>7</td>
</tr>
<tr>
<td>5. Conclusions</td>
<td>9</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>9</td>
</tr>
<tr>
<td>References</td>
<td>9</td>
</tr>
<tr>
<td>Appendix 1 Writing a Typical Master Program</td>
<td>10</td>
</tr>
<tr>
<td>Appendix 2 Notes on Functions, Subroutines and Bricks</td>
<td>14</td>
</tr>
<tr>
<td>Appendix 3 Abbreviated Brick Specifications</td>
<td>27</td>
</tr>
<tr>
<td>Appendix 4 Operating Instructions</td>
<td>41</td>
</tr>
<tr>
<td>Figures 1-9</td>
<td></td>
</tr>
</tbody>
</table>
1. Introduction

The operations involved in calculating the performance of a given thermodynamic cycle are of certain well-defined types, independent of the nature of the particular cycle investigated. Thus the application of an electronic digital computer to such calculations can be greatly facilitated by the provision of a scheme whereby pre-programmed operations can be assembled to form a complete program for analysing the cycle.

This report describes such a scheme, known as "Turbocode", which has been developed for this purpose in the Department of Aircraft Propulsion of the College of Aeronautics. While the scheme described here was developed for use with a Ferranti "Pegasus" computer, and so was written partly in the machine code and partly in the Autocode of that machine, its concepts and methods are readily applicable to other types of computer. An Algol 60 version is in the course of development, with particular reference to the College's I.C.T. 1905 computer, and this version will, subject to variations in hardware representation, be adaptable for any Algol-compatible machine with little extra effort. It is intended to supplement the present report with a report on the Algol version as soon as it has been fully developed and tested.

The scheme originated in "Scheme P", developed for the English Electric "Deuce" computer by Bristol Siddeley Engines Ltd. (ref. 1), whose assistance in the early stages is gratefully acknowledged, but it has since been considerably extended in scope.

2. Basic Concepts

2.1 Requirements

If we consider the calculation of the design-point performance of a simple turbojet in terms of a digital computer (see fig. 1), we find that portions of program are required capable of performing the following functions:-

(a) Data Input.
(b) Calculation of "unit processes" under the following headings:-
   (1) Intake;
   (2) Compressor;
   (3) Diffuser;
   (4) Combustion Chamber;
   (5) Turbine;
   (6) Jet Pipe; and
   (7) Propelling Nozzle.
(c) Calculation of "overall processes" involving linking the results of two or more unit processes:-
   (1) Compressor and Turbine Power Balance;
   (2) Compressor and Turbine Rotational Speed Balance;
   (3) Nozzle Exit Mach Number or Pressure Balance (for choked
and unchoked cases, respectively); and

(4) Calculation of Thrust, Specific Fuel Consumption, etc.

(d) Results Output.

2.2 Bricks and Station Vectors

In order to make it possible to link the various unit processes in a more or less arbitrary order, it is essential that the corresponding program segments, known as Bricks from their Deuce origins, should follow as closely as possible a standard format for data and results. Essentially, any process may be thought of as an operator which acts on a set of quantities defining the state of the gas at inlet to the process, thereby computing a corresponding set of quantities defining the state of the gas at outlet. These sets of quantities are known as Station Vectors, in the sense that a vector is an ordered set of numbers, and the unit processes are primarily concerned with evaluating such vectors at various reference planes or stations within the engine.

For air, or for the products of combustion of a given fuel in air, the gas state is completely defined by five quantities, thus:-

(1) fuel-air ratio;
(2) total or static pressure;
(3) total or static temperature, and
(4) and (5) any two of mass flow, velocity and flow area. (Note: If mass flow and area are specified, there are in general two possible velocities, one subsonic and the other supersonic, which are compatible with the given conditions, but the nature of the particular process will usually determine which solution is applicable).

Thus a minimal Station Vector would consist of just five elements. However, the particular set of five that is convenient or necessary is not the same in all contexts: consequently, a redundant system of eight elements has been adopted, as follows:-

(1) fuel-air ratio \( \alpha \) (non-dimensional)
(2) mass flow \( W \) (lb/s)
(3) static pressure \( p \) (lbf/in\(^2\) abs)
(4) total pressure \( P \) (lbf/in\(^2\) abs)
(5) static temperature \( t \) (°K)
(6) total temperature \( T \) (°K)
(7) velocity \( V \) (ft/s); and
(8) flow area \( A \) (ft\(^2\)).

Provided that sufficient information is given to define at least the minimal five elements, then the remaining three can always be found by using the familiar relations:-
(A) the Perfect Gas Equation of State

\[ p = \frac{Rpt}{144} \]

where \( R \) = gas constant (ft lbf/lb \( ^0 \)K) and \( \rho \) = density (lb/ft\(^3\));

(B) the Continuity Equation

\[ W = \rho AV; \]

(C) the Energy Equation

\[ H = h + \frac{V^2}{2g_c} J \]

where \( H = f_1(T, \alpha) \) and \( h = f_1(t, \alpha) \) are the total and static enthalpies respectively (C.H.U./lb);

(D) the Isentropic Equation

\[ \frac{P}{\rho} = \pi_T/\pi_t \]

where \( \pi_T = f_2(T, \alpha) \) and \( \pi_t = f_2(t, \alpha) \) are functions of the temperature-dependent entropy \( s^0(=\int \frac{Cpdt}{t})(\text{ft lbf/lb} \( ^0 \)K) \) given by

\[ \pi = \exp \frac{Js^0}{R}; \quad \text{and} \]

(E) the Mach Number Equation

\[ M = \frac{V}{\sqrt{g_c \gamma R t}} \]

where \( \gamma = f_3(t, \alpha) \) is the specific heat ratio (non-dimensional)

It will be noticed that "exact" thermodynamics is implied, in the sense that the specific heat at constant pressure, and the enthalpy, entropy function and specific heat ratio that depend on it, are functions of temperature and fuel-air ratio, but not of pressure. (In particular, dissociation effects are not allowed for).

In practice it is not always necessary or possible to compute the whole of a Station Vector: thus if neither the velocity nor the flow area is known, it is not possible to find the static pressure and temperature from given total values, or vice versa. To cater for this eventuality, we note that each of the eight Station Vector elements is essentially non-negative, and it is therefore convenient to denote any quantity which has not been calculated, or otherwise provided, by a suitable negative number. This is equivalent to a blank or dash in a written table of values, and permits a simple test for the presence or absence of any particular quantity.
2.3 Brick Data

Although the program for each Brick will define the corresponding thermodynamic process in principle, and although the relevant Inlet Station Vector will define the gas state at inlet to the process, this information is rarely sufficient. One or more items of the Outlet Station Vector may be needed (e.g. total temperature for a combustion chamber, or static pressure for a nozzle), but in addition certain items which are not part of any Station Vector may be required (e.g. adiabatic efficiency, pressure ratio, or pressure loss factor). Such information, qualitatively different from Station Vector material, is known as Brick Data: its nature and the order of occurrence of the elements cannot be generalised, since it must depend on the particular sequence of Bricks required for any given problem. It is therefore the responsibility of the Turbocode programmer to define the Brick Data list he requires in the course of coding his problem.

2.4 Engine Vector

Although the calculation of the Station Vectors throughout the cycle constitutes the major means whereby successive processes are linked together, and although knowledge of these items is valuable in itself, the linking of processes and the calculation of the "end products" - thrust, specific fuel consumption, etc - requires the generation of intermediate or final results which, like Brick Data, are different in kind from Station Vector material. Such items are collectively known as the Engine Vector which, like the Brick Data, is different in nature and layout for each engine cycle, and must be defined by the Turbocode programmer. On occasion, an Engine Vector result generated by one Brick may be required as data by another, so that provision is made for Engine Vector elements to be used as either input or output for a Brick. Furthermore, it is sometimes necessary to add or otherwise manipulate Engine Vector (or indeed Brick Data) elements, and a Special Brick (Brick 22) is available for this purpose: typical applications would be the adding of main and bypass gross thrusts, or of main and reheat fuel flows.

The relationship between a Brick and its associated Inlet and Outlet Station Vectors, Brick Data and Engine Vector is portrayed diagrammatically in fig. 2.

2.5 Other Types of Brick

The type of Brick described above is that most frequently needed, and indeed the requirements of the thermodynamic processes associated with such Bricks determines the whole pattern of the Turbocode Scheme. There is, however, a need for various non-thermodynamic Bricks, covering such functions as:

(a) Station Vector Input (Brick 16);
(b) Brick Data Input (Brick 15);
(c) Arithmetic on Engine Vector or Brick Data elements (Brick 22);
(d) Station Vector Output (Brick 31); and
(e) Engine Vector Output (Brick 32).
Evidently the detailed Structure and use of such a Brick depends on its particular function, and so will be described later when the various available Bricks are dealt with.

3. The Master Program and Codewords

In terms of the concepts just defined, we see that in order to program a particular problem the user is provided with a set of Standard Bricks. These must be linked together by a suitable Master Program to form a coherent whole, and in the course of writing this the user must define the elements of Brick Data, and of the Engine Vector, and their order of occurrence, and must then provide suitable Data Tapes giving Station Vector and Brick Data information to suit.

Since most Bricks follow a set pattern as regards their individual inputs and outputs, the Master Program can readily be formulated in terms of simple coded instructions, known as Codewords. The most general form of Codeword consists of seven elements, each consisting of an unsigned integer, separated from one another by commas, the whole Codeword being terminated by Carriage Return Line Feed (CRLF) thus:

\[
n, a, b, c, d, e, f \text{ CR LF}
\]

The elements of the Codeword are normally interpreted as follows:

- **n** = Brick Number, i.e. the reference number of the Brick which performs the required function. The system permits the use of Brick numbers in the range 0 to 63 inclusive, although by no means every number in this range has so far been allocated;

- **a** = Inlet Station Vector Number: the allocation of Station Vector Numbers, within the range 0 to 23 inclusive, is largely under the user's control, except that Station Vector 0 always refers to free stream (ambient) conditions, and that certain Bricks refer to more than one Outlet Station Vector, in which case all Outlet Station Vectors must be numbered in sequence, starting with:

- **b** = Outlet Station Vector Number;

- **c** = First Brick Data Item Number: any further Brick Data Items must follow in sequence. The Brick Data list for the whole program starts at item 0, and there is provision for items numbered up to 63;

- **d** = First Engine Vector Results Item Number: any further Engine Vector Results Items must follow in sequence. The Engine Vector list for the whole program starts at item 0, and there is provision for items numbered up to 31;

- **e** = First Engine Vector Data Item Number: this permits reference to a previously calculated Engine Vector Item as data. Certain Bricks require more than one Engine Vector Data Item; since it is not always possible to arrange for these items to be computed or
stored in strict sequence, a non-standard Codeword interpretation is used in these cases, whereby elements besides \( e \) may be used to define additional items (see, for example, Brick 33); and

\[ f = \text{Jump Codeword Number}. \] Normally the Codewords are obeyed in the order in which they are written in the Master Program. Occasionally, however, it is necessary to make a jump (conditional or unconditional) to a Codeword other than the next in sequence: the number of the Codeword concerned is therefore written as element \( f \). For this purpose the Codewords are numbered in sequence from 0, although it must be stressed that these Codeword numbers (which may range up to 87) are not part of the Codewords themselves, and must not appear on the Master Program Tape as punched.

In many cases, not every item of the Codeword is needed, in which case the corresponding element is written as zero. To allow as compact a Master Program as possible, and to reduce the risk of errors due to miscounting long strings of zeros, it is permissible to omit any zeros, with their preceding commas, that may occur at the right-hand end of the Codeword. For example

1. \( n, 0, 0, 0, 0, 0, 0 \) CRLF may be written as \( n \) CR LF
2. \( n, a, b, 0, d, 0, 0 \) CRLF may be written as \( n, a, b, 0, d \) CRLF

but

3. \( n, a, b, 0, 0, 0, f \) CRLF must be written in full, unless \( f \) happens to be zero, i.e. unless the Jump Codeword Number is zero, in which case \( n, a, b \) CRLF would suffice.

On the other hand, a few Bricks (e.g. Bricks 27 and 34) require more items to be specified than one Codeword can accommodate. For this purpose Brick 26 is provided: this has the effect of introducing a further Codeword which can be used to augment the Codeword immediately following, for the benefit of the Brick corresponding to this second Codeword.

The Master Program is normally prefixed by the standard Pegasus directives \( D \) (to punch the Date and Serial Number) and \( N \) (to punch the program Name which follows it, terminated by at least two blanks). Next comes the directive \( J200.0 \), which calls in the Codeword Input section of the Turbocode Scheme to read the Codewords which then follow. The last item in the Master Program is the sequence

\[ (\theta) \text{CRLF} \]

where \( \theta \) is an unsigned integer denoting the Codeword number (usually 0) at which the Master Program is to start operating once the Bricks have been read in. A typical Master Program is described in Appendix 1.
4. **Assembly and Execution of Program**

Although the user could copy the requisite Bricks onto the same tape as the Master Program, or could select and input the relevant Brick Tapes manually, this is obviously tedious, inconvenient, and a potential source of error. Consequently the whole process has been made automatic, its action being best described by following the input and obey procedures for the Master Program, Bricks and Data. A few explanatory notes are added for the benefit of those unfamiliar with the Pegasus computer, but this is not intended to be a full operator's guide, which is given in Appendix 4.

(a) The main Turbocode Scheme is input. This consists of five main sections:-

(i) the Codeword Input routine, which controls the reading and storage of Codewords;

(ii) the Assembly routine, which reads and stores the requisite Bricks;

(iii) the Codeword Obey routine, which interprets the codewords in the required sequence by calling up the relevant data, entering the corresponding Brick to perform the calculation, and storing the results;

(iv) a set of Subroutines which perform tasks common to a number of Bricks, such as: finding enthalpy change or isentropic pressure ratio for a given temperature change; finding outlet temperature for given inlet temperature and enthalpy change or isentropic pressure ratio; finding any one of mass flow, velocity or area from the other two; finding Mach Number from velocity, or vice versa; finding static conditions and area for given total conditions, mass flow and velocity; and finding static conditions and velocity (subsonic or supersonic as specified) for given total conditions, mass flow and area; and

(v) a number of Function routines, which calculate specific heat \( (C_p) \), enthalpy \( (h) \) and isentropic pressure ratio \( (\pi) \) as functions of temperature and fuel-air ratio, and also combustion fuel-medium ratio as a function of inlet temperature and fuel-air ratio and of outlet temperature. These Functions are used by various Bricks and Subroutines. The Functions and Subroutines are described in detail in Appendix 2.

(b) The Master Program is input. The directives D and N are read and obeyed by Initial Orders, as is the directive J 200.0 to transfer control to Codeword Input. The Codewords are then read, each being packed and stored as a single 39-bit word with suitable numbers of bits allocated to each of the seven elements of the Codeword. At the same time a list is built up of the Brick Numbers called for. When the terminal sequence "(0)" is read, the Entry Codeword Number is stored and control passes to the Assembly routine.

(c) The Bricks Tape is now scanned. Each Brick is preceded by its Number, which is read and compared with the list of Bricks required that was built up during Codeword input. If the particular Brick is needed, it is read in and stored, the corresponding Brick Number being deleted from the list, and
the Brick Number and starting block number of the Brick are punched on the Output Tape, but otherwise the Brick is rejected. As soon as all Bricks (which are stored end-to-end in the order in which they are read) have been input, the computer comes to a stop order without attempting to read the rest of the Bricks Tape. The Bricks are punched in decreasing order of frequency of use, in order to keep scanning time to a minimum. If, however, some Brick is called for which is not on the tape, the computer will come to a stop order in a different register. The operator can then identify the missing Brick by reference to the list of Brick Numbers punched out and, if the missing Brick is genuinely required and has not been called for by a punching error in the Codeword, can input a separate tape bearing this Brick. When the last Brick required has been input, its final block number is punched on the output tape, so that the operator can check that the available storage capacity has not been exceeded. (The last available block number is 858). Details of the various Bricks are given in Appendices 2 and 3.

(d) While the computer is stopped, the Data Tape is loaded, certain handswitches are set, and the computer restarted. Control now passes to Codeword Obey, which calls up the Entry Codeword (Number 0) specified by the sequence "(0)". The various elements of the Codeword are unpacked and used to address the various data and results items, and to identify the Brick required, which is then obeyed. The input of data occurs under the control of either Brick 15 (Station Vectors) or Brick 16 (Brick Data), while the output of results is controlled by Brick 31 (Station Vectors) or Brick 32 (Engine Vector). As each Brick completes its task, control reverts to Codeword Obey which then calls up the next Codeword in sequence or, if a jump has occurred, the Codeword whose Number is element f of the Codeword just obeyed. This process continues until such time as the data is exhausted, denoted in standard Pegasus fashion by the directive Z on the Data Tape, when a stop order is obeyed. At this point another Data Tape can be loaded and the computer restarted, if desired.

Since various Bricks require to know whether or not particular Station Vector Items have been provided, either as data or as the result of previous calculation, it is essential to restore each Station Vector to its initial state (consisting of "blanks" - actually negative numbers, as described in 2.2 above - or values which have been read as data) after printing out the results for one data point, and before reading fresh data. Also, it is desirable that the amount of fresh data should be minimised, all data items being assumed to be the same as for the previous point unless explicitly altered. To meet these requirements, a second copy is kept of all Station Vectors, in which only the data items appear, and not items subsequently computed. This copying is carried out automatically by Brick 15 as soon as any alterations have been made to a Station Vector. The calculations for a particular data point normally end when the Engine Vector is printed out by Brick 32. As soon as printout has finished, all Station Vectors in use are reset by being overwritten by their second copies: the Master Program is then re-entered at the beginning via the Jump Codeword Number of the Codeword associated with Brick 32.
5. Conclusions

The "Turbocode" Scheme for programming thermodynamic cycle calculations on an electronic digital computer has been described specifically in terms of the Ferranti "Pegasus" version, though an Algol 60 version is in course of development. The structure and general mode of operation of the scheme is described in the main body of the Report, while Appendices describe in some detail the Functions, Subroutines, and Bricks (i.e. major program segments) of which the Scheme is built up; they also give detailed information on the method of programming, the Brick specifications and the method of running a Turbocode program on the computer. Further information and copies of the various Program Tapes, may be obtained from the Department of Aircraft Propulsion of the College of Aeronautics.

Acknowledgement

The assistance and encouragement of Mr. B.V. Archer, formerly of the Department of Aircraft Propulsion, in the early stages of planning and programming the Turbocode Scheme is gratefully acknowledged.

References

(1) Bristol Siddeley Engines Ltd. - D.E.U.C.E. Programme No. 11017/1
    Scheme P

(2) Fielding, D. and Topps, J.E.C. - Thermodynamic Data for the
    Calculation of Gas Turbine Performance - Aeronautical Research
    Council R. & M. 3099
Appendix 1 - Writing a Typical Master Program

This appendix describes the process whereby a Turbocode Master Program is planned and written.

The problem is to find the design-point performance of a single-shaft turbojet, without reheat, having a fixed area convergent propelling nozzle.

The first step is to draw a sketch of the engine (see Fig. 3), and to mark on it the various components and the Stations, numbered in sequence through the engine, at which Station Vectors are to be calculated. Using the list of Bricks available (see Appendix 3), we now write on the diagram the numbers of the Bricks required, aligning them with the stations, or between the pairs of stations, to which they refer. Input Brick numbers are written on the left, output Brick numbers on the right, and Bricks dealing with calculations performed after the Station Vectors have been found are also written on the right. The choice of Bricks and the order in which they occur are largely self-evident, but the following points should be noted:

1. The compressor is here defined by its design-point pressure ratio, which requires the use of Brick 2: had the temperature rise been chosen, Brick 10 would have been used instead.

2. The compressor work must be calculated, by use of Brick 3, after the compressor calculation has been performed, but before undertaking the turbine calculation (since compressor work is an input data item to the turbine Brick 4): it is therefore convenient to place Brick 3 immediately after Brick 2.

3. Whether Brick 29 (in which the fundamental pressure loss is ignored) is used for the combustor, as here, or Brick 6, which calculates this loss, it is still necessary to allow for frictional pressure loss in the combustor. This is most conveniently done by using Brick 1 (also employed for the intake and jet pipe), which is placed before Brick 29. Since the fundamental loss is not calculated, a suitable allowance should be made for it in selecting the values of the parameters $\lambda_P$ and/or $\Delta P$ of Brick 1.

4. Since the convergent nozzle Brick 8 assumes isentropic flow, any losses due to the nozzle (other than those taken account of by the discharge and thrust coefficients in Brick 8) are combined with those of the jet pipe in the appropriate use of Brick 1.

5. Note that the output Bricks, 31 and 32, must occur in that order, since the latter resets the Station Vectors after output has finished and then jumps back to start a new calculation.

To facilitate the actual coding, printed Program Sheets A (for Codewords) and B (for listing Brick Data and Engine Vector items) are used, as shown in figs. 4a, 4b and 5. These are used as follows:

(a) On Sheet A (figs 4a, 4b) fill in the Codeword Numbers, Brick Numbers and Inlet and Outlet Station Vectors by referring to fig. 3.
Note that Codewords nos. 0-1 and 11-14 inclusive do not have Inlet and Outlet Station Vectors in the normal sense, so that elements a and b of these Codewords should be left blank at this stage.

(b) Using the Abbreviated Brick Specifications (see Appendix 3), list the various items of Brick Data, Engine Vector Results and Engine Vector Data opposite each Codeword.

(c) On Sheet B (fig. 5) fill in and number the Brick Data and Engine Vector items by copying them in order from Sheet A (the Engine Vector items being copied from the Engine Vector Results Column).

(d) On Sheet A, fill in the numbers appropriate to the first item for each Codeword in the Brick Data and Engine Vector Results Columns, copying them from Sheet B. Also number each Engine Vector Data Item individually.

(e) The columns c, d, and e for the "standard form" Codewords (i.e. Codewords nos. 2-10 inclusive) can now be filled in, remembering that it is not essential to fill in any right-hand side zeros. Note that the following right-hand side quantities, though significant, are zeros and so have not been filled in: Codeword 2 item a, Codeword 3 item c, Codeword 5 item d and Codeword 8 item e.

(f) Of the "non-standard form" Codewords, only nos. 11-14 require further additions as shown, in accordance with the requirements of their specifications. It will be noticed that Codeword 13 instructs Brick 31 to print Station Vectors 0-7 inclusive, while Codeword 14 instructs Brick 32 to print Engine Vector items 0-6 inclusive, and then to reset Station Vectors 0-6 inclusive (see remarks below on Data Tape). Item f of Codeword 14 is a significant zero, since it is required to jump back to Codeword 0 to read new data, but is omitted in accordance with the usual convention.

(g) Fill in the heading "D N SIMPLE TURBOJET TEST J200.0" and the ending "'(0)" instructing the program to start operating at Codeword 0.

The Program Tape can be punched directly from Sheet(s) A, remembering that, apart from the heading and ending, only the items in the columns between double vertical lines are actually punched, each item being separated from its neighbours by commas, and each Codeword being terminated by "Carriage Return, Line Feed" - as noted in the column headings. Figure 6 shows the Program Tape as punched.

In order to prepare the Data Tape, we note firstly that Brick 15 here precedes Brick 16, so that Station Vector Data must precede Brick Data, and secondly that each set of data in either category must be terminated by the characters "-1 CRLF". Figure 7 shows a Data Tape as punched. The following points should be noted:

(a) Each Station Vector item is a triad of signed numbers - viz: S.V. number, Item number, Item value - terminated by
CRLF. (see Specification of Brick 15).

(b) The items given fall into three groups:-

1. The minimum number of items ($\alpha_0, W_0, p_0$ and $V_0$) necessary to define S.V.O. (Brick 30 computes the rest).

2. The Turbine Inlet Temperature $T_4$.

3. Velocities at planes 1, 2, 3, 5 and 6 - these are necessary if the corresponding areas and static pressures and temperatures are needed, but otherwise can be omitted. The velocities at planes 4 and 7 are not given, since these are computed by Bricks 29 and 8 respectively (see specifications).

(c) Each Brick Data item is a pair of signed numbers - viz: Item number, Item value - terminated by CRLF. (see specification of Brick 16).

(d) Only one set each of S.V. Data and Brick Data are here shown, so that only one point will be calculated, but any number of further sets could have followed, subject to the following:-

1. S.V. Data must always precede Brick Data.

2. Only these items which have changed from the previous point need be stated, all other quantities being automatically reset to their previous values.

3. Nevertheless, something must always be provided for each of Bricks 15 and 16 to read: if no items have been changed, merely the terminating "-1 CRLF" suffices. (In the unlikely event of no change of item in either category - i.e. a repetition of the previous point - two successive "-1 CRLF" 's would be needed).

4. In accordance with Standard Pegasus practice, the Data Tape ends with the directive "Z", which causes the computer to stop. It can be restarted when a fresh Data Tape has been loaded, if necessary.

5. Name Sequences - i.e. the directive "N", followed by any sequence of characters not including two or more successive blanks, followed by at least two blanks - can be used freely.

(e) Obviously the numbering of the various items must agree precisely with the numbers allocated on the program sheets. It is not, however, essential for the items to be punched in the same order, though there is little reason not to do so.

(f) As noted earlier, the question of how many Station Vectors need resetting must be dealt with. Obviously one solution is to reset them all, but this is unnecessarily time-consuming, so the general rule is - reset only up to the last Station Vector for which one or more items are given on the Data Tape (in this case S.V.6) - see Data Tape (fig. 7) and Codeword 14(fig. 4b). Any higher-numbered Station Vectors must necessarily be recomputed afresh for each
Finally, fig. 8a shows the output resulting from running this Master Program with the given Data. The following points may be observed:

(a) Each "N" directive (including that on the Autocode master tape) causes the name following it to be punched.

(b) The "D" directive on the Master Program causes the Data and Serial Number to be punched.

(c) As each Brick is read in, its number and first block address in the main store is punched. After all Bricks have been read, the number of the last block used is punched.

(d) Since Brick 31 precedes Brick 32, the Station Vectors are punched before the Engine Vector.

(e) Fig. 8b shows a second section of output for the same program and data, for which Station Vector Output (apart from the printing of the S.V. nos. and areas for those S.V.'s whose areas are known) has been inhibited by depressing handswitch 13 on the console.
Appendix 2 - Notes on Functions, Subroutines and Bricks

While it is not essential for the user to know the mode of operation of the Bricks, nor of the Subroutines and Functions on which most of them depend, in order to write and use Turbocode Master Programs successfully, it may be useful and instructive to describe briefly how these basic constituents of the scheme work. Not every Brick is described, since some are so essentially simple in conception that their specifications (see Appendix 3) can provide all the necessary information.

A.2.1 Functions
A.2.1.1. Specific Heat

The specific heat of air at constant pressure is, at moderate pressures and in the absence of dissociation, a function only of temperature. In the range 200 - 2000 K, the empirical $C_p$ - temperature curve can be approximated to an adequate order of accuracy by a fourth-order polynomial in $x = \frac{T-1100}{900}$, where $T$ is the temperature in K, thus:

$$C_p^A = \sum_{i=0}^{4} a_i x^i$$

Another similar polynomial gives the specific heat of the stoichiometric products of combustion, thus:

$$C_p^S = \sum_{i=0}^{4} a_i x^i$$

Then for any fuel-air ratio $\alpha$ between zero and the stoichiometric value $\alpha_s$, we have $C_p = C_p^A + \frac{\alpha}{1 + \alpha} \cdot \frac{1 + \alpha_s}{\alpha_s} (C_p^S - C_p^A)$.

A simple routine known as SPHT, evaluates first $C_p^A$, then $\frac{1 + \alpha_s}{\alpha_s} (C_p^S - C_p^A)$, for given $T$, and finally $C_p$ for given $\alpha$.

Since the fuel assumed is the Standard Kerosine proposed by Topps, (ref. 2) which has the property that the molecular weight of its products of combustion is equal to that of air at all fuel-air ratios, its gas constant $\frac{R}{J}$ (equal to the difference between the specific heats at constant pressure and at constant volume) is fixed at a value of 0.0685522 C.H.U./lb, giving

$$C_v = C_p - \frac{R}{J}$$

and thence the specific heat ratio $\gamma = C_p/C_v$.

A.2.1.2. Enthalpy

Since the increment of enthalpy $dh = C_p dT$, the enthalpy relative to a given datum is obtained by term-by-term integration of the specific heat polynomials, leading to

$$h_A = \sum_{i=0}^{5} b_i x^i$$

and

$$h_S = \sum_{i=0}^{5} b_i x^i$$

whence at fuel-air ratio $\alpha$
\[ h = h_A + \frac{a}{1 + a} \cdot \frac{1 + a_S}{a_S} (h_S - h_A). \]

These expressions are evaluated by a routine (Called ENTH) similar to that used for \( C_p \), only the coefficients and the number of terms being different.

A.2.1.3. Isentropic Pressure Ratio

The increment of temperature-dependent entropy is given by \( ds^0 = C_p dT/T \), so that the value of \( s^0 \) relative to a given datum can be obtained by dividing the terms of the specific heat polynomial by \( T \) and then integrating, giving

\[ s_A^0 = \sum_{i=0}^{4} c_i x^i + c_5 \log T \text{ and } s_S^0 = \sum_{i=0}^{4} c_i' x^i + c_5' \log T, \]

whence at fuel-air ratio \( \alpha \)

\[ s_A^0 = s_A^0 + \frac{\alpha}{1 + \alpha} \cdot \frac{1 + a_S}{a_S} (s_S^0 - s_A^0). \]

In practice the temperature-dependent entropy is used to relate temperature and pressure changes across isentropic processes, in accordance with the relation

\[ \frac{R}{J} \log \frac{P_2}{P_1} = s_2^0 - s_1^0, \]

or

\[ \frac{P_2}{P_1} = \exp \frac{J(s_2^0 - s_1^0)}{R} = \exp \frac{J s_2^0}{R} \cdot \exp \frac{J s_1^0}{R}, \]

so that it is useful to work in terms of the function \( \pi = \exp \frac{J s_2^0}{R} \), from which \( \frac{P_2}{P_1} = \frac{\pi_2}{\pi_1} \). Physically, \( \pi \) is the isentropic pressure ratio necessary to reach the given temperature from the reference temperature (273.16°K).

The routine for evaluating \( \pi \) (called PRES), is largely the same in principle as those for \( C_p \) and \( h \), apart from the evaluation of the logarithmic term, and the final exponentiation.

A.2.1.4. Combustion Fuel-Air Ratio

It may readily be shown that if fuel is burnt at 100% efficiency in a medium (which is not, in general, pure air, but the products of an earlier combustion at fuel-air ratio \( \alpha_{in} \)), causing a temperature rise from \( T_{in} \) to \( T_{out} \), then the fuel-medium ratio \( \alpha_{out} \) (i.e. mass of added fuel divided by mass of medium) is given by

\[ \alpha_{out} = \Delta h^0_M / E.C.V. \]

where \( \Delta h^0_M \) = increase in enthalpy of medium between \( T_{in} \) and \( T_{out} \) and
E.C.V. = Effective Calorific Value = True Calorific Value of fuel
+ inlet enthalpy of fuel + \( \frac{1}{\alpha_s} \) \times \text{enthalpy of air at } T_{\text{out}}
- (1 + \frac{1}{\alpha_s}) \times \text{enthalpy of stoichiometric products at } T_{\text{out}}

Consequently the E.C.V. is a function of \( T_{\text{out}} \) for a given fuel composition and fuel temperature, and represents the gross calorific value less the heat necessary to raise the stoichiometric products through the temperature interval. The relevant routine (called FUEL) uses the ENTH function to evaluate \( \Delta h_M \) and E.C.V.

It can be shown that the final fuel-air ratio is given by
\[
\alpha_{\text{in}} + \alpha_{\text{out}} (1 + \alpha_{\text{in}}).
\]

A.2.2 Subroutines (S)

(a) S1 evaluates the change of enthalpy between two given temperatures at given fuel-air ratio by direct use of the ENTH function.

(b) S2 is the inverse of S1, finding the outlet temperature for given values of inlet temperature, fuel-air ratio and enthalpy change. The ENTH function gives the inlet enthalpy, and thence the outlet enthalpy. The outlet temperature appropriate to this enthalpy value is found iteratively, the enthalpies corresponding to the various outlet temperature values being compared with the known value. The starting approximation is \( T_{\text{in}} = \text{given } h/Cp (T \text{ inlet}) \) and the iteration formula is
\[
T_{n+1} = T_n - \frac{h(T_n) - \text{given } h}{Cp(T_n)}
\]

[Note: here and subsequently the notation \( Cp(T), h(T), \) etc, denotes that these quantities are functions of the bracketed argument.]

(c) S2a is a modification of S2 to find the temperature corresponding to a given absolute enthalpy and fuel-air ratio.

(d) S4 evaluates the isentropic pressure ratio between two given temperatures at given fuel-air ratio by direct use of the PRES function.

(e) S5 is the inverse of S4, finding the outlet temperature for given values of inlet temperature, fuel-air ratio and isentropic pressure ratio. The PRES function gives the inlet value of \( \pi \), and thence the outlet value. The outlet temperature appropriate to this \( \pi \) value is found iteratively, the \( \pi \) values corresponding to the various outlet temperature values being compared with the known value. The starting approximation is \( T_{\text{in}} = 273.16 \) (given \( \pi \)) \( R/JCp (T \text{ inlet}) \)
\( \text{since } \pi = 1 \text{ at } T = 273.16^\circ \text{K and } \alpha = 0 \), and the iteration formula
is \( T_{n+1} = T_n \left\{ 1 - \frac{R}{J\text{Cp}(T_n)} \left( 1 - \frac{\text{given } \pi}{\pi(T_n)} \right) \right\} \)

(f) \( S_7, S_{14} \) and \( S_{15} \) solve the continuity equation \( W = 144pAV/Rt \) to find \( V, A \) or \( W \) respectively for given values of the other two quantities, and of \( p \) and \( t \).

(g) \( S_8 \) and \( S_9 \) solve the Mach Number equation \( M = \sqrt{\frac{g_c\gamma(t)Rt}{V}} \) for \( M \) or \( V \) respectively for given values of the other of these quantities and of \( t \), using the SPHT function to find \( C_p \) and thence \( \gamma \).

(h) \( S_{11} \) finds the critical (i.e. sonic) values of \( p, t, V \) and \( A \), given \( \alpha, W, P, T \). This involves iterating on \( t \) to find values of

\[
V = \sqrt{2g_c\gamma(h(T) - h(t))}
\]

and of sonic velocity \( \alpha = \sqrt{g_c\gamma(t)Rt} \) which are equal. The starting approximation is \( t_0 = T-0.01^\circ K \),

and the iteration formula is

\[
t_{n+1} = t_n + \frac{\gamma(t_n)}{g_cJ\text{Cp}(t_n)} \frac{V(t_n)^2 - a_n^2}{1 + \gamma(t_n)Rt_n}
\]

(i) \( S_{12} \) and \( S_{17} \) find \( p, t, \) and \( V \) (supersonic and subsonic respectively) for given \( \alpha, W, P, T \) and \( A \). A preliminary use of \( S_{11} \) ensures that the given area is not less than the critical area (otherwise a solution is impossible). The value of \( t \) is then found iteratively to give a calculated area equal to the given value, using the relations

\[
p = \frac{P\pi(t)/\pi(T)}
\]

\[
V = \sqrt{2g_c\gamma(h(T) - h(t))}
\]

and \( A = WRt/144pV \). The starting approximation is \( t_0 = 200^\circ K \) (\( S_{12} \)) or \( t_0 = T - 0.01^\circ K \) (\( S_{17} \)), and the iteration formula is

\[
t_{n+1} = t_n - \frac{1 - \text{given } A/A(t_n)}{J\text{Cp}(t_n)\left[ \frac{g_c}{V^2(t_n)} - \frac{1}{\gamma(t_n)Rt_n} \right]}
\]

(j) \( S_{16} \) finds \( p, t \) and \( A \) for given \( \alpha, W, P, T \) and \( V \). The static-to-total enthalpy change is \( V^2/2g_cJ \), and \( S_2 \) is used to find \( t \) from this, enabling \( p = P\pi(t)/\pi(T) \) to be found. \( S_{14} \) then determines \( A \).

A.2.3 Bricks (B)

A.2.3.1 Input (B15 and B16)

Input of Data is controlled by B15 and B16, whose specifications give all necessary information.
A.2.3.2 Output (B31 and B32)

Output of Results is controlled by B31 and B32, whose specifications give all necessary information.

A.2.3.3 Other Non-Thermodynamic Bricks (B22 and B26)

(a) B22 permits arithmetic manipulation of Brick Data, Station Vector and Engine Vector items, as described in its specification.

(b) B26 is effectively a dummy brick whose purpose is to provide an extra codeword to extend the information contained in the codeword immediately following it. This is useful for certain Bricks which require unusually large amounts of data, from randomly distributed parts of the store. For further information see specifications of B26, B27 and B34.

A.2.3.4 Duct Flow, Mixing, etc. (B1, B5, B9, B24 and B30)

(a) B1 allows for relative or absolute changes of mass flow and/or total pressure in adiabatic flow, such as occur in intakes, jet pipes, combustion chamber diffusers and inter-compressor ducts. Conditions at the inlet and outlet Station Vectors (suffices a and b, respectively) are related by the equations:

\[ \frac{W_b}{W_a} = \frac{T_b}{T_a} = \frac{W_{b+1}}{W_{a+1}} = \frac{\lambda W}{\lambda P} = \frac{\Delta W}{\Delta P} \]

where \( \lambda W, \Delta W, \lambda P \) and \( \Delta P \) are given as Brick Data. If \( V_b \) is specified, S16 finds \( P_b, T_b \) and \( A_b \), while if \( A_b \) is specified, S17 finds \( P_b, T_b \) and \( V_b \) (subsonic).

(b) B5 deals with the frictionless parallel mixing of flows \( a \) and \( b \), leading to the mixed flow \( (b + 1) \). The mass flow, area, fuel-air ratio and total temperature are found by simple conservation equations:

\[ W_{b+1} = W_a + W_b \]
\[ A_{b+1} = A_a + A_b \]
\[ h(T_{b+1}) = \frac{W_a h(T_a) + W_b h(T_b)}{W_{b+1}} \]
\[ F_{b+1} = \frac{W_a \alpha_a}{1+\alpha_a} + \frac{W_b \alpha_b}{1+\alpha_b} \]
\[ \alpha_{b+1} = \frac{F_{b+1}}{W_{b+1} - F_{b+1}} \]

S2a is then used to find \( T_{b+1} \) from \( \alpha_{b+1} \) and \( h(T_{b+1}) \). To find the velocity, static pressure and static temperature, the momentum conservation equation must be satisfied, viz:-

\[ \left( \frac{W_{b+1} V_{b+1}}{g_c} + 144 p_{b+1} A_{b+1} \right) = \left( \frac{W_a V_a}{g_c} + 144 p_a A_a \right) + \left( \frac{W_b V_b}{g_c} + 144 p_b A_b \right). \]

This is accomplished by iteration on \( V_{b+1} \): since

\[ h(t_{b+1}) = h(T_{b+1}) - \frac{V_{b+1}^2}{2g_c}, \]

S2a can be used to find \( t_{b+1} \) and the above momentum equation to find \( p_{b+1} \). S7 then finds \( V_{b+1} \), which is substituted for the previous value if they do not agree sufficiently closely. Finally,

\[ p_{b+1} = p_{b+1} \pi (T_{b+1})/\pi (t_{b+1}) \]

(c) B29 deals with adiabatic total pressure loss in a parallel duct in terms of a total pressure loss coefficient

\[ \frac{p_a - p_b}{p_a - p_a} \]

across the duct, and \( p_b, t_b \) and \( V_b \) (subsonic) are found by S17.

(d) B24 merely computes intake momentum drag \( X_R = W_o V_0/g_c \).

(e) B30 is designed to "fill-in" missing items in a Station Vector if sufficient items are available to define it. Specifically, \( \alpha \) and either \( P \) and \( T \) or \( p \) and \( t \) must be given, together with any two of \( W, V \) and \( A \). The calculation is effected by using appropriate subroutines among S2, S4, S7, S14, S15, S16 and S17. This Brick is particularly useful for finding free stream conditions.

A.2.3.5. Compressor (B2, B3, B10 and B12)

(a) B2 calculates outlet conditions from an adiabatic compression process for which the inlet Station Vector and the total pressure ratio \( P_b/P_a \) and the polytropic efficiency \( \eta_{pol} \) (Brick Data items) are given, the relevant equations being;

\[ \alpha_b = \alpha_a \]
\[ W_b = W_a \]
\[ P_b = P_a \times P_b/P_a \]
Tb is found using S5 at the equivalent isentropic pressure ratio \( \frac{P_b}{P_a} \). The remaining elements of the outlet Station Vector are calculated by S16 or S17 according as \( V_b \) or \( A_b \) is given.

(b) B3 determines the work done in an adiabatic process

\[
\Delta H = W_b h(T_b) - W_a h(T_a)
\]

(c) B10 is similar to B2, but with the total temperature rise \( \Delta T_{ab} \) given instead of the total pressure ratio. Consequently \( T_b = T_a + \Delta T_{ab} \). S4 calculates the isentropic total pressure ratio \( \frac{P_b'}{P_a} \), whence the true ratio \( \frac{P_b}{P_a} = \left( \frac{P_b'}{P_a} \right)^{1/\eta_{pol}} \).

In other respects the calculation proceeds as in B2.

(d) B12 is similar to B10, the work input \( \Delta H \) being given instead of the total temperature rise. The enthalpy rise is \( \Delta H/W_a \), from which S2 finds \( T_b \). The rest of the calculation is as in B10.

### A.2.3.6 Turbine (B4 and B27)

(a) B4 calculates outlet conditions from an adiabatic turbine for which the inlet Station Vector, work output (consisting of a portion \( \Delta H \) required to drive a compressor - an Engine Vector Item - and a portion \( \delta H \) required to drive auxiliaries and other loads, and to make good losses - a Brick Data Item) - and adiabatic efficiency \( \eta_{ad} \) are given. The equations are:

\[
\begin{align*}
\alpha_b &= \alpha_a \\
W_b &= W_a \\
\Delta h_{ab} &= - \left( \Delta H + \delta H \right)/W_a, \text{ whence } T_b \text{ from S2.}
\end{align*}
\]

S2 is also used to find the isentropic outlet temperature \( T_b' \) associated with the isentropic enthalpy drop \( \Delta h_{ab}/\eta_{ad} \), and S4 is then used to find \( \frac{P_b}{P_a} \). The remaining items are then found in a similar manner to those for a compressor (see B2 above). For use in subsequent off-design calculations - \( \Delta h_{ab}/T_a \) is computed as an Engine Vector quantity.

(b) B27 deals with two adiabatic turbines in series in a turboprop engine, with provision for the cases of a two-spool engine with power offtake from the L.P. turbine (type 1) and for a free-turbine type (type 2). Station Vectors a, b, (b+1) denote H.P. inlet, L.P. inlet and L.P. outlet respectively. Besides the inlet Station Vector, the outlet static pressure and velocity must be specified. Then we have:-
S5 finds the isentropic outlet static temperature $t_{b+1}$, from which S1 finds the associated total-to-static enthalpy drop
\[ \{h(T_a) - h(t_{b+1})\} \text{. Now } \{h(T_{b+1}) - h(t_{b+1})\} = \frac{v_{b+1}^2}{2g_c} J \text{, and} \]
this quantity is assumed to be unaffected by turbine losses, enabling the isentropic total-to-total enthalpy drop \[\{h(T_a) - h(T_{b+1})\}\] to be found, and thence, by multiplication by the overall adiabatic efficiency, the actual drop. Use of S2 and S4 determines the values of $T_{b+1}$ and $P_{b+1}$, and S16 then finds $p_{b+1}, t_{b+1}$ and $A_{b+1}$.

The conditions at plane b, and the quantity $-\Delta h_{ab}/T_a$ are now computed in the same way as in B4. We then find the L.P. turbine gross work output $\Delta H_{b,b+1} = W_b[h(T_a) - h(T_{b+1}) - \delta h_{ab}]$; subtracting from this the compressor and auxiliary powers $\Delta H_{LP}$ and $\delta H_{LP}$ (both zero in the case of type 2) gives the net work output $\Delta H_{LP}$ net, from which the L.P. power follows from $SHP_{LP} = \Delta H_{LP} \text{ net } \times J/550$. Finally,
\[-\Delta h_{b,b+1}/T_b = -\Delta h_{b,b+1}/W_b T_b \text{ is calculated as an Engine Vector item.}\]

A.2.3.7 Combustion Chamber (B6, B29 and B39)

(a) B6 calculates the outlet fuel-air ratio and fundamental total pressure loss for frictionless constant-area "constant-pressure" heat addition up to a specified outlet total temperature $T_b$. Thus $A_b = A_a$, and $\alpha$ out is found using the FUEL function, leading to $W_b = Wa(1 + \alpha \text{ out})$, $ob = \alpha a + (1 + \alpha a)\alpha \text{ out}$ and the fuel flow (Engine Vector item) $F = 3600 \alpha \text{ out } Wa/n_b$.

A check is now made to ensure that the temperature rise demanded does not exceed the thermal choking limit. The outlet static temperature $t_b^*$ corresponding to choking is found iteratively as follows; from a starting value $t_b^* = Ta$, the SPHT function finds $Cp(t_b^*)$, whence $\gamma(t_b^*)$. By momentum conservation, $p_b^* = (pa + Wa Va/144g_c Aa)/\{\gamma(t_b^*) + 1\}$, and S7 then finds $V_b^*$ and S9 finds $M_b^*$. If $M_b^*$ is not equal to unity, a new value of $t_b^*$ is found from the iteration formula
\[ t_{b,n+1}^* = t_b^* / M_{b_n}^* \]
If the value of \( t_b^* \) exceeds 2000°K (the upper limit of the thermodynamic data), thermal choking cannot occur, and there is no further iteration.

Once \( t_b^* \) (being less than 2000°K) has been found, S2 finds \( T_b^* \) from \( t_b^* \) and \( V_b^*/2g_cJ \): if the value of \( T_b \) required exceeds \( T_b^* \), a symbol denoting "thermal choking" is printed, and exit occurs to the Jump Codeword. In the absence of choking, the actual outlet conditions are found by iteration on \( V_b \): using a starting value of 1000 ft/s, S2 calculates \( t_b \) from \( T_b \) and \( V_b^*/2g_cJ \), and momentum conservation gives

\[
\frac{p_b - p_a}{144g_cA_a} = \frac{W_aV_a}{144g_cA_a} - \frac{W_bV_b}{144g_cA_b}
\]

and a calculated velocity \( V_b' \) is then found by S7. If these values of \( V_b \) do not agree, the later value replaces the earlier and the cycle is repeated. Finally, S4 is used to find \( p_b \).

(b) B29 is a simpler alternative to B6 ignoring the fundamental total pressure loss. Consequently no question of thermal choking arises, and \( P_b \) is taken as equal to \( P_a \). S17 then finds the values of \( p_b, t_b \) and \( V_b \).

(c) B39 was written for a special application, and is not likely to be much used. It is the equivalent of B29 for constant volume heating, the principal difference being that the standard equation for the outlet fuel-medium ratio \( \alpha_{out} = \frac{\Delta h_M - R(T_b - T_a)/J}{E.C.V. + RT_b/J} \) is modified to

\[
\alpha_{out} = \frac{\{\Delta h_M - R(T_b - T_a)/J\}}{E.C.V. + RT_b/J}
\]

A.2.3.8 Heat Exchanger (B37 and B38)

The calculation of an air-to-air heat exchanger is performed jointly by B37(cold side) and B38(hot side). The Station Vectors are a and (a+1) for the cold side, and b and (b+1) for the hot side.

(a) B37. We have \( \alpha_{a+1} = \alpha_a, W_{a+1} = W_a, A_{a+1} = A_a \) and \( P_{a+1} = \lambda_{PC}P_a - \Delta P_c \) (analogously to B1). Now initially the hot side inlet temperature \( T_b \) is unknown, so for the first iteration we take \( T_b = T_{a+1} = T_a \). On subsequent iterations we have \( h(T_{a+1}) - h(T_a) = \eta_{th}\{h(T_b) - h(T_a)\} \), where \( \eta_{th} \) is the thermal ratio (Brick Data item), and from this quantity and \( T_a \) S2 calculates \( T_{a+1} \).
(b) **B38.** We have \( \alpha_{b+1} = \alpha_b, W_{b+1} = W_b, A_{b+1} = A_a \)

and \( P_{b+1} = \lambda P_H P_b - \Delta P_H \). On entering the Brick, \( T_b \) will have been calculated by an earlier brick (usually B4) and in general will not agree with the value most recently used by B37. Consequently exit occurs to the Jump Codeword corresponding to B37, which proceeds to recalculate using the new value of \( T_b \).

When the values of \( T_b \) agree, the iteration ceases, and we find \( h(T_b) - h(T_{b+1}) = \frac{W_a}{W_b} \{ h(T_{a+1}) - h(T_a) \} \), which S2 uses with \( T_b \) to find \( T_{b+1} \). Finally, S17 calculates \( p_{b+1}, t_{b+1} \) and \( V_{b+1} \).

**A.2.3.9 Propelling Nozzle (B8, B23 and B25)**

(a) **B8** deals with the design point calculation of a simple convergent nozzle in isentropic flow, for which the inlet Station Vector and the outlet static pressure (equal to the free stream value \( P_0 \)) are given. Then \( \alpha_b = \alpha_a, W_b = W_a, P_b = P_a \) and \( T_b = T_a \). It is initially assumed that the nozzle is unchoked, so that S5 calculates \( t_b \) from \( T_b \) and \( P_0/P_b \). S1 then finds the associated enthalpy drop \( \Delta h_b \), from which \( V_b = \sqrt{\frac{2g_c J \Delta h_b}{\lambda}} \). S8 then calculates \( M_b \): if this is not greater than unity, the nozzle is indeed unchoked, \( p_b = P_0 \), and S14 calculates the theoretical area \( A_b' \). Then the true area \( A_b = A_b'/C_D \), where \( C_D \) is the discharge coefficient (Brick Data Item), the theoretical gross thrust \( X_G' = W_b V_b / g_c \), and the true gross thrust \( X_G = X_G' C_T \), where \( C_T \) is the thrust coefficient (Brick Data Item), are calculated.

If, however, \( M_b \) is found to exceed unity (on the now fallacious assumption that \( p_b = P_0 \)), the nozzle is choked, and it is necessary to iterate on \( t_b \) to find the exit conditions. From a starting value \( t_b = T_b / 1.125 \), S1 finds the total-to-static enthalpy drop \( \Delta h_b \), whence \( V_b = \sqrt{2g_c J \Delta h_b} \), and S8 calculates \( M_b \). If \( M_b \) is not equal to unity, the iteration formula \( t_{b+1,n} = t_{b,n} \times M_{b,n} \) is used to find a new value of \( t_b \). When the correct value of \( t_b \) has been found, S16 finds \( p_b \) and \( A_b' \). The rest of the calculation proceeds as for the
unchoked case, except that now

\[ X_G' = \frac{W_b}{g_c} V_b + 144 A_b (p_b - p_o) \]

(b) B23 correspondingly deals with the design point calculation of a convergent-divergent nozzle in isentropic flow. Station Vectors a, b and (b+1) refer to inlet, throat and exit respectively. The calculation of exit conditions, and of \( X_G \)

(for a given value of \( p_{b+1} = p_o \)) proceeds exactly as for the unchoked convergent nozzle of B8, except that \( M_{b+1} \) is not constrained to be subsonic. If \( M_{b+1} \) is found to be not greater than unity, the nozzle is in fact convergent, and exit from the Brick occurs at this point after printing an appropriate symbol denoting "convergent nozzle".

Otherwise throat conditions are calculated by the method adopted in B8 for the choked case.

(c) B25 calculates the off-design behaviour of a convergent-divergent nozzle, assuming that either (i) \( A_b/A_{b+1} \) (Brick Data item) or (ii) \( A_{b+1} \) (Station Vector item) is fixed at a known value. Since the alternative possibilities are rather involved, it may be helpful to refer to Fig. 9 in reading this description.

It is first assumed that the nozzle is choked, and the throat conditions are calculated by the method of B8 (choked case) (point A, fig. 9). Since this gives a value for \( A_b \), the value of \( A_{b+1} = A_b : A_b/A_{b+1} \) can be found in case (i). In case (ii), it is necessary to check that the given value of \( A_{b+1} \) is not less than \( A_b \), otherwise \( A_{b+1} \) is too small for the given flow: if this should be so, a symbol is printed to denote "\( A_{b+1} < A_b \)" and exit occurs to the Jump Codeword.

The next stage is to calculate the exit conditions appropriate to the known value of \( A_{b+1} \) for subsonic \( V_{b+1} \) (Point B). Thus \( \sigma_{b+1} = \sigma_a \), \( W_{b+1} = W_a \), \( P_{b+1} = P_a \) and \( T_{b+1} = T_a \), and S17 finds \( p_{b+1} \), \( t_{b+1} \) and \( V_{b+1} \). In case (ii), if this value of \( p_{b+1} \) is less than \( p_0 \), the nozzle cannot accommodate the flow with the given area, so a symbol denoting "\( p_{b+1} < p_0 \)" is printed and exit occurs to the Jump Codeword. In case (i) with \( p_{b+1} < p_0 \), the nozzle is in fact unchoked (i.e. the actual operating point is above Point B in fig. 9) and we recalculate with \( p_{b+1} = p_0 \), using S5 to find \( t_{b+1} \) and S1 to find \( \Delta h_{b+1} \).
whence \( V_{b+1} = \sqrt{2g_c J \Delta h_{b+1}} \) and calculating \( A_{b+1} \) from S14 (Point C). Then \( A_b = A_{b+1} \times A_b/A_{b+1} \) and S17 recalculates \( p_b, t_b \) and \( V_b \) (subsonic)(Point D).

If, however, the subsonic value of \( p_{b+1} \) is not less than \( p_o \), this last phase of the calculation is repeated for the known value of \( A_{b+1} \) for supersonic \( V_{b+1} \) (i.e. using S12 instead of S17)(Point E). We now assume a normal shock at the exit plane, whose outlet conditions are given by Station Vector \((b+2)\). Then \( a_{b+2} = a_a, \ W_{b+2} = W_a, \ T_{b+2} = T_a \) and \( A_{b+2} = A_{b+1} \). We iterate on \( V_{b+2} \) to find exit conditions consistent with momentum conservation in the same way as for B6 (Point F). If the resulting value of \( p_{b+2} \geq p_o \) (i.e. if the actual operating point is below Point F in fig. 9), there is no shock in the divergence, and the calculation is complete.

The remaining possibility (that \( p_{b+2} < p_o \)) implies that there is indeed a shock in the divergence (though not necessarily at exit), so that \( p_{b+1} = p_o \). S12 is therefore used to find \( p_{b+1}, t_{b+1} \) and \( V_{b+1} \) (supersonic) consistent with the known value of \( A_{b+1} \).

In all cases, the calculation concludes by evaluating

\[ X_G' = \frac{W_{b+1} V_{b+1}}{g_c} + 144 A_{b+1} (p_{b+1} - p_o) \text{ and } X_G = X_G' C_T. \]

A.2.3.10 Various Bricks for Completing the calculation (B33, B34, B35 and B36).

(a) B33 is very brief, since it merely calculates the net thrust and specific fuel consumption of a turbojet from the equations

\[ X_N = X_G - X_R \text{ and s.f.c.} = F/X_N. \]

(b) B35 is used in simplified turbojet off-design calculations to find the turbine inlet temperature. For previously calculated or given values of compressor work \( \Delta H \), auxiliary work \( \delta H \), turbine mass flow \( W_b \) and turbine enthalpy drop ratio \( -\Delta h_{ab}/T_a \) (assumed constant at the design-point value), the brick calculates \( \Delta h_{ab} = (\Delta H + \delta H)/W_b \) and \( T_a = -\Delta h_{ab} \div -\Delta h_{ab}/T_a. \) If this value does not agree with the value previously used (by the combustion brick), the new value of \( T_a \) replaces the previous one and a Jump Exit occurs to the codeword corresponding to the combustion brick.
(c) B36 is used in simplified turbojet off-design calculations to find the inlet air mass flow, assuming the propelling nozzle (of given throat area $A_a$, which is a Brick Data item) to be choked. If the value of $A_a$ calculated by the nozzle brick does not agree with the given value, the previously used value of $W_o$ is scaled according to the equation

$$\text{new } W_o = \text{old } W_o \times \frac{\text{given } A_a}{\text{calculated } A_a}$$

and a Jump Exit occurs to the Codeword appropriate to the Brick in which $W_o$ is first used.

(d) B34 finds the off-design bypass ratio for a turbofan (bypass) engine in which the bypass and turbine exhaust streams mix. Station Vectors a, b, (b+1) denote bypass duct exit, (L.P.) turbine inlet and turbine outlet (before actual mixing occurs) respectively. It is necessary for $V_{b+1}$ to be specified.

Since mixing requires that $p_{b+1} = p_a$, conditions at plane (b+1) are found for a known pressure ratio $p_{b+1}/p_b$, in the same manner as for B27. For given auxiliary work $\delta H$, the "free" output of the turbine is $\Delta H_{\text{free}} = W_{b+1} \Delta h_{b,b+1} - \delta H$. Now at an earlier stage in the calculation, the proportions $\lambda_a, \lambda_b$ of the fan mass flow (such that $\lambda_a + \lambda_b = 1$) passing through the gas generator and bypass duct respectively will have been calculated or specified. Then if $\Delta H$ is the work input to the fan, power balance between fan and turbine requires that the value of $\lambda_b$ be scaled according to the equation.

$$\lambda_{b_{n+1}} = \lambda_b \times \frac{\Delta H}{\Delta H_{\text{free}}}$$

leading to the new value $\lambda_{a_{n+1}} = 1 - \lambda_{b_{n+1}}$.

If either of $\lambda_a, \lambda_b$ falls outside the range 0 to 1, it is impossible to obtain a solution (because the turbine exit pressure is below the fan exit pressure even at zero bypass ratio), so a symbol denoting "no solution" is printed and a Jump Exit occurs.

Otherwise, the Engine Vector items bypass ratio $\lambda_a/\lambda_b$ and turbine enthalpy drop ratio $\Delta h_{b,b+1}/T_b$ are calculated, all gas generator mass flows and areas are scaled in the ratio $\lambda_{b_{n+1}}/\lambda_b$, as is the main fuel flow, while the bypass duct mass flow, area and fuel flow (if any) are scaled in the ratio $\lambda_{a_{n+1}}/\lambda_a$. 
## Appendix 3 - Abbreviated Brick Specifications

### List of Bricks Available

<table>
<thead>
<tr>
<th>Brick</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Transformation of Mass Flow and Total Pressure</td>
</tr>
<tr>
<td>2</td>
<td>Compression (given Total Pressure Ratio)</td>
</tr>
<tr>
<td>3</td>
<td>Work Done</td>
</tr>
<tr>
<td>4</td>
<td>Single Turbine</td>
</tr>
<tr>
<td>5</td>
<td>Frictionless Constant-Area Mixing</td>
</tr>
<tr>
<td>6</td>
<td>Constant Pressure Combustion with Fundamental Pressure Loss</td>
</tr>
<tr>
<td>8</td>
<td>Convergent Nozzle</td>
</tr>
<tr>
<td>9</td>
<td>Constant-Area Duct Total Pressure Loss</td>
</tr>
<tr>
<td>10</td>
<td>Compression (given Total Temperature Rise)</td>
</tr>
<tr>
<td>12</td>
<td>Compression (given Work Input)</td>
</tr>
<tr>
<td>15</td>
<td>Station Vector Input</td>
</tr>
<tr>
<td>16</td>
<td>Brick Data Input</td>
</tr>
<tr>
<td>22</td>
<td>Arithmetic on Engine Vector, Station Vectors and Brick Data</td>
</tr>
<tr>
<td>23</td>
<td>Optimum Convergent-Divergent Nozzle</td>
</tr>
<tr>
<td>24</td>
<td>Intake Momentum Drag</td>
</tr>
<tr>
<td>25</td>
<td>Off-Design Convergent-Divergent Nozzle</td>
</tr>
<tr>
<td>26</td>
<td>Supplementary Codeword (for use with Bricks 27 and 34)</td>
</tr>
<tr>
<td>27</td>
<td>Two Turbines in series</td>
</tr>
<tr>
<td>29</td>
<td>Constant Pressure Combustion without Fundamental Pressure Loss</td>
</tr>
<tr>
<td>30</td>
<td>Completing Station Vector</td>
</tr>
<tr>
<td>31</td>
<td>Station Vector Output</td>
</tr>
<tr>
<td>32</td>
<td>Engine Vector Output and Station Vector Reset</td>
</tr>
<tr>
<td>33</td>
<td>Turbojet Net Thrust and Specific Fuel Consumption</td>
</tr>
<tr>
<td>34</td>
<td>Determination of Off-Design Bypass Ratio</td>
</tr>
<tr>
<td>35</td>
<td>Determination of Turbojet Off-Design Turbine Inlet Temperature</td>
</tr>
<tr>
<td>36</td>
<td>Determination of Off-Design Intake Mass Flow</td>
</tr>
<tr>
<td>37</td>
<td>Air-to-Air Heat Exchanger (Cold Side)</td>
</tr>
<tr>
<td>38</td>
<td>Air-to-Air Heat Exchanger (Hot Side)</td>
</tr>
<tr>
<td>39</td>
<td>Constant Volume Combustion</td>
</tr>
</tbody>
</table>
Brick 1: Transformation of Mass Flow and Total Pressure

Description: Given $S_a$, either $V_b$ or $A_b$ (optional) and the factors $\lambda_w$, $\Delta W$, $\lambda_p$, and $\Delta P$, calculates $S_b$ (or part of it) using the equations

\[
\begin{align*}
\alpha_b &= \alpha_a \\
W_b &= \lambda_w W_a - \Delta W \\
P_b &= \lambda_p P_a - \Delta P \\
T_b &= T_a
\end{align*}
\]

$P_b$, $T_b$ and $A_b$ or $V_b$ are calculated from continuity if $V_b$ or $A_b$ given.

Form of Codeword: 1, a, b, c

S.V.: $\alpha_a$, $W_a$, $P_a$, $T_a$

$V_b$ or $A_b$ (optional)

B.D.: $\lambda_w$ (dimensionless), $\Delta W$ (lb/s), $\lambda_p$ (dimensionless), $\Delta P$ (lbf/in²)

E.V. Data: nil

E.V. Results: nil

Brick 2: Compression (given Total Pressure Ratio)

Description: Given $S_a$, either $V_b$ or $A_b$ (optional), $P_b/P_a$ and $\eta_{pol}$, calculates $S_b$ (or part of it).

Form of Codeword: 2, a, b, c

S.V.: $\alpha_a$, $W_a$, $P_a$, $T_a$

$V_b$ or $A_b$ (optional)

B.D.: $P_b/P_a$ (dimensionless), $\eta_{pol}$ (dimensionless: given as a fraction, not as a percentage)

E.V. Data: nil

E.V. Results: nil

Brick 3: Work Done

Description: Given $\alpha_a$, $W_a$, $T_a$, $\alpha_b$ and $T_b$, calculates $\Delta H = W_a(h_{T_b} - h_{T_a})$
Form of Codeword: 3, a, b, 0, d
S.V. : $\alpha_a$, $W_a$, $T_a$, $\alpha_b$, $T_b$
B.D. : nil
E.V. Data : nil
E.V. Results : $\Delta H$(CHU/s)

Brick 4 : Single Turbine
Description: Given $S_a$, either $V_b$ or $A_b$ (optional), compressor work $\Delta H$, auxiliary work $\delta H$, and $\eta_{ad}$, calculates $S_b$ (or part of it) and $-\Delta h/T_a$.

Form of Codeword: 4, a, b, c, d, e
S.V. : $\alpha_a$, $W_a$, $P_a$, $T_a$
$V_b$ or $A_b$ (optional)
B.D. : $\delta H$(CHU/s), $\eta_{ad}$ (dimensionless: given as a fraction, not as a percentage).
E.V. Data : $\Delta H$(CHU/s)
E.V. Results : $-\Delta h/T_a$(CHU/lb°K)

Brick 5 : Frictionless Constant-Area Mixing
Description: Given $S_a$ and $S_b$, calculates $S_{b+1}$ assuming frictionless constant-area mixing.

Form of Codeword: 5, a, b
S.V. : $\alpha_a$, $W_a$, $P_a$, $T_a$, $V_a$, $A_a$
$\alpha_b$, $W_b$, $P_b$, $T_b$, $V_b$, $A_b$
B.D. : nil
E.V. Data : nil
E.V. Results : nil

Remarks : In allocating S.V. Nos., it is important to number those of one of the inlet streams ($S_b$) and of the mixture ($S_{b+1}$) consecutively.
Brick 6: Constant Pressure Combustion with Fundamental Pressure Loss

Description: Given $S_a$, $T_b$ and combustion efficiency $\eta_b$, calculates $S_b$, fuel flow $F$ and thermal choking temperature $T_b^*$. If given $T_b$ exceeds $T_b^*$, latter is printed on a new line and Jump Exit occurs.

Form of Codeword: 6, a, b, c, d, 0, f

S.V.: $\alpha_a$, $W_a$, $p_a$, $T_a$, $V_a$, $A_a$, $T_b$

B.D.: $\eta_b$ (dimensionless - given as a decimal fraction, not as a percentage)

E.V. Data: nil

E.V. Results: $F$ (lb/h)

Brick 8: Convergent Nozzle

Description: Given $S_a$, $p_0$ (ambient), discharge coefficient $C_D$ and thrust coefficient $C_T$, calculates $S_b$ and gross thrust $X_G$, assuming an isentropic convergent nozzle.

Form of Codeword: 8, a, b, c, d

S.V.: $p_0$, $\alpha_a$, $W_a$, $P_a$, $T_a$

B.D.: $C_D$, $C_T$ (both dimensionless)

E.V. Data: nil

E.V. Results: $X_G$ (lbf)

Brick 9: Constant-Area Duct Total Pressure Loss

Description: Given $S_a$ and pressure loss factor $\Delta P/(P_a - p_a)$, calculates $S_b$.

Form of Codeword: 9, a, b, c

S.V.: $\alpha_a$, $p_a$, $P_a$

B.D.: $\Delta P/(P_a - p_a)$

E.V. Data: nil

E.V. Results: nil

Brick 10: Compression (given Total Temperature Rise)

Description: Given $S_a$, either $V_b$ or $A_b$ (optional), $\Delta T_{ab}$ and $\eta_{pol}$, calculates
$S_b$ (or part of it).

Form of Codeword: 10, a, b, c

S.V. : $\alpha_a$, $W_a$, $P_a$, $T$

$V_b$ or $A_b$ (optional)

B.D. : $\Delta T_{ab}$(deg C), $\eta_{pol}$ (dimensionless: given as a decimal fraction, not as a percentage)

E.V. Data : nil

E.V. Results : nil

Brick 12 : Compression (given Work Input)

Description: Given $S_a$, either $V_b$ or $A_b$ (optional), $\Delta H$ and $\eta_{pol}$, calculates $S_b$ (or part of it)

Form of Codeword: 12, a, b, c, 0, e

S.V. : $\alpha_a$, $W_a$, $P_a$, $T_a$

$V_b$ or $A_b$ (optional)

B.D. : $\eta_{pol}$ (dimensionless: given as a decimal fraction, not as a percentage)

E.V. Data : $\Delta H$ (CHU/lb)

E.V. Results : nil

Brick 15 : Station Vector Input

Description: Reads S.V. items from data tape on Reader A and stores them, also making a duplicate copy for subsequent resetting by Brick 32. Missing items in any S.V. automatically set equal to a small negative number, to enable parts of bricks concerned with calculating $p$, $t$, $V$ and $A$ to be bypassed if essential data is not provided.

Each item consists of three signed numbers:-

1. Integer denoting S.V.no., terminated by Sp (or CRLF);
2. Integer denoting item no. within S.V. ($\alpha = 0$, $W = 1$, $p = 2$, $P = 3$, $t = 4$, $T = 5$, $V = 6$, $A = 7$), terminated by Sp (or CRLF);
3. Number denoting value to which item is to be set, terminated by Sp (or CRLF). List of items is terminated by "-1 CRLF".

Form of Codeword: 15

S.V. : nil
Remarks: Since starting S.V. values are reset at the end of each cycle, for second and subsequent cycles it is necessary to punch only those items which are to be changed. If no alterations are required, only the terminal "-CRLF" is needed.

Brick 16: Brick Data Input

Description: Reads B.D. items from data tape on Reader A and stores them. Each item punched as two signed numbers, each terminated by Sp or CRLF, of which first is B.D. Item No. and second its value. List of items is terminated by "-1 CRLF"

Form of Codeword: 16

S.V.: nil

B.D.: nil

E.V. Data: nil

E.V. Results: nil

Remarks: B.D. items are rarely over-written by any brick (exceptions being given in the appropriate specifications); thus for second and subsequent cycles it is usually necessary to punch only such items as are to be altered. If no alterations are required, only the terminal "-1 CRLF" is needed.

Brick 22: Arithmetic on Engine Vector, Station Vectors and Brick Data

Description: This Brick is provided to make possible simple arithmetic operations on Engine Vector Items, and also transfers of items between the Engine Vector and a Station Vector or Brick Data. Element c denotes the operation required, and elements a, b and d the operands, in accordance with the following scheme:

- c = 0: E.V.ₐ × E.V.ₑ to E.V.ₜ
- c = 1: E.V.ₐ/E.V.ₑ to E.V.ₜ
- c = 2: E.V.ₐ + E.V.ₑ to E.V.ₜ
- c = 3: E.V.ₐ - E.V.ₑ to E.V.ₜ
- c = 4: -E.V.ₐ to E.V.ₜ
- c = 5: B.D.ₐ to E.V.ₜ
c = 6: E.V. to B.D.

\[ \text{c = 7: S.V.} \rightarrow \text{E.V.} \]
where \( k \) denotes item b of S.V.

\[ \text{c = 8: E.V.} \rightarrow \text{S.V.} \]

If \( c \) is put greater than 8, a long loop stop (jumping back to label 22) occurs.

Form of Codeword: 22, a, b, c, d

S.V. )
B.D. )
E.V. Data )
E.V. Results)

Brick 23: Optimum Convergent-Divergent Nozzle

Description: Given \( S_a \) (ambient), \( p_0 \) (ambient), discharge coefficient \( C_D \) and thrust coefficient \( C_T \), calculates \( S_b \) (throat) and \( S_{b+1} \) (exit) and gross thrust \( X_G \), assuming a convergent-divergent nozzle designed for the pressure ratio actually prevailing. But if this pressure ratio is less than the critical value, a convergent nozzle is assumed, and \( S_{b+1} \) is meaningless - this case is indicated by automatic punching of the symbol "-7777" on a new line ahead of the main results, thereby warning the user to ignore \( S_{b+1} \) results.

Form of Codeword: 23, a, b, c, d

S.V.: \( p_0, \alpha_a, W_a, P_a, T_a \)
B.D.: \( C_D, C_T \) (both dimensionless)
E.V. Data: nil
E.V. Results: \( X_G \) (lbf)

Brick 24: Momentum Drag

Description: Given \( S_a \) (free stream panel), calculates momentum drag \( X_R \).

Form of Codeword: 24, a, 0, 0, d

S.V.: \( W_a \) and \( V_a \)
B.D.: nil
E.V. Data: nil
E.V. Results : \( X_R \) (lbf)

**Brick 25 : Off-Design Convergent-Divergent Nozzle**

**Description:** Given \( S_a \), \( p_o \) (ambient), discharge coefficient \( C_D \) and thrust coefficient \( C_T \), calculates \( S_b \) (throat) and \( S_{b+1} \) (exit) and gross thrust \( X_G \) for either given divergence area ratio \( A_b / A_{b+1} \) or given exit area \( A_{b+1} \). Jump exit occurs in latter case if no solution is possible.

**Form of Codeword:** 25, a, b, c, d, 0, f

**S.V.** : \( p_o, \alpha_a, W_a, P_a, T_a \)

\( A_{b+1} \) (unless \( A_b / A_{b+1} \) specified)

**B.D.** : \( A_b / A_{b+1} \) (set equal to zero if \( A_{b+1} \) specified), \( C_D, C_T \) (all dimensionless)

**E.V. Data : nil**

**E.V. Results : nil**

**Remarks:** Impossible cases are identified by printed symbols as follows:

1. If \( A_b > A_{b+1} \), "-999" is printed
2. If subsonic \( p_{b+1} < p_o \), "-888" is printed

**Note** Station Vector \((b+2)\) must be kept free for working space for this Brick.

**Brick 26 : Supplementary Codeword (for use with Bricks 27 and 34)**

**Description:** Supplies additional codeword elements for Bricks 27 and 34, for which standard codeword length is insufficient.

**Form of Codeword:** 26, a, b, c, d, e

where the elements denote the addresses of the following items:

<table>
<thead>
<tr>
<th>Element</th>
<th>For B27</th>
<th>For B34</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Exit pressure S.V. No.</td>
<td>H.P. Compressor Inlet S.V. No.</td>
</tr>
<tr>
<td>b</td>
<td>Type No. (1=2 spool with L.P. offtake, 2 = free L.P.)</td>
<td>F main ((lb/h))</td>
</tr>
<tr>
<td>c</td>
<td>( \delta H_{HP}(\text{CHU/lb}) ), followed by ( \eta_{adHP} ) (dimensionless)</td>
<td>( \lambda_a ) (dimensionless)</td>
</tr>
</tbody>
</table>
d \quad -(\Delta h/T)_{HP} \ (\text{CHU}/°K) \quad \text{Bypass ratio (dimensionless), then } -(\Delta h/T)_{LP} \ (\text{CHU}/°K) \\
\begin{align*}
e & \quad \Delta H_{HP} \ (\text{CHU}/\text{lb}) \\
& \quad F_{\text{bypass}} \ (\text{lb}/\text{h})
\end{align*}

S.V. : nil \\
B.D. : nil \\
E.V. Data : nil \\
E.V. Results : nil \\
Remarks: The Codeword for this Brick must precede that for Bricks 27 or 34.

Brick 27 : Two Turbines in series

Description: Given $S_a$, $V_b$ or $A_b$ (optional), $P_{b+1}$, $V_{b+1}$ and power offtakes and adiabatic efficiencies of each turbine, for either a two spool (Type 1) or free turbine (Type 2) arrangement, calculates $S_b$ and $S_{b+1}$, also $-\Delta h/T$ for each turbine and residual SHP of L.P. turbine.

Form of Codeword: 27, a, b, c, d, e \\
S.V. : $a_a$, $W_a$, $P_a$, $T_a$ \\
\quad $V_b$ or $A_b$ (optional) \\
\quad $V_{b+1}$ \\
B.D. : $\Delta H_{LP} \ (\text{CHU}/\text{lb})$ and $\eta_{ad}$ overall (dimensionless - given as a decimal fraction, not as a percentage) \\
E.V. Data : $\Delta H_{LP} \ (\text{CHU}/\text{lb})$ \\
E.V. Results : $-(\Delta h/T)_{LP} \ (\text{CHU}/°K)$, $\text{SHP}_{LP}$, also $-(\Delta h/T)_{HP} \ (\text{CHU}/°K)$ \\
(addressed by item d of Brick 26 Codeword, q.v.) \\
Remarks: The Codeword for this Brick must follow that for Brick 26. L.P. turbine inlet and outlet planes must be numbered consecutively.

Brick 29 : Constant Pressure Combustion without Fundamental Pressure Loss

Description: Given $S_a$, $T_b$, $A_b$ (optional) and combustion efficiency $\eta_b$, calculates $S_b$ and fuel flow $F$.

Form of Codeword: 29, a, b, c, d \\
S.V. : $a_a$, $W_a$, $T_a$, $T_b$ \\
\quad $A_b$ (optional)
B.D. : $\eta_b$ (dimensionless - given as a decimal fraction, not as a percentage)

E.V. Data : nil

E.V. Results : F (lb/h)

Remarks: Since no allowance is made for total pressure loss or mass addition, this Brick must be preceded by Brick 1 (Transformation) when these effects are to be allowed for. (Compare Brick 6).

Brick 30 : Completing Station Vector

Description: Given $\alpha_a$ and either $p_a$ and $t_a$ or $P_a$ and $T_a$, and any two of $W_a$, $V_a$ and $A_a$, calculates the missing items of $S_a$.

Form of Codeword: 30, a

S.V. : $\alpha_a$, and either $p_a$ and $t_a$ or $P_a$ and $T_a$, and any two of $W_a$, $V_a$, $A_a$.

B.D. : nil

E.V. Data : nil

E.V. Results : nil

Brick 31 : Station Vector Output

Description: Punches $S_a$ to $S_b$ inclusive, provided that handswitch 13 is up. Each $S$ is given in the following format:-

First Line: S.V. No.
Second Line: $\alpha$, $W$, $p$, $P$
Third Line: $t$, $T$, $V$, $A$

All items are in fixed point representation. Missing items will appear as small negative numbers.

When handswitch 13 is down, only the following items are punched:-

S.V. No. and $A_a$, both on the same line, in fixed point representation. But there is no printing if $A_a$ has not been calculated.

Form of Codeword: 31, a, b

S.V. : $S_a$ to $S_b$ inclusive (not necessarily complete)

B.D. : nil

E.V. Data : nil

E.V. Results : nil
Remarks: This Brick must always precede Brick 32 (E.V. output and S.V. Reset) when both are used.

Brick 32: Engine Vector Output and Station Vector Reset

Description: Punches the E.V. list, up to a specified last item. Items are punched in floating point representation, four to a line. $S_a$ to $S_b$ inclusive are then reset to the values obtaining at the start of the current calculation.

Form of Codeword: 32, a, b, 0, d

where $a$, $b$ are first and last S.V. Nos. to be reset and $d$ is the number of last E.V. item to be punched.

S.V.: nil

B.D.: nil

E.V. Data: The first $(d+1)$ items of the E.V. list.

E.V. Results: nil

Remarks: This Brick must always follow Brick 31 (S.V. Output) when both are used.

Brick 33: Turbojet Net Thrust and Specific Fuel Consumption

Description: Given gross thrust $X_G$, momentum drag $X_R$ and fuel flow $F$, calculates net thrust $X_N$ and specific fuel consumption $s.f.c.$

Form of Codeword: 33, 0, b, c, d, e

where $b$ = E.V. item no. of $F$

$c$ = " " " " $X_G$

d = " " " " $X_N$

e = " " " " $X_R$

S.V.: nil

B.D.: nil

E.V. Data: $X_R$ (lbf), $X_G$(lbf), $F$(lb/h)(not necessarily consecutive, or in that order)

E.V. Results: $X_N$(lbf), s.f.c.(lb/h lbf)

Remarks: This Brick is suitable for turbojets, turbofans, bypass engines and ramjets, but not for shaft power engines.
Brick 34: Determination of Off-Design Bypass Ratio

Description: Given bypass duct outlet and L.P. turbine inlet S.V.'s \( \text{S}_\text{a} \) and \( \text{S}_\text{b} \), compressor work \( \Delta \text{H} \), auxiliary work \( \delta \text{H} \), and L.P. turbine adiabatic efficiency \( \eta_{\text{ad}} \), calculates L.P. turbine outlet S.V. \( \text{S}_{\text{b}+1} \). \( V_{\text{b}+1} \) is given. Also finds bypass ratio to give power balance between fan and L.P. turbine, assuming bypass and turbine flows mix at same static pressure, and alters mass flows and areas to suit. Jump Exit if Bypass ratio negative.

Form of Codeword: 34, a, b, c, d, e, f

where:
- \( a \) = bypass duct outlet S.V.
- \( b \) = L.P. turbine inlet S.V.
- \( c \) = B.D. item no. of \( \lambda_b \)
- \( d \) = B.D. item no. of \( \Delta \text{H} \)
- \( e \) = B.D. item no. of \( \delta \text{H} \), followed by \( \eta_{\text{ad}} \)
- \( f \) = Jump Codeword No.

S.V.: \( \text{P}_\text{a} \)
- \( \text{c}_\text{b}, \text{W}_\text{b}, \text{P}_\text{b}, \text{T}_\text{b} \)
- \( V_{\text{b}+1} \)

B.D.: \( \lambda_b \) (dimensionless)
- \( \delta \text{H}(\text{CHU}/\text{lb}) \) and \( \eta_{\text{ad}} \) (dimensionless)
- \( \Delta \text{H}(\text{CHU}/\text{lb}) \)

E.V. Data: nil

E.V. Results: nil

Brick 35: Determination of Turbojet Off-Design Turbine Inlet Temperature

Description: Given (H.P.) compressor work \( \Delta \text{H} \), auxiliary work \( \delta \text{H} \), turbine inlet mass flow \( \text{W}_\text{a} \) and design-point value of \( - \Delta \text{h}/\text{T} \) for (H.P.) turbine, calculates turbine inlet temperature \( \text{T}_\text{b} \). If this is not equal to previously calculated value, Jump Exit occurs.

Form of Codeword: 35, 0, b, c, 0, e, f

S.V.: \( \text{W}_\text{b}, \text{T}_\text{b} \)

B.D.: \( \delta \text{H}(\text{CHU}/\text{lb}), - (\Delta \text{h}/\text{T})(\text{CHU}^\circ \text{K}) \)
E.V. Data : ΔH(CHU/lb)
E.V. Results : nil

Remarks: It is assumed that (H,P,) turbine operates between choked nozzles, and this Brick can therefore be used only when this assumption is valid.

Brick 36 : Determination of Off-Design Intake Mass Flow

Description: Given design-point propelling nozzle throat area $A_a$, alters inlet mass flow $W_0$ so as to give calculated $A_a$ equal to given value. If previous value of $A_a$ does not agree, Jump Exit occurs.

Form of Codeword: 36, a, b, c, 0, 0, f

S.V. : $W_0$

$A_a$

B.D. : $A_a$ (design)(ft²)

E.V. Data : nil
E.V. Results : nil

Brick 37 : Air-to-Air Heat Exchange (Cold Side)

Description: Given cold side inlet S.V. $S_a$, hot side inlet temperature $T_b$ (except on first entry) and values of $\lambda_{PC}$, $\Delta P_C$, $\eta_{th}$ and $\lambda_{PH}$, calculates cold side outlet S.V. $S_{a+1}$.

Form of Codeword: 37, a, b, c

S.V. : $\sigma_a$, $W_a$, $P_a$, $T_a$

$A_a$ (optional)

$T_b$ (not specified on first entry)

B.D. : $\lambda_{PC}$ (dimensionless), $\Delta P_C$(lbf/in²), $\eta_{th}$ (dimensionless - given as decimal fractions, not as percentage), $\lambda_{PH}$ (dimensionless)

E.V. Data : nil
E.V. Results : nil

Remarks: Note that cold side inlet and outlet planes must be numbered consecutively. $\lambda_{PH}$ is needed to determine whether or not this is the
first iteration: if not, it is made negative, the sign being restored by B38 after the last iteration.

**Brick 38: Air-to-Air Heat Exchanger (Hot Side)**

Description: Given cold side inlet and outlet S.V. $S_a$ and $S_{a+1}$, and hot side inlet S.V. $S_b$, also $\eta_{th}$, $\lambda_{PH}$ and $\Delta P_H$, calculates $S_{b+1}$ if $T_b$ equals value last used by Brick 37. Otherwise Jump Exit (to Brick 37)

Form of Codeword: 38, a, b, c, 0, 0, f

S.V. : $a_a$, $W_a$, $T_a$

$T_{a+1}$

$\sigma_b$, $W_b$, $P_b$, $T_b$

$A_b$ (optional)

B.D. : $\eta_{th}$ (dimensionless: given as decimal fraction, not as percentage),

$\lambda_{PH}$ (dimensionless), $\Delta P_H$ (lbf/in²)

E.V. Data : nil

E.V. Results : nil

Remarks: Note that hot side inlet and outlet planes must be numbered consecutively.

**Brick 39: Constant Volume Combustion**

Description: Given $S_a$, $T_b$ and combustion efficiency $\eta_b$, calculates $S_b$ and F for constant volume combustion.

Form of Codeword: 39, a, b, c, d

S.V. : $\sigma_a$, $W_a$, $P_a$, $T_a$

$A_a$ (optional)

$T_b$

B.D. : $\eta_b$ (dimensionless: given as decimal fraction, not as percentage)

E.V. Data : nil

E.V. Results : nil
Appendix 4 - Operating Instructions

1. The Turbocode Scheme is punched on three tapes as follows:

   (a) CAP 286/1 Functions, Subroutines, Codeword Input, Brick Input, Codeword Obey.

   (b) CAP 286/2 Bricks

   (c) RESTORE R600 Restores portions of Autocode overwritten by CAP 286/1

2. The normal method of operation is:

   (a) (If R600 is not already in the computer). Load Autocode Scheme (R600) on Reader A with handswitch 0 down. START/RUN.

   (b) At Z stop, STOP, load CAP 286/1 on reader A, START/RUN.

   (c) At "STOP" order, STOP, load Master Program on Reader A and CAP 286/2 on Reader B, START/RUN.

   (d) If all required Bricks have been read in, there will be a 77 stop in U4.6, and in general part of CAP 286/2 will not have passed through the tape reader. But if any Brick is missing, the 77 stop will be in U4.1, and the whole of CAP 286/2 will have been scanned: in this case, the missing Brick can be identified by comparing the printout of Bricks read with the Bricks specified in the Master Program, and the requisite tape can be read in via Reader B on STOP/RUN.

   (e) STOP, load Data Tape on Reader A, put handswitch 10 down to suppress optional printing of label numbers on jumps: also put handswitch 13 down if station vector printing is to be suppressed.

   (f) At conclusion of run, STOP, load RESTORE R600 on Reader A, START/RUN.

   (g) At Z stop, STOP.

3. If for any reason it is desired to restart a Turbocode program which has already been read, with its bricks, without reloading, it is necessary to have available a Re-entry Tape of the following form:

   J 1.2
   (STOP
   n1 = \theta
   \rightarrow M1760)

where \theta denotes the Codeword number at which it is required to restart.
This should normally be the number corresponding to Brick 31, thereby enabling the current S.V. and E.V. values to be printed, and S.V. to be reset, before reading new data.

This tape is used thus:-

(a) STOP, load Re-entry Tape on Reader A, clear all handswitches except 0, START/RUN.

(b) At "STOP" order, STOP, reload Data Tape on Reader A at point at which it is required to restart, set handswitches as described at 2 (e) above, RUN.

4. When it is required to run two or more Master Programs in the same session, time can be saved in reading the bricks tape (CAP/286/2) by proceeding as follows, provided that the bricks called for by the largest Master Program include all those required by the other Master Programs.

(a) Proceed as at 2(a) to 2(c) above, using the largest Master Program.

(b) If it is convenient to operate the largest Master Program first, carry on in the normal way, as at 2(d) above. If not or if, having run the largest Master Program, it is required to start one of the others, then:

(c) STOP, load Master Program tape concerned on Reader A, clear all handswitches except 0, remove any tape from Reader B, START/RUN.

(d) When control is transferred to Reader B (indicated by its motor starting up and by the "Input/output Busy" light on the console going on and staying on), STOP, load appropriate Re-entry Tape on Reader A and proceed as at 3(a) above.
Figure 1 - Processes for Simple Turbojet Design Point Calculation

Figure 2 - The Action of a typical Brick

Figure 3 - Station Vectors and Bricks for Specimen Master Program
**SIMPLE TURBOJET TEST**

**FIGURE 4a - Master Program coded ready for punching**
## TURBOCODE PROGRAMME SHEET A.

<table>
<thead>
<tr>
<th>Code Word</th>
<th>Data and Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Word No.</td>
<td>Brick No.</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>24</td>
</tr>
<tr>
<td>12</td>
<td>33</td>
</tr>
<tr>
<td>13</td>
<td>31</td>
</tr>
<tr>
<td>14</td>
<td>32</td>
</tr>
</tbody>
</table>

FIGURE 4b - Master Program (continued)
### Brick Data vs. Engine Vector

<table>
<thead>
<tr>
<th>No.</th>
<th>Item</th>
<th>No.</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\lambda_{W01}$</td>
<td>0</td>
<td>$\Delta H_{12}$</td>
</tr>
<tr>
<td>1</td>
<td>$\Delta W_{01}$</td>
<td>1</td>
<td>$F_{34}$</td>
</tr>
<tr>
<td>2</td>
<td>$\lambda_{P01}$</td>
<td>2</td>
<td>$\delta h_{45}/T_{4}$</td>
</tr>
<tr>
<td>3</td>
<td>$\Delta P_{01}$</td>
<td>3</td>
<td>$X_{G}$</td>
</tr>
<tr>
<td>4</td>
<td>$P_{2}/P_{1}$</td>
<td>4</td>
<td>$X_{R}$</td>
</tr>
<tr>
<td>5</td>
<td>$\eta_{pol\ 12}$</td>
<td>5</td>
<td>$X_{N}$</td>
</tr>
<tr>
<td>6</td>
<td>$\lambda_{W23}$</td>
<td>6</td>
<td>s.f.c.</td>
</tr>
<tr>
<td>7</td>
<td>$\Delta W_{23}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>$\lambda_{P23}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>$\Delta P_{23}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>$\eta_{b34}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>$\delta H_{45}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>$\eta_{ad45}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>$\lambda_{W56}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>$\Delta W_{56}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>$\lambda_{P56}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>$\Delta P_{56}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>$C_{D7}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>$C_{T7}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 5 - Brick Data and Engine Vector Lists**
SIMPLE TURBOJET TEST

J200.0

FIGURE 6 - Typical Turbocode Program

15 16 30 1,0,1 3,1,2,4 3,1,2 1,2,3,6 29,3,4,10,1 4,5,6,11,2 1,5,6,11 8,10,7,17,3 24,0,0,0,4 3,0,1,3,5,4 3,0,7 3,0,6,0,6 (0)

SIMPLE TURBOJET TEST S.V. DATA

FIGURE 7 - Data Tape for Typical Turbocode Program

+0 +0 +0 +0 +0 +1 +100 +0 +3 +7 +4 +288 +0 +6 +7116 +1 +6 +600 +2 +6 +400 +3 +6 +150 +4 +5 +1500 +5 +6 +800 +6 +6 +500

SIMPLE TURBOJET TEST BRICK DATA

+0 +1 +1 +0 +1 +0 2 +0,95 +3 +0 +4 +1 +6 +5 +0,9 +6 +1 +7 +5 +8 +0,95 +9 +0 +10 +1 +11 +0 +12 +0,9 +13 +1 +14 +1 +15 +0,95 +16 +0 +17 +0,99 +18 +0,99 -1
**7.168 AUTOCODE MK. 4**

**CAP286/1 TURBOCODE FUNCTIONS, ASSEMBLY AND SUBROUTINES, MK. 6 (KERO SINE)**

29/3/67 - 2

**SIMPLE TURBOJET TEST**

**CAP286/2 TURBOCODE BRICKS, MK. 6 (KERO SINE)**

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>404</td>
<td>16</td>
<td>421</td>
<td>31</td>
<td>446</td>
<td>32</td>
</tr>
<tr>
<td>01</td>
<td>402</td>
<td>02</td>
<td>473</td>
<td>03</td>
<td>488</td>
<td>04</td>
</tr>
<tr>
<td>29</td>
<td>516</td>
<td>06</td>
<td>572</td>
<td>34</td>
<td>567</td>
<td>30</td>
</tr>
<tr>
<td>33</td>
<td>617</td>
<td>619</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SIMPLE TURBOJET TEST S.V. DATA**

**SIMPLE TURBOJET TEST BRICK DATA**

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0800 +100.00 +4.70 +27.83</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +345.5 +1116.0 +1.706</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +409.52 +422.93</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +800.3 +807.0 +400.0 +0.3258</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +95.00 +399.97 +401.78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +806.1 +807.0 +1.050.0 +0.8511</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +398.26 +401.78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000 +806.1 +807.0 +1.050.0 +0.8511</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 8a - Output of Typical Turbocode Program (Station Vector Printout not inhibited)**

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>+1.1559 +4</td>
<td>+6.9507</td>
<td>+3</td>
<td>+7.9500</td>
<td>-2</td>
<td>+9.1317</td>
<td>+3</td>
</tr>
<tr>
<td>+3.4686</td>
<td>+3</td>
<td>+5.6631</td>
<td>+3</td>
<td>+1.3274</td>
<td>+0</td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 8b - Output of Typical Turbocode Program (Station Vector Printout inhibited)**

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>+1.1559 +4</td>
<td>+6.9507</td>
<td>+3</td>
<td>+7.9500</td>
<td>-2</td>
<td>+9.1317</td>
<td>+3</td>
</tr>
<tr>
<td>+3.4686</td>
<td>+3</td>
<td>+5.6631</td>
<td>+3</td>
<td>+1.3274</td>
<td>+0</td>
<td></td>
</tr>
</tbody>
</table>
Figure 9 - Flow Diagram for Brick 25
(Off-Design Convergent-Divergent Nozzle)