Version 1, August 2011. **CCS-N1X2**

Architecture of the English Electric DEUCE computer.

Contents.

1. Historical background to the English Electric DEUCE.

DEUCE was the production version of a research prototype called the NPL Pilot ACE, which was developed at the National Physical Laboratory, Teddington, between 1945 and 1950. It was called *Pilot* because it was seen at the time as a reduced-facility version of Alan Turing's original design for a larger computer named ACE, the *Automatic Computing Engine.*

Alan Turing OBE, FRS, (1912 – 1954), was a famous mathematician. He had published a classic paper entitled *On Computable Numbers, with an application to the Entscheidungsproblem* in 1936 whilst still a post-graduate student at the University of Cambridge – (see section N1X5, reference 1). *On Computable Numbers* was Turing's attempt to tackle one of the important philosophical and logical problems of the time: Is mathematics decidable? In order to reason about this so-called *Entscheidungsproblem*, Turing had the idea of using a conceptual, automatic, calculating device. The 'device' was a step-by-step process – more a thought-experiment, really – that manipulated symbols according to a small list of very basic instructions. The working storage and the input/output medium for the process was imagined to be an infinitely long paper tape that could be moved backwards and forwards past a sensing device.

It is now tempting to see Turing's mechanical process as a simple description of a modern computer. Whilst partly true, Turing's *Universal Machine* was much more than this: it was a logical tool for proving the decidability, or undecidability, of mathematical problems. As such, Turing's *Universal Machine* continues to be used as a conceptual reference by theoretical Computer Scientists to this day. Certainly, it embodies the idea of a *stored program*, making it clear that instructions are just a type of data and can be stored and manipulated in the same way.

After some brilliant war-time work on code-breaking at Bletchley Park, Alan Turing arrived at NPL on 1st October 1945. He was employed in the NPL Mathematics Division specifically to design an electronic universal computing machine. By the end of 1945, in the remarkably short time of three months, Turing had finished his first NPL report. It was entitled *Proposed electronic calculator* – (see section N1X5, reference 2). Historians now judge it to be the first substantially complete description of a practical stored-program computer. The typewritten document was very detailed, running to the modern equivalent of 83 printed pages including 25 pages of diagrams. It was what we would now call a *register-level* and *system-level* description, rather than a precise engineering design, though it did contain sample electronic circuits.

Alan Turing's 1945 report makes reference to another seminal document, written in America in June 1945 and entitled *First draft of a report on the EDVAC.* EDVAC (*Electronic Discrete Variable Automatic Computer)* was a widely-circulated proposal for a stored-program universal electronic computer, written by the Princeton mathematician John von Neumann – (see section N1X5, reference 3). Alan Turing's 1945 report at NPL uses the same basic notation and terminology as the EDVAC report. In the light of hindsight, we now judge von Neumann's report to be less complete and less general-purpose, placing more emphasis on a computer as a numerical calculator intended for scientific applications. In contrast, Turing's report described a more complex and more flexible machine, indicating a much wider range of applications.

Soon Alan Turing's proposed computer had been named ACE. Then over the period mid-1946 to mid-1947 more NPL staff were assigned to the ACE project. However, due to many internal NPL factors, the practical construction of the Pilot model of ACE did not get under way at NPL until early in 1949. Meanwhile in September 1947 Alan Turing, perhaps disillusioned with the lack of progress, went on a year's leave of absence to Cambridge University. He eventually resigned from NPL and joined the Mathematics Department of Manchester University in October 1948. In Turing's absence, Jim Wilkinson headed the Pilot ACE project at NPL. The Pilot ACE first ran a program on 10th May 1950 – (see section N1X5, reference 4).

Engineers from the English Electric Company had been seconded to the Pilot ACE project in 1949. It was thus natural that the company should take the basic design, make a number of minor improvements, and produce a commercially-available version. This was called DEUCE – (see section N1X5, reference 5). 33 DEUCE computers were delivered between 1955 and 1960, of which 12 remained within English Electric where they were put to work on a range of engineering problems and computing bureau activity. In this, they benefited from the numerical algorithms and software already developed by the Mathematics Division at NPL.

DEUCE went through various phases of upgrading. The successive models were called the Mark 0, Mark I, Mark II and Mark IIA. By September 1961 there were believed to be two Mark 0, 19 Mark 1, five Mark 2 and four Mark 2A DEUCE machines in service.

2. Architectural overview of Pilot ACE and DEUCE.

A general comparison of the original Pilot ACE and the original DEUCE is shown in Table 2.1.

	NPL Pilot ACE, as at May 1950	***English Electric DEUCE, as at 1955***
Word length, bits – *see note (a)*	32	32
Instruction length, bits	32	32
Instruction format	2 + 1	2 + 1
Instruction set: number of ops, *See note (b)*	Approx. 28	Approx. 30
Primary store size, words – *see note (d)*	128	402
Primary store type – *see note (c)*	Mercury delay lines	Mercury delay lines
Secondary store size, words	-	8K
Secondary store type	-	Drum
ADD time, min., millisecs. ADD time. max., millisecs.	0.064 1.064	0.064 1.064
MULTIPLY time, min., millisecs. MULTIPLY time, max., millisecs.	2 3	2 3
Digit period, microseconds	1	1
Main type of vacuum tube	ECC81 (12AT7) double triode	similar
Approx. number of vacuum tubes (including thermionic diodes)	Approx. 1000	1,450
Input medium	Card reader (CDR)	Card reader (CDR)
Output medium	Card punch (CDP)	Card punch (CDP); printer
Approx. cost of a production model	-	£42,000 - £50,000

Table 2.1. General comparison of the Pilot ACE and DEUCE computers.

Notes for Table 2.1.

(a) Word length: specific hardware support was given for double-length (64 bit) arithmetic – (see section N1X3).

(b) This is the number of effective distinct operations, as would be recognised by a modern programmer. The actual instruction lay-out would look strange to modern eyes – (see below). There was no single explicit accumulator, nor was there an index register (B register) for address-modification. For the English Electric DEUCE, a facility called Automatic Instruction Modification (AIM) was added in 1957 as an upgrade. Neither the Pilot ACE nor DEUCE had floating-point hardware.

(c) The choice of delay line for the primary memory affected instruction times in the following manner. When a computer obeyed instructions directly from a delay line store, access-time was affected by the address of the instruction or operand relative to the current position of information circulating in the store. That is to say, access to primary memory was *sequential* and not (as in the case of a modern computer) *random* as in RAM. Thus, minimum and maximum times are quoted in the Table for the ADD and MULTIPLY instructions. For both the Pilot ACE and DEUCE, the instruction format allowed the address of the next instruction to be specified. So-called *optimum programming* (or *minimum latency*) techniques were used to try and keep execution times as close to the minimum as possible.

(d) The first production version of DEUCE had 402 words of ‘quick-access’ storage, arranged as: 12 long lines of 32 words, 4 lines of 1, 3 lines of 2, 2 lines of 4 words. In DEUCE Mark IIA seven extra long lines were added, giving a total capacity increased from 402 to 626 words.

3. Tutorial on the Pilot ACE instruction set and architecture.

Since Alan Turing’s original notation and the design of both the Pilot ACE and DEUCE are difficult for the modern computer user to understand. It is helpful to give an explanatory tutorial. We first give the Pilot ACE’s instruction set in its original form, then derive a set of modern equivalents for the main facilities and finally produce a register-level diagram that is useful for an understanding of both ACE and DEUCE.

The format of a Pilot ACE instruction is as follows:

2	*3*	*5 bits*	*5 bits*	*2*	*5 bits*	*3*	*5 bits*	*1*	*1*
U	***NIS***	***Source***	***Destin***	***Ch***	***Wait***	***U***	***Timing***	***U***	***Go***
0	2 4	5 9	10 14		17 21	22 24	25 29	30	31

Least sig *most sig*

U:		unassigned
N	NIS:	next Instruction Source: indicates long delay line number (1 -> 7)
S	Source:	number of selected source
D	Destination:	number of selected destination
C	Characteristic:	gives length of transfer
W	Wait number:	gives first minor cycle of transfer
T	Timing number:	gives minor cycle of next instruction
G	Go digit:	If G = 0, wait for handkey to be pressed; if G = 1, full speed ahead.

A minor cycle is equivalent to a word, ie to 32 digit-periods. A digit-period is one microsecond. W and T are specified in terms of minor cycles. A description of their purpose is postponed to section N1X3.

To the modern programmer, it might at first sight appear that the *Source* and *Destination* digits in a Pilot ACE instruction simply specify the location of operands that take part in computational operations. The actual story is not so simple – if only because there seem to be no *op code* or *function* bits in the instruction to specify which operation is to be performed. In fact, some source-numbers and some destination-numbers have implicit functions associated with them.

The full story is that the *Source* and *Destination* digits are actually used for several purposes:

to specify one of eleven 32-word delay lines (DL) that form the main memory;
to specify one of five single-word delay lines, called *temporary stores (TS);*
to specify one of two double-length delay lines, called DS12 and DS14;
to specify an arithmetic or logical function;
to specify one of five hard-wired constants, called P1, P17, P32, zero and ones..

This picture is further complicated by the fact that the Pilot ACE design evolved over the period 1946 – 1949. The final notation and numbering system reflect the many changes that took place and may appear somewhat confusing.

To aid the modern programmer in an initial understanding the Pilot ACE, Table 2.2 given below re-interprets the original source and destination numbers in today’s terminology. The Table uses the following modern concepts and abbreviations:

Mn = main memory line n, ie the nth of eleven 32-word delay lines

Rn = central register n, ie the nth of five one-word delay lines
DRn = double-length register n, ie the nth of two two-word delay lines
Fn = the nth arithmetic or logical function – see below for more details
Cn = the nth wired-in constant. (The values, 0, -1, +1, 2^{16} and 2^{31} are provided).

Given the above modern abbreviations, we can derive some control-signal names that refer to them. From these signals we can then build up a simplified register-level diagram of the Pilot ACE. Signal-names beginning with ‘S’ in Table 2.2 control the data fed to the ‘source’ side of the highway in Figure 2.1. Signal-names beginning with ‘D’ control the data fed to the ‘destination’ side of the highway in Figure 2.1.Names in red are associated with operations that are described later.

Original Source number	***Modern signal name***	***Original Destination number***	***Modern signal name***	***Original Next-instruction number***	***Modern signal name***
0	**SF0**	0	Instruction register	0	NM11
1	SM1	1	DM1	1	NM1
2	SM2	2	DM2	2	NM2
3	SM3	3	DM3	3	NM3
4	SM4	4	DM4	4	NM4
5	SM5	5	DM5	5	NM5
6	SM6	6	DM6	6	NM6
7	SM7	7	DM7	7	NM7
8	SM8	8	DM8		
9	SM9	9	DM9		
10	SM10	10	DM10		
11	SM11	11	DM11		
12	**SDR1**	12	**DDR1**		
13	**SF1**	13	**DF8**		
14	**SDR2**	14	**DDR2**		
15	**SR1**	15	**DR1**		
16	**SR2**	16	**DR2**		
17	**SF2**	17	**DF9**		
18	**SF3**	18	**DF10**		
19	**SF4**	19	**DF11**		
20	**SR3**	20	**DR3**		
21	**SF5**	21	**DF12**		
22	**SF6**	22	*(unassigned)*		
23	SC1	23	**DF13**		
24	SC2	24	**DF14**		
25	SC3	25	**DF15**		
26	**SR4**	26	**DR4**		
27	**SR5**	27	**DR5**		
28	SC4	28	**DF16**		
29	SC5	29	**DF17**		
30	**SF7**	30	**DF18**		
31	(unassigned)	31	**DF19**		

Table 2.2. An interpretation of the Pilot ACE’s *Source* and *Destination* numbers as equivalent signals using modern terminology.

The implied notation of Table 2.2 is that SF0 – SF7 and DF 8 – DF19 are similar to *op codes* or *function specifiers.* The actions of these signals is further explained in Table 2.3.

Control signal name	***Approx. function***	***Description of action***
SF0	INPUT	Take a 32-bit row from the card reader as S
SF1	ARITH SHIFT RIGHT	Take DR2 / 2 (ie DR divided by 2) as S
SF2		Take R4 as S
SF3	ARITH SHIFT RIGHT	Take (R4 / 2) as S
SF4	ARITH SHIFT LEFT	Take (R4 x 2) as S
SF5	AND	Take (R4 & R5) as S
SF6	NEQ	Take (R4 NEQ R5) as S (ie exclusive OR)
SF7	TEST INPUT	S ≠ 0 when last row of a card is in position
DF8	ADD D	DR2 := DR2 + S
DF9	ADD	R2 := R2 + S
DF10	SUB	R2 := R2 - S
DF11	MPY	Multiply (independent of S). The product appears in DR2 (*see note 1*)
DF12	SWITCH R3 MODE	Set R3 either to be fed from M10 or to be treated as normal (see *note 2*)
DF13	SWITCH DR2 MODE	Set DR2 either as two single-length registers or one double-length register (see *note 2*)
DF14	JLT	Branch on sign of S (*see note 3*)
DF15	JNZ	Branch on whether S is zero (*see note 3*)
DF16	OUTPUT	Transfer a 32-digit row to a card (ie the card punch)
DF17	HOOT	Send a pulse to the console hooter
DF18	PREPARE CDP	Commence punching a card (see note 4)
DF19	PREPARE CDR	Commence reading a card (independent of S)

Table 2.3. Description of the arithmetic and logical functions controlled by SF0 to SF7 and DF8 – DF19 of Table 2.2.

Notes on the functions in Table 2.3.

Note 1: Prior to the multiplication of two quantities p, q, p must be sent to R3, q must be sent to the odd half of DR2 and zero sent to the even half of DR2. When this has been done, any instruction with destination D = 19 will activate the multiplication process. The double-length answer appears in DR2. However, if one or both p, q are negative, a correction must be built up in R2 by the programmer after activating the multiplication process (and therefore overlapped in time with the multiplication process).

Note 2: F12 and F13, which allow a programmer to modify the respective working modes of registers R3 and DR2, also use the value specified by the *characteristic* bits in the current instruction. If, for F12, the characteristic is an *odd* number then R3 ceases to have an independent existence and is fed continuously from M10. This means that Source 20 then gives the contents of M10 one word later than from Source 10. R3 only returns to its normal existence if F12 is issued with an *even* characteristic. For F13, an *odd* characteristic causes DR2 to behave as if it were two single-length registers. This mode persists until F12 is issued with an *even* characteristic.

Note 3: The two possible destination instructions must be located in consecutive addresses in the main memory. For the JLT function, the first-occurring instruction is taken if the value of the source-operand is positive or zero; the second-occurring instruction is taken if the value is negative. For the JNZ function, the first-occurring or second-occurring instruction is taken depending upon whether the value of S is zero or non-zero.

Note 4: The card reader (CDR) is activated to read a card by the F19 function, following which the function F0 is used to transfer each row, where a row is regarded as a 32-bit number. The card punch (CDP) is activated in readiness to punch a card by the F18 function, following which the function F16 is used to transfer each row (regarded as a 32-bit number) to the device. Both F18 and F19 are independent of the value of S. Both F0 and F16 are always used with the GO digit = 0, normally signifying that the computer will halt until a manual switch has been pressed on the console. However, with F0 and F16 the CDR and CDP effectively take the place of the manual switch, such that the computer halts during F0 and F16 until the respective input/output device is ready to transfer a row of information. In this manner, the high-speed computer is made to keep pace with the relatively slow input/output activity at appropriate points in a program.

Given the signal-names of Table 2.2, we can now draw a register-level diagram of the Pilot ACE which shows the main data highways for a sub-set of the total facilities.

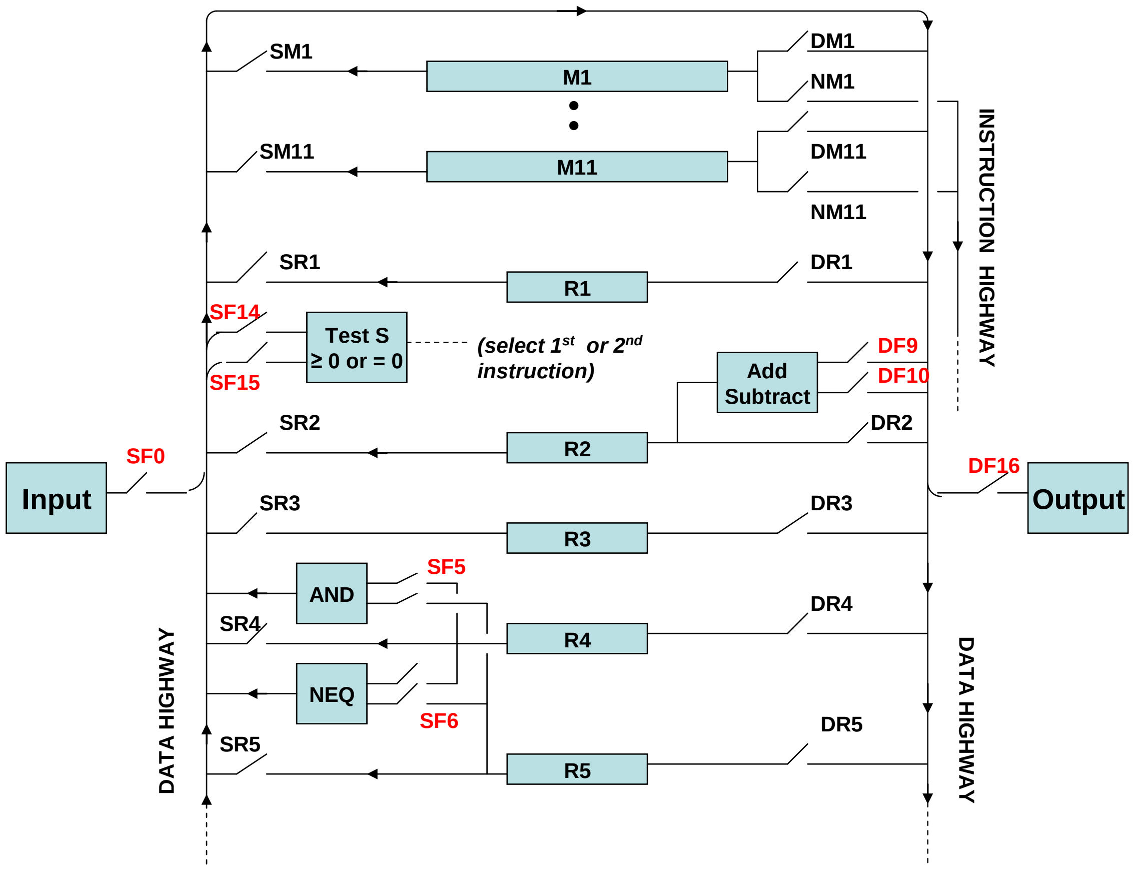


Figure 2.1. Simplified diagram of the Pilot ACE, using the control signals defined in Table 2.1. The diagram shows the main data highways and a small sub-set of the Pilot ACE's functionality.

The register-level diagram of DEUCE is similar to that shown above, except that the signal-notation has been rationalised. The DEUCE facilities are described more fully in section N1X3.

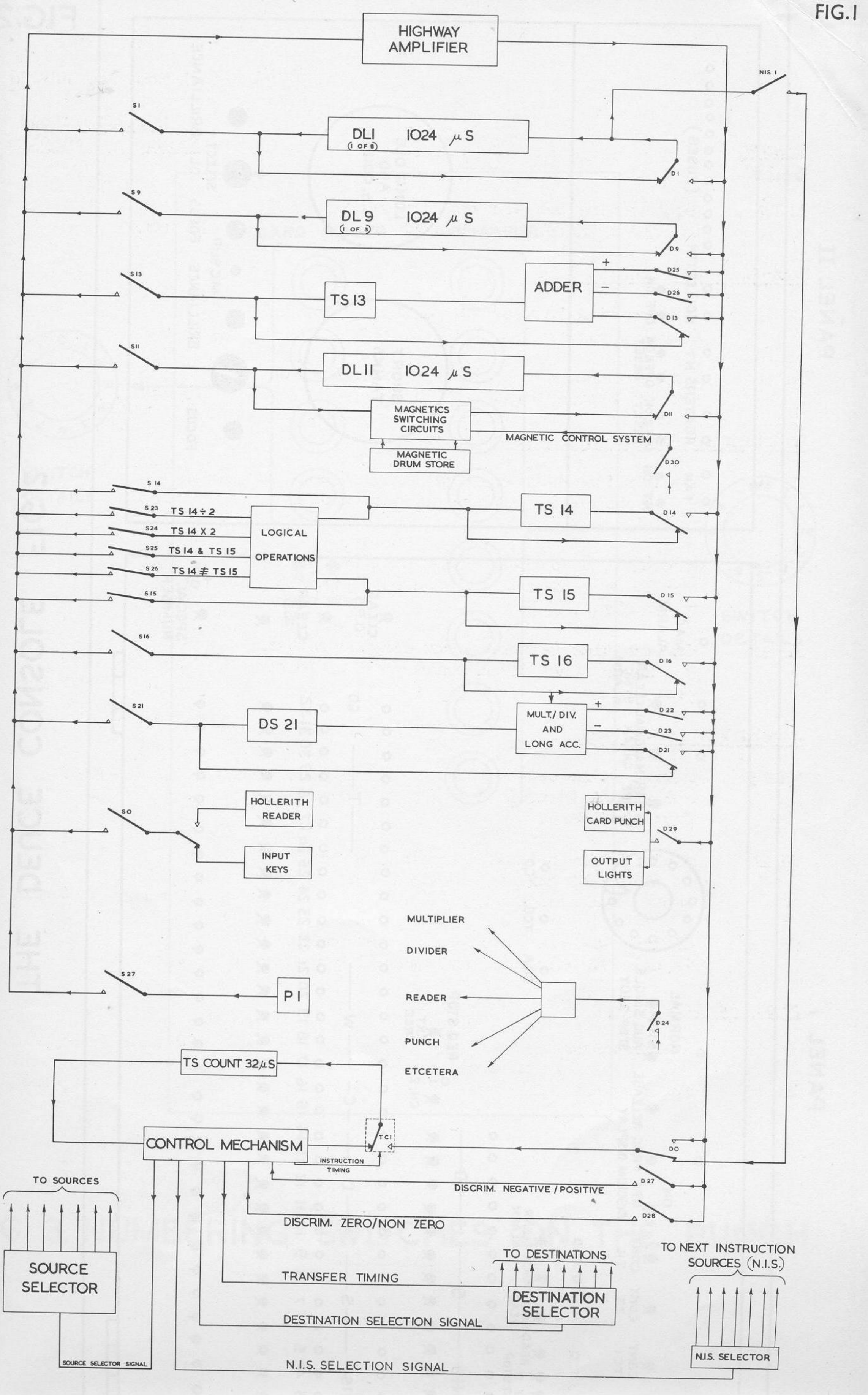
4. System diagram of DEUCE.
An original DEUCE register-level diagram, taken from reference 7 (see section N1/X5), is given on the next page. Note that the signal names on the switches are the English Electric ones rather than modern names. For a fuller understanding of these original signal-names, refer to Table 3.1 in section N1X3. Note also that the diagram on the following page is not intended to be precise. Rather, it simply gives an indication of how data is routed along the DEUCE internal highways during various DEUCE instructions.

…. (continued) …

MS. 3052

T.N. MS. 38

FIG. 1


HIGHWAY AMPLIFIER
NIS 1
S1
DL1 (1 OF 8) 1024 μS
D1
S9
DL9 (1 OF 8) 1024 μS
D9
S13
TS 13
ADDER
D25
D26
D13
S11
DL11 1024 μS
D11
MAGNETICS SWITCHING CIRCUITS
MAGNETIC DRUM STORE
MAGNETIC CONTROL SYSTEM
D30
S14
S23 TS 14 ÷ 2
S24 TS 14 X 2
S25 TS 14 & TS 15
S26 TS 14 ≢ TS 15
S15
LOGICAL OPERATIONS
TS 14
D14
TS 15
D15
S16
TS 16
D16
S21
DS 21
MULT./DIV. AND LONG ACC.
D22
D23
D21
S0
HOLLERITH READER
INPUT KEYS
HOLLERITH CARD PUNCH
OUTPUT LIGHTS
D29
MULTIPLIER
DIVIDER
READER
PUNCH
ETCETERA
S27
P1
D24
TS COUNT 32μS
CONTROL MECHANISM
TC1
INSTRUCTION TIMING
D0
D27
D28
DISCRIM. NEGATIVE / POSITIVE
DISCRIM. ZERO/NON ZERO
TO SOURCES
SOURCE SELECTOR
SOURCE SELECTOR SIGNAL
TRANSFER TIMING
TO DESTINATIONS
DESTINATION SELECTOR
DESTINATION SELECTION SIGNAL
TO NEXT INSTRUCTION SOURCES (N.I.S.)
N.I.S. SELECTOR
N.I.S. SELECTION SIGNAL