Version 2, October 2011. **CCS-F6X2**

Systems Architecture of the Ferranti Poseidon, Hermes, Apollo and Argus computers.

1. Ferranti Poseidon.

Since this computer was intended for use by the Royal Navy, it was initially designated NEPTUNE, a pertinent acronym for *Nutter's Easily Programmable Type for Use on Naval Exercises*. Geoff Nutter was the logical designer. However, the name Neptune had already been applied to the nuclear reactor in the RN submarine programme, so the Greek equivalent for the god of the sea was chosen. Poseidon was designed for real-time Royal Naval applications such as radar data processing, radar and weapon platform stabilisation, weapon firing control and organising the display of tracks and tactical data on Operations Room screens. In that context, to maximise the benefit that could be provided by the speed available in the technology of the day, the Instruction Set (or Order Code) was strongly biassed towards the provision of powerful bit-handling facilities.

There were two Instruction Sets - called the Inner Order Code and the Outer Order Code, both of 24-bit words. The basic logic of the machine implemented the 'function-rich' three-address Inner Order Code. It had nine function bits, conveniently split up as three octal digits, U, V and W. The remaining 15 bits were generally used as three 5-bit address fields. The 32 possible addresses were generally allocated as follows, where *v* with a qualifier denotes the contents of the registers in this kernel:-

*v*0	Zero
*v*1 - *v*23	General Registers, some with special meanings.
*v*24	Order Counter.

There were also a number of other registers (compare the somewhat similar arrangement for Hermes:-

N, the Data Store address register,
L, the sub-routine Link Address register
lambda, a counter register
alpha, beta, ??
kappa, a register used to hold a collate filter.
mu, ??

The symbols *v* and *n* were available on the teleprinters used to prepare program tapes. These teleprinters had the same pedigree as those used for the Ferranti Pegasus computer, particularly for preparing Pegasus Autocode programs. The main operations on the three addresses could be supplemented in the same operation time by other operations on subsidiary registers, allowing, for example, incrementing the store address registers or counting.

Orders could be stored in the core store (generally the case during program development) or in a separate *fixed* store – 'fixed' in an attempt to ensure that the program could not be lost in the case of machine or power supply malfunction, a real potential hazard in a ship on active service! The Fixed Store in Operational Systems consisted of small copper rings threaded by wires with adjacent ‘cards’ of magnetic material punched with a hole for a 1 and unpunched for a 0. These cards were removable for different Operational Program functions or updates. Both fixed and ‘variable’ (data) store were in blocks of 4096 words.

The Outer Order Code was of a conventional type, with two octal function characters, two modifier bits and the rest Store Address bits. The first 256 locations of the fixed store were used to hold the Inner Orders that implemented the Outer Code, as well as other basic input and set-up routines.

Order Code facilities were available to control inter-computer transfers (systems were delivered to the Navy with up to three connected computers). Not all combinations of UVW A B C were meaningful - and undefined combinations were not detected. They might thus do something without warning – probably not what was required, so the programmer had to be extra careful!

2. Ferranti Hermes

The Hermes three-address Instruction Set (Order Code) was an improved version of that used in Poseidon. Improvements included adopting a more logical allocation of functions, so that the amount of order decoding logic, and thus the machine size, was considerably reduced. There was only one Order Code (no Outer Code), since experience with the two Order Codes of Poseidon had showed that maximum benefit from use of the machine in its intended applications was obtained from use of the Inner Order Code only.

Hermes had a 24-bit word, the bits being treated in parallel (not serial), and a clock-rate of 1 MHz. It employed germanium *NOR* logic. The basic model had 4K words of core store and the whole machine occupied a single cabinet measuring 6 ft high, 2 ft wide and 1.5 ft deep. A basic system cost about £35,000 in October 1963. The core store could be incremented in modules of 4K. Two speeds were provided, with cycle times respectively of 3 and 6 microseconds. Simple arithmetic operations (eg ADD) were executed in four store cycle-times, ie in 12 or 24 microseconds respectively. Multiplication took 24 – 30 microseconds or 36 – 44 microseconds respectively. Ampex half-inch magnetic tape decks could be added to a Hermes system, via a Magnetic Tape Control Unit. Similarly, magnetic drums could be added via a Magnetic drum Control Unit. A very large Hermes system might cost up to about £900,000.

The layout of an instruction was as follows:

3	2	1	3	5	5	5
F	I	D	S	A	B	C

F field: 3 bits defining the main functions (op codes)
I field: 2 bits that either define the Index Register to be referred to (F=0 to 3), or determines th etreatment of the double-length accumulator in multiplication, division and shift orders. If I = 0, a jump instruction is specified.
D field: if D = 0, treat C as a number; if D = 1, treat D as an address.
S field: the subsidiary function bits (op codes).

A, B, C are'kernel store' registers or input/output devices – (see also section F6X3 for more details).

Each of A, B and C allowed reference to 32 possible addresses, which were generally allocated as follows, where *v* with a qualifier denotes the contents of the registers in this kernel:-

*v*0	Zero
*v*1 - *v*23	General Purpose Registers, the so-called 'kernel' store.

	In A	***In B***	***In C***
*v*24	Tape Punch	*n*0 (Order counter)	Handswitches
*v*25	Other o/p	*n*1	Tape reader
*v*26	Other o/p	*n*2	Other i/p
*v*27	*n*3	*n*3	Other o/p

*v*28	*vn*0, order being implemented.
*v*29	*vn*1, contents of store at addresses in the Index Register *n*1
*v*30	*vn*2, contents of store at addresses in the Index Register *n*2
*v*31	*vn*3, contents of store at addresses in the Index Register *n*3.

The symbols *v* and *n* used were available on the teleprinters used to prepare program tapes. The main operations on the three addresses could be supplemented in the same operation time by other operations on subsidiary registers, allowing, for example, incrementing the store address registers or counting.

Access to a location in the main 4K memory is, in the first instance, via an address held in one of the three Index registers. Hermes also had a facility called a *Link Nest Pointer* N_L, which shared its physical address with Index Register N3. The setting of a one-bit flag called Q determined whether a programmer was referring to N3 or to N_L. The Link Nest Pointer is typically used on entering subroutines. The Nesting Store itself is just an area of main store previously set up by the programmer. Any number of Nesting Stores up to a maximum limit of 512 words can be established.

3. Ferranti Apollo.

Apollo had a 20-bit word length with a conventional Instruction Set with six function bits, two modifier and 12 address bits, allowing access to the 4096 word core-store. This computer was designed to produce Flight Progress Strips for Air Traffic Controllers to use for directing traffic over the North Atlantic.

4. Ferranti Argus Series.

Each Argus instruction occupied one location or word of 24 binary digits ('bits'), together with a parity bit that was invisible to programmers. The normal word length for data in the computer was 12 bits and so one order location of 24 bits could also be used to hold two 12-bit variables. The arrangement of the store was such that the first 2048 words of store could be used for either program instructions or for variables, and the last 2048 words could be used for program instructions only. In computers having 2048 words of store or less, instructions or variables could be held in any part of the store. This method provided a very flexible arrangement, giving equal facility to programs requiring large or small numbers of variables or constants.

Numbers in the computer normally occupied 12 binary digits, one of which was a sign digit, giving an accuracy of 1 part in 2000. For most control applications this was sufficient, but in some cases a higher accuracy was required and the computer had been designed so that operations on 12 or 24 bit numbers could be carried out with equal ease. One 24-bit number occupied the space of two 12-bit numbers. There was one sign bit and the remaining 23 bits formed the number. Numbers of 12 bits were referred to as 'short' numbers; numbers of 24 bits were referred to as 'long' numbers.

The computer had six accumulators numbered from 1 to 6, together with accumulator 0, which was a null accumulator, that is it always contained zero. When arithmetic was being carried out on short numbers then accumulators 1 to 5 were specified; when long numbers were being used, accumulator 6, a long accumulator was specified. Accumulators 4 and 5 combined together could also be used as a long accumulator which was addressed as accumulator 7.

The Argus Instruction Set was derived from that of the Ferranti Pegasus computer. Ignoring the parity bit for the moment, each Argus instruction was built up as follows:-

13	3	6	2
N	X	F	M

In the original (Bloodhound) version of Argus, instructions were set up in binary form on a pegboard, together with any constants required during a calculation. The parity digit was arranged to give an odd number of bits or pegs in the word. Each instruction was checked before it was obeyed to ensure that its parity was odd.

Generally the computer itself was held in one cabinet, with an adjacent cabinet holding racks of equipment to convert input from and output to a wide range of external devices depending upon the particular application. Analogue inputs from rotating shafts, voltages, pressures, temperatures, etc. were converted to digital form for processing in the computer. Digital signals could be output to operate switches or converted to analogue form for direct use for control purposes.