6809 evaluation system for £100

Uprated Nanocomp and cassette interface

by R. Coates

The 6809 is a recent 8-bit microprocessor which uses a 16-bit architecture to considerably improve the performance available from an 8-bit device. Because development of conventional 16-bit processors is accelerating, many designers think that the 6809 represents the practical limit for an 8-bit device. Unfortunately, few potential users have been able to evaluate this processor because there is very little hardware available at present and information is still scarce. This design is based on the well-tried and tested Nanocomp (Jan. 81) and provides a useful low-cost evaluation system for the 6809.

The 6809 is the most recent addition to the M6800 family of microprocessors, and provides a much more advanced architecture than the 6802. Internally the device is a 16-bit processor, which can perform 16-bit operations, with several extra registers and other improvements. However, because the device retains an 8-bit external data bus, the hardware is very similar to the 6802 and can therefore be used with a slightly modified Nanocomp.

The improved performance of the 6809 is attributable to several factors besides the potential of 16-bit operations. An important advantage is the addition of extra and more powerful addressing modes which enable the processor to recognize 1464 different variations of instructions and addressing modes from a basic instruction set of 59. Despite this large number of instructions, the improved architecture makes the device easier to program.

To preserve software compatibility with earlier Motorola microprocessors, the 6809 is compatible at source-code level with the 6800 so all but a few of the existing mnemonics are included. Exceptions such as INX have been excluded to maintain as rigidly as possible the regularity of the architecture. Extra addressing modes have been provided for the existing instructions and new instructions, unique to the 6809, have been added. Therefore, source programs written for the 6800 can be re-assembled using the 6809 op-codes, (not all are the same as the 6800) and existing software can be transferred. All mnemonics excluded from the 6809 can be performed by new instructions. Although it may seem pointless to transfer existing software to a more powerful processor, it allows users to upgrade their systems without having to re-learn completely.

Table 1. Revised memory map

eprom		7FFF
	monitor	
		7C00
	user	7800
	eprom	7400
		7000
		4003
	pia	
		4000
2114 rams		13FF
	display buffer	
		13FA
	monitor workspace	
		13E0
		13D0
	monitor stack	
		13B0
	user stack	
	program ram	
		1000

pia
4000 output/data direction register A
4001 output/data direction register B
4002 control register A
4003 control register B

Branch instructions have been improved by adding 16-bit 2's complement offsets as well as 8-bit. This permits a branch to be made from anywhere to anywhere in the full 64K address range, which makes the writing of position independent programs much easier.

Circuit modifications

The block diagram of this design is identical to the original version except that the 6809 does not have an on-chip r.a.m. and the 128 bytes at address 0000 to 007F are not available. The circuit diagram in Fig. 1 is almost identical to the original and, apart from the obvious change of c.p.u. chip, the main difference is that the 74LS00, which generated the VMA.E signal, is omitted. This i.c. is not required because there is no valid memory address signal on the 6809, and invalid memory addresses are forced to FFFF. The E output can be used directly in place of VMA.E. Another alteration is the provision of an extra interrupt input, the fast interrupt request FIRQ. This input is not used in the present design, but is brought out to a pin for possible future use. Reset on the 6809 has a Schmitt-trigger input which, with capacitor C_9, provides automatic power-on reset. Because the c.p.u. on-chip r.a.m. is not available, the memory map has been revised and is shown in Table 1. The monitor workspace is now positioned at the top of the 1K memory and therefore about 40 bytes are lost for user programs. All other aspects of the circuit and testing are as described for the 6802 version.

Operation

Operation is more or less the same as the 6802 version. As the monitor software listing now includes cassette-tape handling routines, the full 1K allocated to the monitor program, 7C00-7FFF, is now used. These routines use the L and P keys and are described later. The main alteration to the monitor is the register display command R which has been revised to take account of the increased number of c.p.u. registers. This command is automatically entered after a SWI, but may be re-entered with the R key. The condition-code register contents are first displayed with the right-hand digit denoting the register being displayed as shown,

C	=	condition-code register
A	=	acc A
b	=	acc B
d	=	direct page register
H	=	X register (index)
Y	=	Y register
U	=	user stack pointer
P	=	program counter
S	=	hardware stack pointer

The I key will increment through the various registers, and their contents will be shown on the left four digits. After displaying S, the unit automatically returns to the monitor start. The two new software interrupt instructions, SW12 and SW13, are not used by the monitor but, with the hardware interrupts, the program can jump to and continue from a specified address in certain memory locations. These are listed in Table 2.

When an interrupt occurs, the continuation address is fetched, which is usually the interrupt service routine, and proces-

Table 2. Interrupt jump locations

When an interrupt occurs, an address is fetched from the memory shown and processing continues from that address.

Interrupt	Address
SW13	13E5/6
SW12	13E7/8
FIRQ	13E9/A
NMI	13F2/3
IRQ	13F4/5

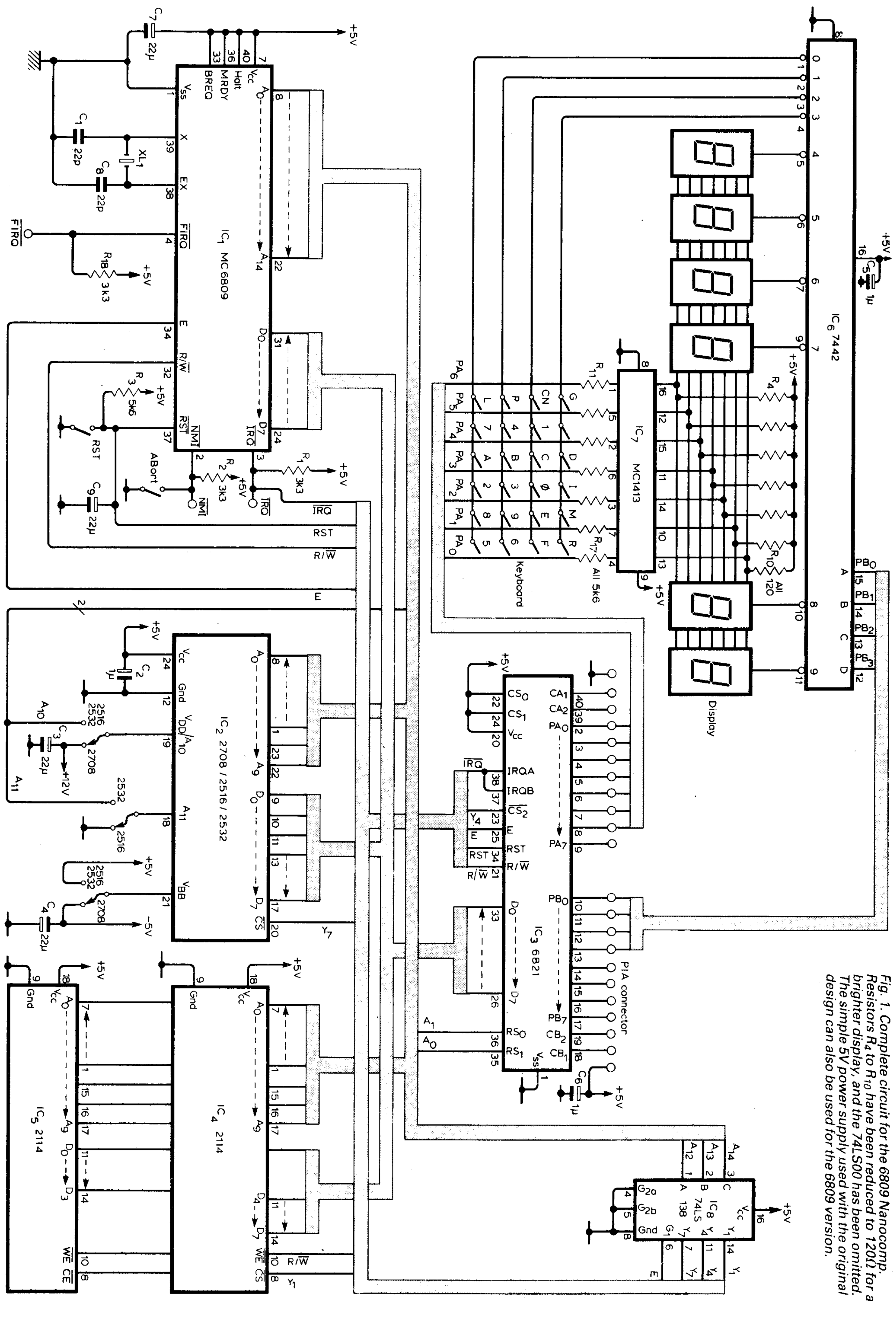


Fig. 1. Complete circuit for the 6809 Nanocomp. Resistors R_4 to R_{10} have been reduced to 120Ω for a brighter display, and the 74LS00 has been omitted. The simple 5V power supply used with the original design can also be used for the 6809 version.

sing continues from that point. The NMI input, however, is used for the abort key and its jump address is set automatically when a reset occurs, but it may be modified for other purposes by a users program. The monitor has been written to ensure that the useful monitor subroutines, listed in Table 2 of the original article, function identically and have the same entry address points. The re-entry point to the monitor from a user program is 7D97, which also applies to the 6802 version.

The four original programs can be included if a 2 or 4K e.p.r.o.m. is used. The start address for hex-decimal/decimal-hex converter is 7800, duckshoot – 7940, branch calculator – 7A00 and mastermind –7A80. For duckshoot, the speed is set at location 1000 and 1 because there is no memory at 0000 in the 6809 version. Two's complement offsets used by the branch instructions of the 6802 are limited to 8 bits but the 6809 also uses 16-bit offsets, with the PC relative addressing mode, therefore the branch calculator program now caters for these. In addition to requesting the start and destination addresses, the program requests the number of bytes in the instruction, b in the right-hand display, which must be entered. If an instruction has only two bytes, it must be an 8-bit offset so an 8-bit value is given or two dashes if it is out of range.

An instruction which requires a 16-bit offset must be three bytes or more, so a 16-bit answer is displayed if a byte value of three or greater is given.

Programming

Because programming information for the 6809 is not widely available yet, a brief description of the architecture is given together with the instruction set and programming details. However, for serious programming, Motorola's MC6809 Preliminary Programming Manual is essential.

A programming model of the 6809 is shown in Table 3, and details of the registers are given below.

Accumulators (A, B, D)
The A and B registers are general purpose 8-bit accumulators for arithmetic calculations and data manipulation. Some instructions link the registers to form a single 16-bit accumulator (D) with A as the most significant byte.

Direct page register
The direct addressing mode in the 6800 allows a shorter form of instruction to be used for accessing the bottom 256 bytes of memory. This facility has been enhanced in the 6809 so that the 8-bit direct page register is used as the most significant byte for direct addressing. This allows the direct mode to be used under program control at any place in memory.

Index registers (X, Y)
These are the same as the single 6800 register. The 16-bit address in the register takes part in the calculation of effective addresses and can be used to point to data directly. The address can also be modified by an optional constant or register offset. The 8-bit constant offsets are supplemented with 5 and 16-bit offsets. All four pointer registers (X, Y, U, S) can be used as index registers.

Stack pointers (U, S)
The hardware stack pointer, S, is used by the processor during subroutine calls and interrupts, and points to the top of the stack instead of the next free location as in the 6800.

The user stack pointer, U, allows arguments to be passed to and from subroutines. Both stack pointers can also be used as index registers, and have additional Push and Pull instructions which can operate on any or all of the registers (except themselves).

Program counter
Used by the processor to point to the address of the next instruction to be executed.

Condition code register
This register, also known as the flag register, defines the state of the processor at any time. The register comprises
C (bit 0) CARRY. Indicates that a carry occurred on the last ALU operation, or a borrow from subtraction instructions.
V (bit 1) OVERFLOW. Set by an operation which causes a two's complement arithmetic overflow.
Z (bit 2) ZERO. Set if the result of the previous operation was zero.
N (bit 3) NEGATIVE. Contains the m.s.b. from the result of the preceeding operation. Therefore, a negative two's complement will leave N set.
I (bit 4) IRQ mask bit. Interrupts on this line will not be recognised if I is set. NMI, FIRQ, IRQ, RESET and SWI all set the I bit, but SW12 and SW13 do not affect it.
H (bit 5) HALF CARRY. Indicates a carry from bit 3 in the ALU after an 8-bit addition. Used by the decimal adjust instruction to perform the b.c.d. decimal add adjust operation.
F (bit 6) FIRQ mask bit. Interrupts on this line will not be recognised if I bit is set. NMI, FIRQ, SWI and RESET set the F bit. IRQ, SWI and SW13 do not affect it.
E (bit 7) ENTIRE FLAG. Used to indicate whether all the registers were stacked or a subset (PC and CC) performed by a FIRQ. The E bit of the stacked condition code register is used on return from interrupt (RTI) to determine the extent of unstacking.

The main improvement offered by the 6809 is the proliferation of addressing modes which are summerised below.

INHERENT. In this mode the opcode contains all necessary address information (single byte instruction).

IMMEDIATE. The data to be used by the instruction immediately follows the opcode in memory. Can be an 8-bit or 16-bit value depending on the instruction.

EXTENDED. The contents of the two bytes following the opcode specify the 16-bit effective address used by the instruction.

EXTENDED INDIRECT. A special case of indexed addressing where one level of indirection is added to extended addressing, ie, the two bytes following the postbyte of an indexed instruction contain the address of the address of the data.

DIRECT. Similar to extended but only the lower 8 bits of the effective address are specified in the byte following the opcode. The upper 8 bits of the effective address are supplied by the direct page register. Therefore, programs using this mode rather than extended will use less memory.

INDEXED. The most complex addressing mode. In all indexed addressing, one of the pointer registers (X, Y, U, S and sometimes PC) is used in a calculation of the instruction. The postbyte of an indexed instruction specifies the basic type and variation of addressing mode, and the pointer register to be used. Table 4 gives the details necessary for calculating the postbyte opcode for all forms of indexed addressing. The five basic types of indexing are
Zero Offset. The selected pointer register contains the effective address of the data to be used by the instruction.
Constant Offset. A two's complement offset and the contents of one of the pointer registers are added to produce the effective address of the operand. The pointer register's initial content is unchanged by the addition.

Three sizes of offset are available, ± 4-bit (–16 to +15), ± 7-bit (–128 to +127) and ± 15-bit (–32768 to +32767). The 5-bit offset is included in the postbyte, whereas 8-bit and 16-bit offsets require 1

Table 3. Programming model of the 6809

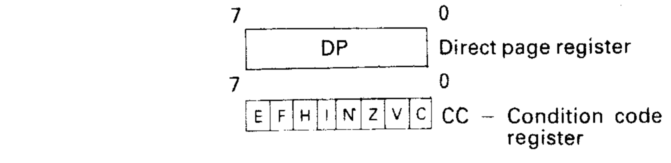


Table 4. Indexed addressing modes

		Non indirect				Indirect			
Type	Forms	Assembler form	Postbyte op-code	+	+ #	Assembler form	Postbyte op-code	+	+ #
Constant offset from R (signed offsets)	no offset	R	1RR00100	0	0	[R]	1RR10100	3	0
	5-bit offset	n, R	0RRnnnnn	1	0	defaults to 8-bit			
	8-bit offset	n, R	1RR01000	1	1	[n, R]	1RR11000	4	1
	16-bit offset	n, R	1RR01001	4	2	[n, R]	1RR11001	7	2
Accumulator offset from R (signed offsets)	A – register offset	A, R	1RR00110	1	0	[A, R]	1RR10110	4	0
	B – register offset	B, R	1RR00101	1	0	[B, R]	1RR10101	4	0
	D – register offset	D, R	1RR01011	4	0	[D, R]	1RR11011	7	0
Auto increment decrement R	increment by 1	,R+	1RR00000	2	0	not allowed			
	increment by 2	,R++	1RR00001	3	0	[,R++]	1RR10001	6	0
	decrement by 1	,–R	1RR00010	2	0	not allowed			
	decrement by 2	,– –R	1RR00011	3	0	[,– –R]	1RR10011	6	0
Constant offset from PC	8-bit offset	n, PCR	1XX01100	1	1	[n, PCR]	1XX11100	4	1
	16-bit offset	n, PCR	1XX01101	5	2	[n, PCR]	1XX11101	8	2
Extended indirect	16-bit address	–	–	–	–	[n]	10011111	5	2

R = X, Y, U or S X = 00 Y = 01 X = don't care U = 10 S = 11
+ and +# indicate the number of additional cycles and bytes for the particular variation

or 2 bytes respectively after the postbyte for the offset.
Accumulator Offset. Similar to constant offset indexed except that the two's complement value in one of the accumulators (A, B or D) is used as the offset, the postbyte specifies which. Neither register is altered by the operation.
Auto Increment/Decrement. Similar to zero offset, but with auto increment. After the pointer register is used it is incremented by 1 or 2 and then used.
Indexed Indirect. All indexing modes, except auto increment/decrement-by-one and 5-bit offset, can have an additional level of indirection. This means that the effective address is contained at the location specified by the content of the index register plus any offset.
RELATIVE. Branch instructions use the relative addressing mode, i.e. the byte(s) following the branch opcode is a signed offset which is added to the program counter. If the branch condition is true, the calculated address (PC + signed offset) is loaded into the program counter. Execution then continues from the new address. Short branches require a 1-byte offset and long branches require 2 bytes.
PROGRAM COUNTER RELATIVE. Another type of indexed addressing where the program counter is used as the pointer, with an 8 or 16-bit offset. This is very useful for pointing to blocks of data in a program which must be relocatable, i.e. runs anywhere in memory. The Load Effective Address instruction makes use of this mode. For example, to point the X register to a block of data by specifying an offset, relative to the current PC position, where the data block resides. This offset will remain constant wherever the program is run, whereas with a LDX instruction the absolute address must be specified. An additional level of indirection is available with this mode.

New instructions

PSH/PUL. These instructions allow any combination of registers to be pushed onto or pulled off the hardware (S) or user (U) stack. Which registers are pushed or pulled is defined by an immediate byte after the opcode. Each bit in the byte specifies a register.

C C	=	bit 0
A,D	=	bit 1
B,D	=	bit 2
D P	=	bit 3
X	=	bit 4
Y	=	bit 5
U,S	=	bit 6
PC	=	bit 7

TFR/EXG. Any register may be transferred or exchanged with any other register of the same size, i.e. 8-bit to 8-bit or 16-bit to 16-bit. Also, a 16-bit register can be transferred to an 8-bit. The registers to be

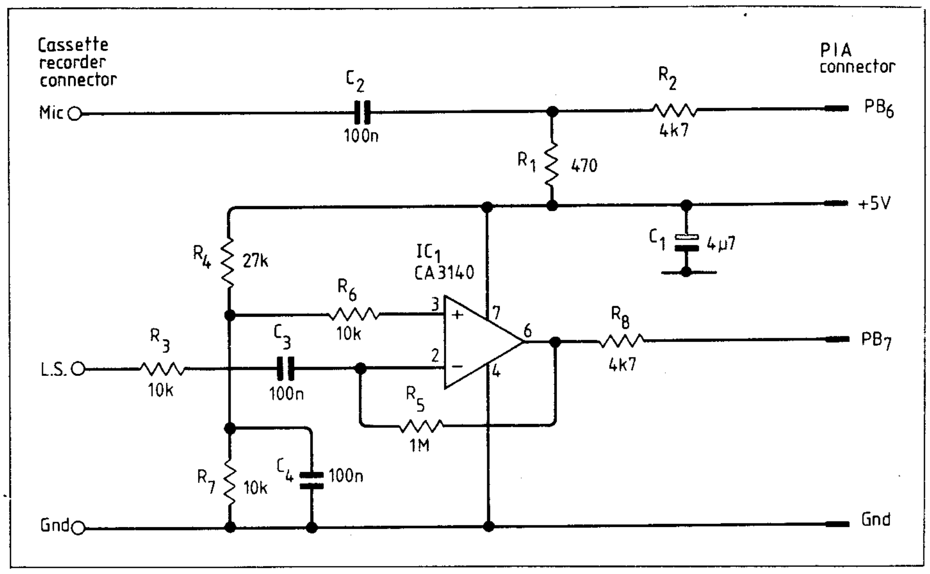
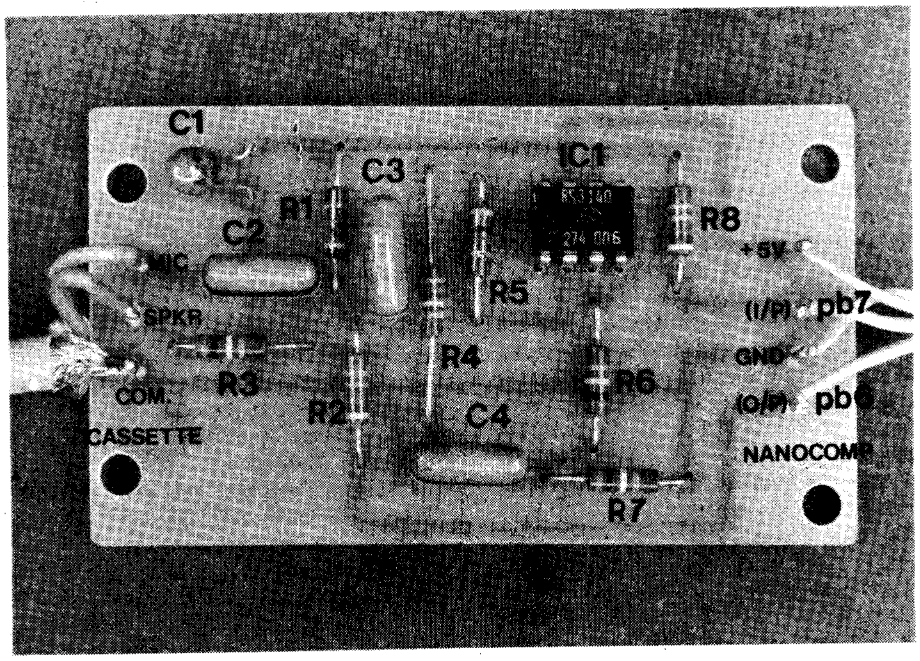


Fig.2. Cassette interface. This circuit is powered from the Nanocomp via the p.i.a. connector.



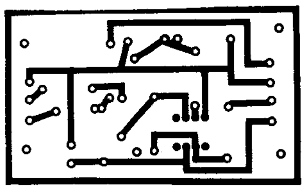


Fig.3. Layout and component placement for the cassette interface.

transferred are specified in an immediate byte. A code contained in the most significant four bits specifies the first register and the least significant four bits specify the second.

The register codes are

0000	= D
0001	= X
0010	= Y
0011	= U
0100	= S
0101	= PC
1000	= A
1001	= B
1010	= CC
1011	=DP

MUL. Multiplies the unsigned binary numbers in the A and B accumulators and places the unsigned result into the 16-bit D accumulator.

Although this short account of the 6809 is by no means complete, it should enable the constructor to start programming this very powerful processor.

Cassette tape interface

One facility which is more or less essential with any computer system is a means of storing programs. The cheapest convenient method of storage is a cassette tape and, as most users will have access to a cassette recorder, all that is required is the appropriate interface and software. This simple interface can be used with either version of the Nanocomp and will load the 1K memory in about 15s. An important part of the tape storage system is a set of routines, so readers using the original monitor will need to reprogram their e.p.r.o.m.

Data to be stored is transmitted to the recorder from a p.i.a. output line in the usual asynchronous serial format of a start bit, eight data bits and two stop bits for each data byte. Data bytes are transmitted in blocks of up to 16 bytes and each block starts with a 2-byte code, which identifies the start of a block on playback, followed by 2 bytes which give the start address of the block. The data bytes are then sent, followed by a checksum byte which is calculated by adding all the bytes in the block. The end of a recording is identified by an end-of-file code. Each bit is encoded onto the tape as one cycle of a square wave, and the period of 500μs or 2ms determines whether it is a 1 or 0 respectively. When loading a program, the period of each cycle is measured by checking whether it is greater or less than the average period, which makes the system reasonably tolerant of tape speed changes between different machines.

The interface plugs into the p.i.a. connector and is powered by the Nanocomp. Spare lines PB6 and PB7 are used for data transmission and reception so the interface can be permanently connected. The data to be recorded is transmitted from PB6 and reduced in amplitude by the potential divider R_1, R_2 in Fig. 2. On playback, the output from the recorder is limited and squared for driving the logic input of the p.i.a. A CA3140 is used for IC_1 because it operates satisfactorily from a single 5V supply.

The cassette interface can be assembled on a small p.c.b. as shown in Fig. 3. Only four connections are required to the Nanocomp and the connector numbers are

+5V	7a
0V	2a
PB6	12a
PB7	12b

If a ribbon cable is not available, ordinary stranded wire can be used and soldered onto the connector.

Operation

The L and P keys are used to load and dump data respectively. To save a program, key P and the display will request the start address of the memory block to be saved S, followed by the finish address F. Transmission will start immediately the last key is released, so the recorder should be started before this. When the recording is finished, F will appear in the left of the display which indicates that the recorder can be stopped. Abort or Reset will return the monitor prompt.

To load a program, key L and start the recorder just before the beginning of the program. To provide a form of feedback, the top and bottom segments of the lefthand display are turned on as data is received. When a 1 is received, the top segment is on and when a 0 is received, the bottom segment is on. If the program is loaded correctly, when the end-of-file code is received F is displayed. Abort or Reset returns the prompt. If a checksum error is encountered in one of the data blocks, a C is displayed and loading is stopped. If this occurs the tape must be rewound and restarted.

With some experimentation the record and playback levels can be optimised although, with a reasonable recorder, they are not critical. It should be noted that the requirements for recording data on a cassette tape are high so only high quality audio cassettes or, preferably, certified data quality should be used. Also, a worn recorder which does not give an acceptable performance with speech or music is unlikely to produce reliable data recordings. Auto record-level machines may also cause problems because their circuits are not designed to be used with a low mean-to-peak ratio square wave.

Although the Nanocomp was originally intended as a microprocessor trainer, many constructors may want to uprate the unit as shown, and interface the circuit to other systems. We intend to support this design with a further article describing extra peripheral devices such as a-to-d and d-to-a converters and a simple e.p.r.o.m. programmer.

The original monitor/utility program has been revised to remove a potential bug in the master mind program, and to improve the performance if poor quality keys are used. A hex list of the new monitor, which also contains the cassette interface software, can be obtained from the editorial office by sending a large s.a.e. clearly marked 6802 or 6809.

A set of p.c.b.s for the 6809 Nanocomp (power supply and logic board) will be available for £9.00 and a cassette interface board for £1.50, inclusive of v.a.t. and UK postage, from M. R. Sagin, 23 Keyes Road, London N.W.2.

Technomatic Ltd, 17 Burnley Road, London N.W. 10, 01-452 1500, and Magenta Electronics Ltd, 135 Hunter Street, Burton-on-Trent Staffs, 0283-65435, will be offering a kit of components. Both companies will also reprogram e.p.r.o.m.s for both versions of the Nanocomp.