

DREAMER

Nö 10

JUNE '81

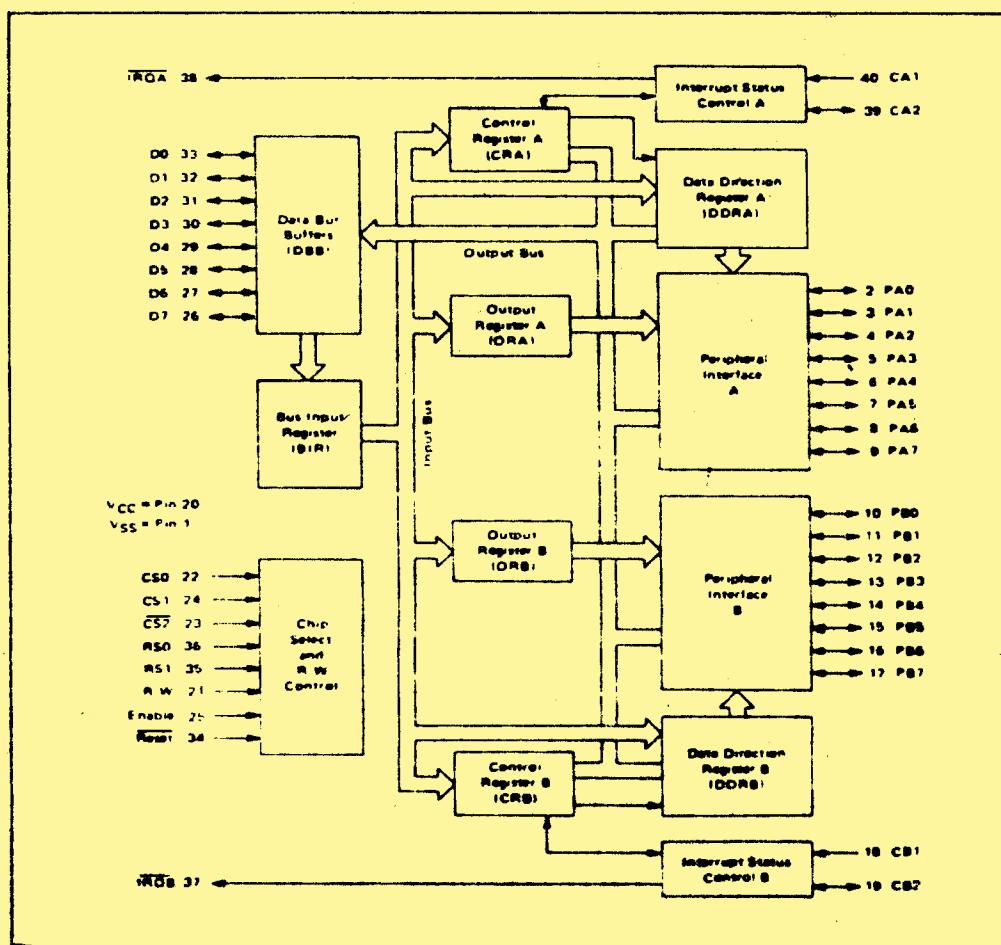
GRAEME SAMWAYS & GARRY NELSON,

N.S.W. 6800 USERS GROUP,

0200
 0202
 0204
 0206
 0208
 020A
 020C
 020E
 0210
 0212
 0214
 0200
 0202
 0204
 0206
 0208
 020A
 020C
 020E
 0210
 0212
 0214
 00E0
 SC00
 CA3F
 CB1F
 C30F
 F329
 DAB5
 7C01
 3C40
 1204
 1200
 ERASE
 $VC = 00$
 $VA = RND. 3F$
 $VB = RND. 1F$
 $V3 = RND. 0F$
 $I = DSP. V3$
 SHOW 5 AT VA
 $VC = VC + 01$
 SKF
 $VC = 40$
 GO TO 0204
 GO TO 0200

PERIPHERAL INTERFACE
ADAPTER

EXPANDED BLOCK DIAGRAM



PIN ASSIGNMENT	
1 O	C4 D 40
2 PA0	CA1 D 16
3 PA1	IRDA D 3F
4 PA2	IR0F D 37
5 PA3	AS0 D 36
6 PA4	AS1 D 35
7 PA5	REG01 D 34
8 PA6	DC D 33
9 PA7	D1 D 32
10 PB0	D2 D 31
11 PB1	D3 D 30
12 PB2	D4 D 29
13 PB3	D5 D 28
14 PB4	D6 D 27
15 PB5	D7 D 26
16 PB6	E D 25
17 PB7	CS1 D 24
18 CR1	CS2 D 23
19 CB2	CS0 D 22
20 VCC	R/W D 21

0200 CA3F
 0202 CB1F
 0204 A20A
 0206 DAB1
 0208 1200
 020A 80
 VA = RND. 3F
 VB = RND. 1F
 $I = 020A$
 SHOW 1 AT VA , VB
 GO TO 0200

* DREAMSOFT *

* 2K OF NON-VOLATILE DREAMSOFTWARE *

* SEE REVIEW IN DREAMER No. 6 *

DREAMTEXT - Creates alphanumeric displays with full 64 character ASCII subset.

BLOCK MOVE & BLOCK COMPARE - Copies any block of data to any other area of RAM.

TAPE LOAD & DUMP DISPLAY (AND TAPE VERIFY) - Simplifies tape handling.

BRANCH OFFSET CALCULATOR - Essential resident software for the Machine Code buff.

SUPERTYPEDREAM - Four formats including a disassembler provides hard copy via a disposals Baudot Teletypewriter.

SUBROUTINES - Many powerful User-callable subroutines.

All for \$30-00 which includes a programmed 2716 EPROM and a 37 page handbook containing installation and test instructions, interfacing information, data, list of subroutines, and a fully commented listing.

The DREAMSOFT EPROM is designed to fit on the J.R.Components Expansion board, but can, of course, be used without one. (See DREAMER No.7 for details.) For those using an E.A. 4K RAM board, we can supply full instructions for incorporating our EPROM. These include circuit diagram, board layout, and pin-by-pin connections. NO ADDITIONAL CHIPS ARE REQUIRED. Add \$5-00 if required.

Make Cheque or Money Order payable to :- DREAMSOFT,
P.O. Box 139,
MITCHAM, VIC. 3132

ANNOUNCING THE BIG ONE!

Wondering what to do with all that space in your expansion board memory? ----- Why not fill it with Dream Pontoon?

Dream Pontoon is that exciting card game Pontoon 21 translated into Chip 8. It has 4K of powerful logic that not only makes it a damned good player, but also results in a versatile game that can be played for hours without becoming boring.

IT FEATURES: * Memory mapped card deck for absolute realism
* Fully floating player options (anything you can do
your Dream can do better!)
* Probability based betting routines give high skill
* Automatic level of play settings and checksum

This is the biggest and most intelligent programme available for the Dream. To hell with Level II Basic, load this one up and see how smart a Dream can be.

Cassette and Instructions \$17.50

Fully Commented Listing \$7.50 Extra

Dream Rummy is an easy game to learn and great fun to play. High intelligence, memory mapped card deck, manual checksum and level of play settings give it reliability and realism. A bonus game of "Strip Jack Naked" is supplied free with this game - both require 2K, although "Strip Jack Naked" can be cut to 1K.

Cassette and Instructions \$10.00 Commented Listing (Rummy only) \$5.00 Extra

* DREAMCARDS

8 Highland Court, North Eltham 3095 Vic.
SOFTWARE THAT THINKS

Here it is, June already, and the end of our first subscription period. There were times when we thought we would never make it this far, but we did, and now, with ten issues behind us, we are saddling up for another six, but with a lot more confidence than we had last time, plenty of material, and lots of ideas. Our thanks to all those who contributed articles and programs. Without you, we can not exist, so keep them coming in.

SUBSCRIPTIONS are now due for the period JULY to DECEMBER, 1981. Enclosed with this issue are six blank labels. To renew your subscription, just print your name and address on all six of them, and send them back to us with your cheque or Money Order. The new subscription rates, as advised in the April DREAMER, are:-

SA18-00 within Australia
SA24-00 N.Z., Malasia and Singapore.
SA27-00 Other Overseas areas.

Of course, you may still purchase single copies if you prefer. Commencing with the July issue, the Single Issue Prices will be:-

SA3-50 within Australia
SA4-50 N.Z., Malasia and Singapore
SA5-00 Other Overseas areas.

All rates include postage by Airmail.

If you have some 'Free Issue' vouchers as payment for articles or programs we have published, you can use these as part or full payment of your subscription. To arrive at the correct amount to pay, deduct one sixth of the subscription price for each voucher you send back to us. If you have six, use them to pay the lot.

Don't delay, DO IT NOW, PLEASE! The sooner we get them, the more it helps us give you a fast and efficient service.

Also, don't forget, if you are paying by cheque from ANYWHERE OUTSIDE N.S.W., to add -10c STAMP DUTY, which goes to the N.S.W. State Government, (Bless their greedy little hearts), NOT US! This is not required on Money Orders.

Now, the results of our survey.

We received 202 cards back, which is just over 80% of those we sent out.

These 202 newsletters are shared by 296 readers, (excluding family), who have 288 DREAMs between them.

Of these, 114 are 6800's, 102 are 6802's, 2 Home Brew, and 70 were not specified.

How much RAM do they have? 79 have 1K, 15 have 2K, 49 have 3-4K, 63 have 5-8K, 3 have over 8K, and 79 did not specify.

Of the 139 DREAMs we know of with extra RAM, 57 have the J.R. Components Expansion Board, (and another 11 said they will have one shortly), 29 built the E.A. design, 11 built ours, (published in the first newsletter), and 31 have 'Home Brew' designs.

There are 45 'DREAMSOFT' Eproms in operation, plus 5 to come.

An amazing 144 out of 202 have built Joysticks. (Our most successful project.)

The item most requested was a 'High Resolution Graphics' project. Unfortunately, we do not know any way of doing this. We understand that Michael Bauer tried for some time to design such a modification, but was forced to give the idea away, as it would have meant completely redesigning the DREAM. If anyone has an answer to this one, we would be very pleased to hear from them.

The next most unpopular item was print legibility. We are aware that we have had some problems in this department, and are trying to improve it by changing the ribbon in the printer more often, and have also spoken to the man who prints them for us to ensure better reproduction. We do not want to stop using the Centronics Printer unless we just cannot cure the problem, as it saves us a lot of time, and ensures that the listings are correct, as we can run the program to check it, before printing the listing. If anyone has a copy of a program they cannot read, PLEASE, drop us a line

and let us know the Issue and Page numbers, and we will send you a replacement page with your next newsletter. We want you to USE the programs, not put them away because you can not decipher them.

Some of the other requests were for;

- More 'serious' programs. Sure, as soon as someone sends them to us - Over to you, programmers.

- Holes for binders. A good idea, but which type? 3 ring, 4 ring, 2 ring, spiral? There are so many different types in use, we decided to let you put your own in, otherwise the edge of the page would look like a swiss cheese if we try to please everyone.

- A Tape Motor Controller. See Lindsay Ford's article in E.A., May 1981.

- How to Interface and Control Devices, e.g., LED's, Servo's etc., from the PIA. We would welcome some articles on this subject, so if your DREAM controls your electric train set, or monitors your amateur radio transceiver, tell us all how you did it.

- A 'Question & Answer' page. We do not have the time to answer the questions, but we are prepared to print them, with your name and address, so that someone who can answer it can write to you direct, or send the answer in to us to be printed, if it is of general interest.

- 'Letters to the Editor' column. If you want this, send them to us, suitably marked as such, and we will print them.

- Tapes of Programs. Sorry, No. We do not have the equipment or the time to produce 300 high quality cassettes each month that will load reliably on a variety of machines, all operating at different frequencies. Micro-80, who have a much higher circulation than us, produce tapes for their programs, and they have to charge an extra \$3-00 a month for them, to cover costs.

- Flow Charts of Programs. Space usually precludes these, but we hope to have an article on 'How to Flow Chart' shortly. In the meantime, we will include fully commented disassembled listings as often as we can.

- How to Interface a Printer. Coming up soon.

All the other requests for things that we do not have an article in the pipeline for, we will add to our 'Wanted' column, so go to it.

That is it for now, because it is 1.30a.m., and I have to finish typeing this and take it to Graeme to check, before it goes to the printer. (I forgot to check read it last month, in the rush to get it to the printer.) Did you notice the spelling mistakes? Sorry about that.

NEXT MONTH, we will have;

- ZOONAYMAN. An observation game. Two versions, one for two players, and one to play against the computer.

- KNOT-SO-EASY. This will 'tie you up' for hours.

- A MORSE CODE SENDER in Machine Language.

- A MENU SYSTEM, by Bruce Mitchell.

- DRAWING, including a Flow Chart.

- VIDEOTELTYPEWRITER. An eye-catching advertising display.

- TIC-TAC-TOE, against the computer.

- MULTIPLICATION TABLES. Help your children learn their tables.

- CASTLE INVADERS. A 1K Space Invaders game.

- COMPUCROSSES. Can you beat the computer?

- LOTTO. We haven't heard of any big winners from the last Lotto number selector we published, so here is another one to try.

- DUCK SHOOT. It is a riot.

- SLIDE. For two players, test your skill.

PLUS, an article on a Video Tape Recorder Controller, and an ASCII keyboard for your DREAM, so, DON'T FORGET TO RENEW YOUR SUBSCRIPTION.

Until next time,

Happy DREAMing,

Garry and Graeme.

SELECT A PROGRAM ON TAPE.

K. SEMRAD,

By encoding a program before dumping it onto tape, you can then have the computer search for it and load only the program you asked for. This also eliminates having the program shifted one or more bytes, caused by noise in your leader tone. (FF in 0200.)

NOTE: Can not be used where 0080 - 0100 is used by program on 1K machines.

DUMPING.

Instead of dumping a program by; FN, 2, key in the following first:-
0080 8D0B BDC3 4186 0KBD C32B 8D03 7EC3 484F
0090 BDC2 F34A 26FA 39**

(K stands for your code, 0 - F.)

Store start and end locations of program as usual at location 0002-0006, then go from 0080 (0080, FN, 3). The tape will then have at least 256 Bit-time of 2400 Hz tone, then your code, (0 - F), another 256 Bit-time of 2400 Hz tone, and then your program.

LOADING.

This program looks for a short leader tone, followed by a single byte, followed by another short leader tone. It then compares the loaded Byte with whatever was keyed in, and if it is the same, it will commence loading.

0080	C0MM	MMDF	02CE	NNNN	DF04	BDC0	79CE	419E
0090	DF1E	BDC1	A186	1C97	2E8D	41BD	C2C4	810X
00A0	22F9	9730	8D3F	BDC3	1097	3191	3027	168D
00B0	34BD	C079	7F00	2E96	308D	1E86	3C97	2E96
00C0	318D	1620	2F8D	1EBD	C310	A700	089C	0426
00D0	F686	3FB0	C2FE	7EC0	00BD	C193	C610	D72F
00E0	C605	7EC2	247F	0032	BDC3	41BD	C2F5	B680
00F0	122A	F27C	0032	26F0	39**			

(MMMM stands for starting location, e.g. 0200)

(NNNN stands for end of program, e.g. 0300 or 0400)

(X stands for number of programs on tape, maximum F)

Now go from 0080 (0080, FN, 3.) A question mark will appear on the screen. Start the tape and key in program code, (0 - F). You will see on the left side of the screen the code you asked for and on the right side the code of the program the machine is receiving but not loading. When the selected program comes up, loading will start. For this, DMA has to be off, so the screen will 'blank out'.

Location 00D7 can be changed to C360, if you do not want the program to start straight away.

Both programs are fully locatable, so, if you have more than 1K, put the load program somewhere in the second K, and that lets you use the programs that use the 0080 - 0100 area.

Finally, a short program I use for sending in programs to our newsletter.

0080	8D0D	C640	8D09	C641	8604	8D05	7EC3	4886
0090	0F97	304F	F780	12BD	C2F3	4C26	F77A	0030
00A0	26F1	39**						

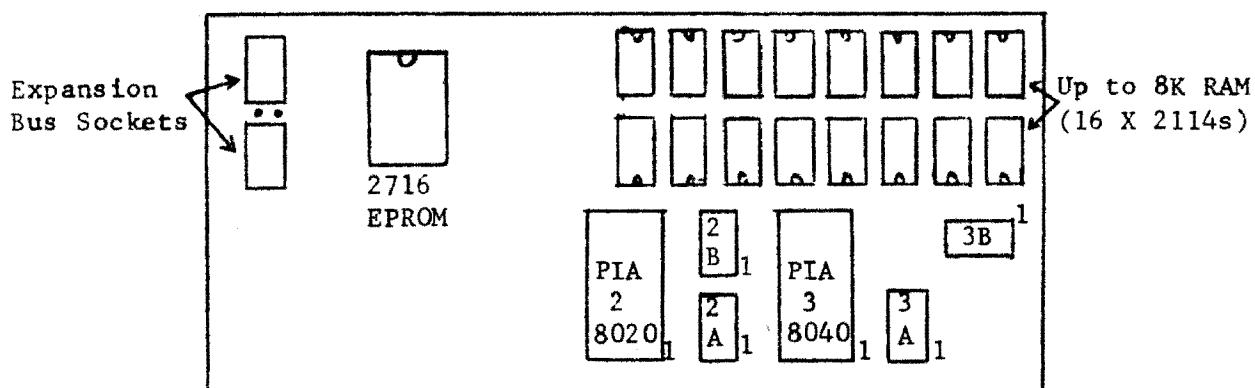
This puts 22 seconds of 2400 Hz tone, then 22 seconds of 1200 Hz tone, then 5 seconds of 2400 Hz tone, and then the program on tape. (Don't forget to put the start and end locations in 0002 - 0006 first.)

One of the suggestions that we received recently was that we publish the output port connections on the J.R.Components Expansion board, so those who did not have one could see how they are laid out.

You can put two PIA's (Peripheral Interface Adaptors) on the board. (These are not included in the kit.) They reside at addresses 8020, 1, 2, 3, and 8040, 1, 2, 3. There is also provision for up to 8K RAM memory expansion, and one 2K EPROM.

I call these PIA2 and PIA3. PIA1 is on the main board at address 8010-3. Each PIA has two nearly identical I/O (Input/Output) Ports. (A and B) Each port is terminated on a 16 pin DIL socket. PIA2A and PIA3A sockets are identical, and they are the ones closest to pin 1 of each PIA.

This is the layout of the expansion board.



The pin connections for the I/O Port sockets are:-

PIA2A & PIA3A

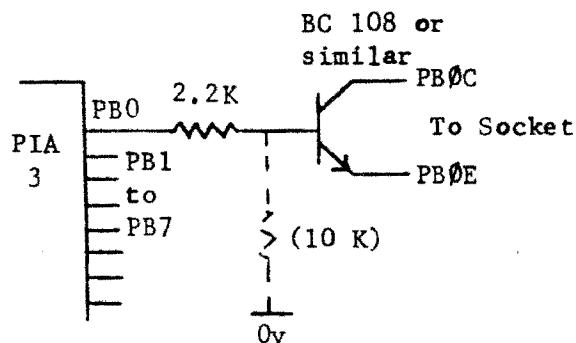
9	PA7	CA1	8	9	PB7	CB2	8	9	PB3E	PB4C	8
10	PA6	N/c	7(+5v)	10	PB6	N/c	7(+5v)	10	PB3C	PB4E	7
11	PA5	N/c	6(+5v)	11	PB5	N/c	6(+5v)	11	PB2E	PB5C	6
12	PA4	N/c	5(+5v)	12	PB4	N/c	5(+5v)	12	PB2C	PB5E	5
13	PA3	N/c	4(0v)	13	PB3	N/c	4(0v)	13	PB1E	PB6C	4
14	PA2	N/c	3(0v)	14	PB2	N/c	3(0v)	14	PB1C	PB6E	3
15	PA1	N/c	2(0v)	15	PB1	N/c	2(0v)	15	PBOE	PB7C	2
16	PA0	CA2	1	16	PB0	CB1	1	16	PBOC	PB7E	1

PIA2B

9	PB7	CB2	8	9	PB3E	PB4C	8
10	PB6	N/c	7(+5v)	10	PB3C	PB4E	7
11	PB5	N/c	6(+5v)	11	PB2E	PB5C	6
12	PB4	N/c	5(+5v)	12	PB2C	PB5E	5
13	PB3	N/c	4(0v)	13	PB1E	PB6C	4
14	PB2	N/c	3(0v)	14	PB1C	PB6E	3
15	PB1	N/c	2(0v)	15	PBOE	PB7C	2
16	PB0	CB1	1	16	PBOC	PB7E	1

CB2 CB1 (Not on skt.)

Pins 2 to 7 were not connected, so I put jumpers on the P.C.B. (be careful of the tracks between the P.C.B. pads), so that pins 2, 3 and 4 are connected to 0v, and pins 5, 6 and 7 are connected to +5v. (To run Sound Effects Generators, etc.) PIA2B is similar to PIA2A, except CB1 and CB2 are opposite to CA1 and CA2. (You can change P.C.B. jumpers to make them the same.)



The PIA3B port is designed to be buffered. This is done as shown at left. Both the Emitter and Collector are connected to the output socket so that it can be used in 'open collector' or 'emitter follower' configuration. The 10K resistor to 0v is optional, and I have omitted them on my board.

On the PIA3B Port, PB0 C is output PB0 Collector, PB0 E is output PB0 Emitter. CB1 and CB2 are separate P.C.B. pins, as they do not fit on a 16 pin socket.

For those of you who do not like abbreviations, P.I.A. stands for Peripheral Interface Adaptor, and you all have one. (In your DREAM, of course.)

It is located at \$8010 through 8013. If you have a J.R.Components Expansion board, you can have two more PIA's. (See the article on the layout of the J.R. board, 'Dream Connections', in this issue.)

WARNING: Do not try re-assigning the PIA in your DREAM, especially the B port, without careful research, because assigning PB7 as an output will in 90% of cases destroy the PIA. This is because PB7 is permanently connected to the Tape Input circuitry, which will place two outputs on the one line. This is a NO NO! The rule is, one line, one output, unless all but one are in tristate mode.

Playing with the A and B port control lines can also have a similar effect, or turn off the display. Otherwise, the worst that can happen is that you turn the speaker on and off unexpectedly. You may also lose control of the keyboard if you play with the A port, as PIA 1 on the DREAM board uses the A port for the keyboard, and the B port for Tape In/Out and Video control. The B port is as follows; Bit 7, Tape In, (Must always be an input.)

Bit 6, Speaker On, (1), Speaker Off, (0)

Bit 0, Speaker and Tape Out, (0), 1200Hz, (1), 2400Hz.

The other lines are all assigned as outputs, so you can control things from them without re-assigning them.

The two extra PIA's on the J.R. Components board are not restricted, and this article will mainly apply to them, as PIA 1 is already 'dedicated'.

Each PIA consists of two nearly identical Input/Output (I/O) Ports. Each port consists of 10 lines. Eight bi-directional Data lines, and two Control lines. The only difference being that the first port, (called the 'A' port), has 2K pull up resistors on the Data lines and its second Control line. (More on this later.) The second port, (called the 'B' port), is Tristate. (High impedance Inputs.)

The Data lines are numbered PA \emptyset -7, or PB \emptyset -7. That is, P for Peripheral, then whether they are A or B port, then their number. For convenience, I am going to call them P \emptyset -7, but normally, whether they are A or B is specified. Each line corresponds to the 6800 Data lines. (I.E., P \emptyset corresponds to D \emptyset .) Each of these can be used individually (or as a group) as an Input or an Output.

Each PIA has six registers, three for each port. These are the

Output Register, (OR)

Data Direction Register (DDR)

Control Register (CR)

These reside in memory in four locations. (What, six Registers in four locations?) Taking the DREAM PIA, 8010 - 8013, We have the A port Output Register, and the A port Data Direction Register at 8010. The A port Control Register is at 8011. Then, it repeats itself for the B port, I.E., B port Output Register and B port Data Direction Register at 8012, B port Control Register at 8013.

The Control Registers are ALWAYS present at 8011 and 8013, but the Output Register and Data Direction Register have to be selected.

To select the Data Direction Register, (DDR), you must store a binary 0 in bit 2 of the Control Register for that port. Once this is done, if you 'look' at the location before the Control Register, you will see the DDR. At this stage, you can assign each bit, (or line), in the port to be an Input, by writing a 0 into that bit in the DDR, or an Output, by writing a 1 into that bit in the DDR. So, if you want all eight lines outputs, write FF into the DDR, or if you want all inputs, write 00 into the DDR. If you want say Line 4 as an Output, and the rest as Inputs, you write 10 into the DDR. Get it? If you don't, read it again, until you do!

NOTE: As soon as you assign a line as an Output, a logical 0 state appears on that bit in the port.

To gain access to the Output Register, you write a 1 into the second bit in the Control Register. Now the Output Register will be present at the

THE 6821 PIA (Cont)

location before the Control Register. Here you can read what Inputs are present on each line, or what output is on each line. To change the state of Outputs, (if any), you simply write the required information into the port. As you do, that state appears in the Output port. So, to make line P4 a 1, write 10 into the Output port, and a 1 will appear in line P4. If a line is an Input, this will have no effect, and the next time you read the Output port, the Input state will still be seen.

If you wish to see what configuration the port is in, just select the DDR, then read it.

Now to the Control Register. Firstly, you will find a copy of the Control Register Organisation on the next page. (It is the one out of the 'Chips' manual. Ssssh, don't tell Michael.) Refer to it as you read this next section, as the table tells you how to do things.

The Control Register controls which register you see. (DDR or Output.) It sets the mode of both Control Lines, and indicates their state. It also controls the System Interrupt Request (IRQ).

NOTE: On the J.R. Board, you have the option to connect the PIA's to the system Reset and IRQ. I do not advise this unless you have a specific purpose, because: 1) On System Reset, the PIA registers are all set to inputs. If Reset were connected, you could not set them up using Memod.

2) If you got an IRQ request from the second or third PIA the DREAM would go 'haywire' unless you had special routines.

Leave both disconnected unless you really need them, if you don't, well, I did warn you.

Meanwhile, back to the Control Register bits.

BITS 0 and 1 : CA1 (CB1) Control. (Always Input.)

Bit 0: This bit has no effect, unless the PIA is connected to the System IRQ, which is NOT recommended. (See above.)

Bit 1: This bit tells the PIA how to turn on the IRQ Flag. (Bit 7 in the Control Register.) As you can not see the C1 or C2 inputs directly, a 0 in bit 1 sets bit 7 High when C1 goes from High to Low, and a 1 in bit 1 sets bit 7 High when C1 goes from Low to High.

Bit 2 : DDR Access. 0 = DDR Access, 1 = Output Register Access.

Bits 5, 4, 3 : Control CA2 (CB2)

Bit 5 : Tells the PIA whether C2 is an Input (0), or an Output (1). If Input, (bit 5 = 0,) Bit 4 is the same as bit 1, but sets bit 6, not bit 7. Bit 3 is the same as bit 0.

If Output, (bit 5 = 1,) Bit 4, if a 1 gives direct access to C2. I.E. Put a 1 in bit 3 and C2 will go High, or put a 0 in bit 3 and C2 will go Low.

If Bit 4 is 0, CA2 is set up in handshake or strobe mode. Here CB2 is different from CA2. See the Register breakdown sheet. This mode is the most complicated and is used with Serial Interfaces, etc. If you can design one of these, you will understand this control format. If not, don't worry, it is beyond the scope of this article, but I will go into it in the future.

Bits 6 and 7 : Show the states of C2, (bit 6), and C1 (bit 7). They are called IRQ Flags because the 6800 uses these to find out which device asked for an interrupt. These two bits are READ ONLY. (You can not change them by writing 00 into the Control Register.)

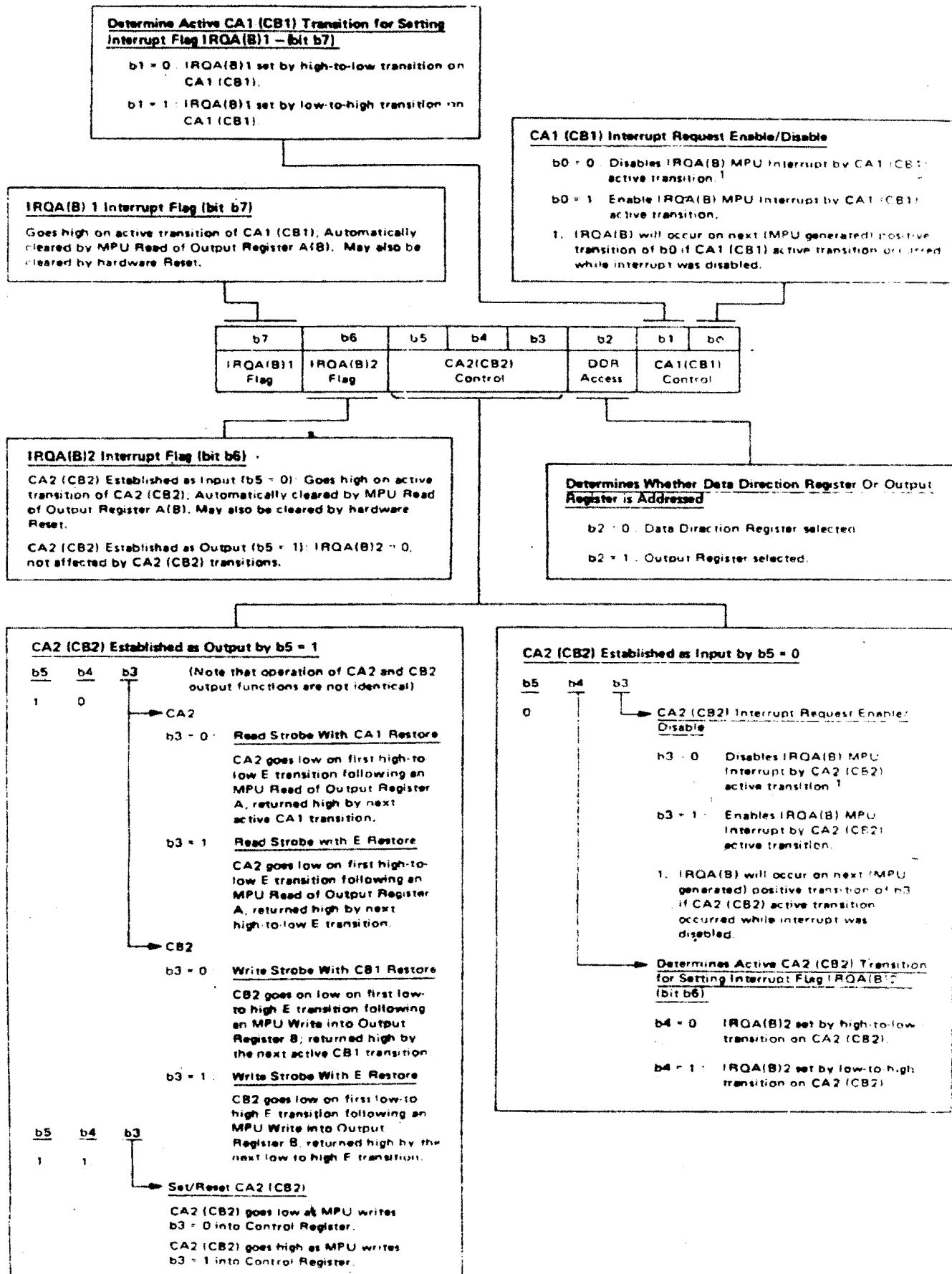
Each bit is set as defined in the respective Control Line control bits. To reset the bits, you either; A) Reset the whole chip, (a bit drastic), or B) Read or Test the associated Data Register, in some form or another. These are all Machine Code instructions, which we will not go into here. If you want to know more, re-read the article on 6800 Assembly Language programming in last months issue, and buy yourself some text books.

Well, what a lot of BITS and pieces. That is the 6821. It is a BIT B*!% complicated, isn't it? Use the Control Register chart as a guide to its use, and go to it.

Next month, I will include an article on an Output device controlled by the DREAM, which I use to control my Father's Video Tape Recorder. (It changes channels, turns it on and off, etc.)

THE 6821 PIA (Cont)

CONTROL REGISTER ORGANISATION



THE DREAMPLIFIER

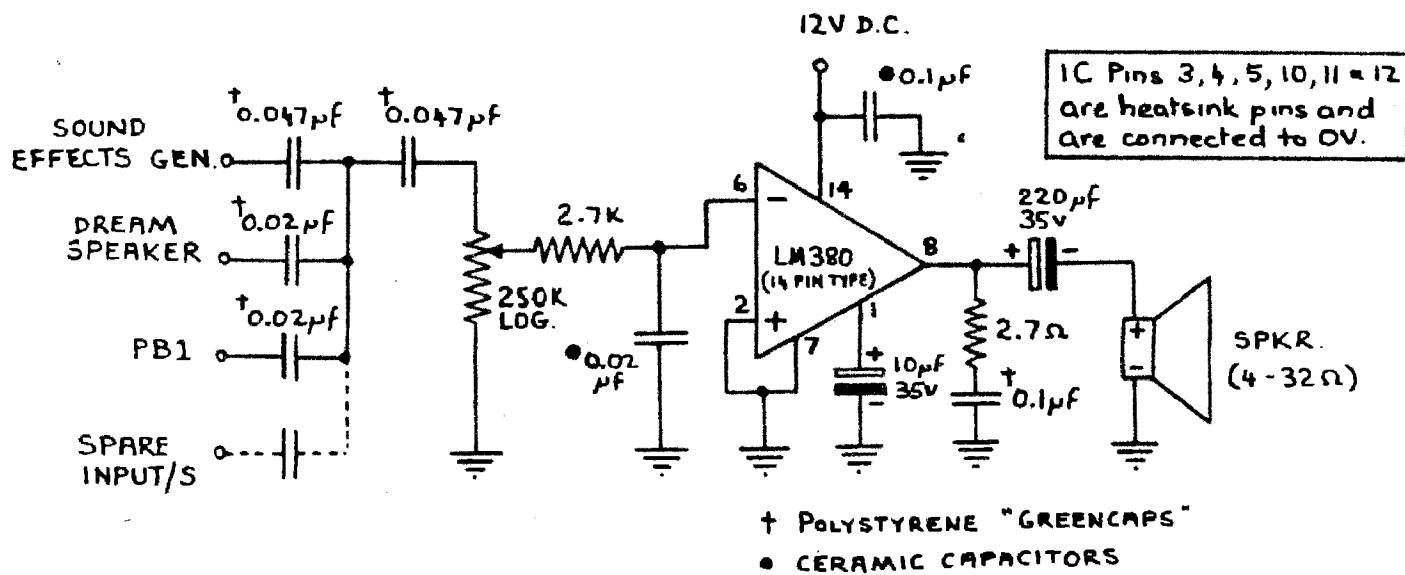
by Lindsay R. Ford,

"Dreamcards",
[REDACTED]

Is your Sound Effects Generator lying mute on a shelf, gathering dust? Does 'Opus 1' struggle forth from your "old pocket radio" like a digital death-rattle? Well, break through the sound barrier with the amazing Dreamplifier! Now you can hear your Dream's crashing crescendos, dazzling discords, beeps, squawks and farts in glorious living technicolour (almost).

In fact the Dreamplifier is nothing more than a mild-mannered 1 watt op-amp, cleverly disguised as a computer amplifier. The circuit shown here links into your Dream (so you can hear the reset beep, keydown etc.), into PB1 (allowing you to play 'Opus 1') and into the Sound Effects Generator. Extra inputs can be fed via Greencap capacitors to the track marked with an asterisk on the veroboard diagram. For instance, you could couple PB7 to this track via a 0.02 μ F cap. so you could hear the tape load. Choose capacitor values between 0.01 μ F and 0.1 μ F for these additional inputs, selecting the value to balance the volume of the added sound to the reset beep.

The circuit of the Dreamplifier is so simple it's not worth describing, although its balance is critical and so a list of possible fault conditions is given below. Just remember when assembling it not to bridge the copper tracks with great gobbs of solder (or little ones) and use a socket for the LM380. The LM380 must also be fitted with a heatsink. Cover the underside of the heatsink with heatsink compound except for a few mm. at each end, then put epoxy resin on the end bits. Now stick it onto the LM380 and you're in business (unless you forgot to mark the Pin 1 end, in which case you're the one that's sunk!)

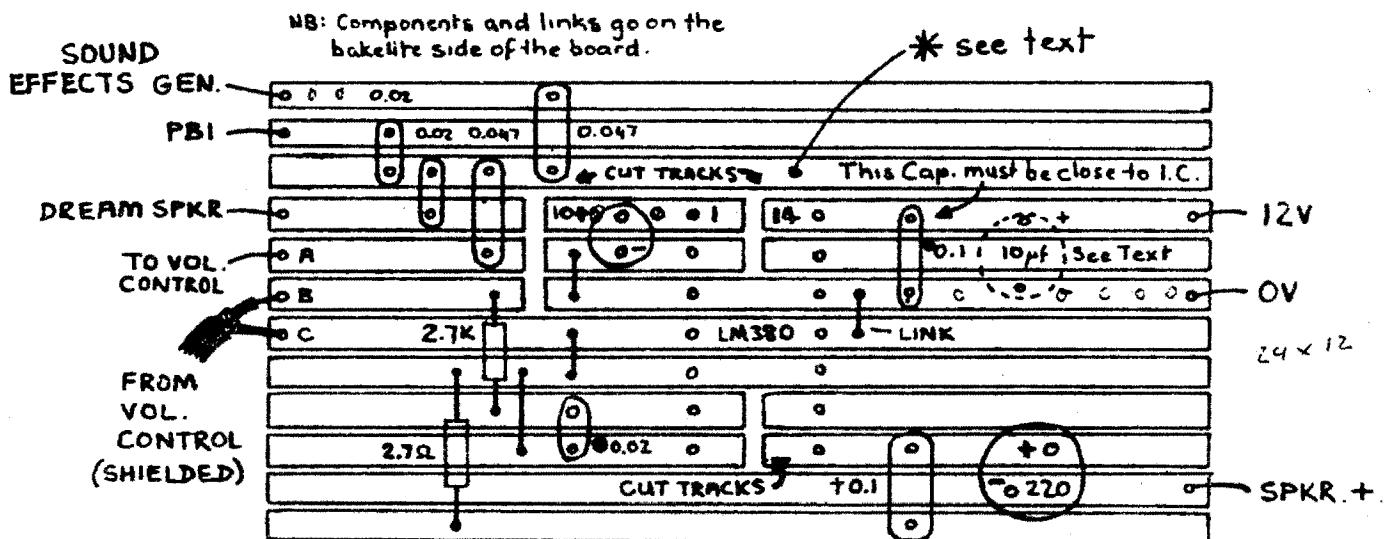
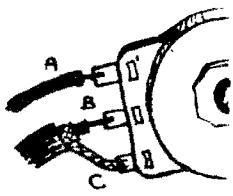


One hassle you may discover with the Dreamplifier is that amplifiers don't get on terribly well with computers. You see, there's all these stray square waves and harmonics drifting around inside your Dream, just waiting to violate some hapless op-amp. To ensure that your Dreamplifier doesn't disintegrate with a banshee wail (or suffer some less fatal malady), check the following;

- 1) Is the 12 volt regulator in your power supply a 1 amp type?
If not, fit one (the Dreamplifier won't overload your supply).

- 2) Is the LM380 getting hot even when there's no sound being put through? Is it sending your Dream's video display haywire? If so, it's probably amplifying stray square waves radiated from the clock circuit and feeding them back to the Dream. Increasing the value of the ceramic capacitor on Pin 6 of the LM380 should cure it, but don't go too high or you'll lose treble response.
- 3) Are you getting a sound like a motor boat from the speaker? This indicates lousy power supply regulation. Fitting a 10 μ F 35 VDC electrolytic cap. in parallel with the ceramic cap. on Pin 14 of the IC should help a lot.
- 4) Does the Dreamamplifier sound terrible at high volume (conversely is the sound output at maximum volume too low)? Change the value of the electrolytic capacitor on Pin 8 of the IC to suit your particular speaker (increasing its value gives a higher output and vice versa), but don't go over 560 μ F.
- 5) Make sure the OV lead is taken by the shortest possible route to the power supply and use good quality multi-strand copper wire. Don't cut the veroboard tracks between pins 3&12, 4&11 or 5&10 of the IC. Otherwise the Dreamamplifier will howl in protest.

VOL. & M.F. CONTROL



P.S: Did you see Lindsay Ford's article on a TAPE MOTOR CONTROLLER for the DREAM in the May issue of Electronics Australia? If you missed it, you should get hold of a copy, as a lot of people asked us on the survey cards whether this was possible.

NOTE. This is a 'free plug' for LINDSAY, NOT E.A., as he is also the man behind the DREAMCARDS organisation. (See his advertisement in this issue, DREAM RUMMY is fantastic!)

For those of you astute enough to notice, yes, the address shown on the 'DREAMCARDS' ad is different to the one at the start of this article. The reason is simple, Lindsay is moving, and the address for DREAMCARDS will change to 8 Highland Crt, Nth. Eltham 3095, from the start of JULY.

G & G.

DREAMSOFT,
P.O.BOX 139,
MITCHAM. 3132

This program will disassemble your Chip-8 programs into the format ADDRESS, INSTRUCTION, MNEMONICS., on your teleprinter. You need a DREAMSOFT EPROM and a teleprinter to run it.

All you do is load the program you wish to disassemble, load the Disassembler program from 0400 - 070F, and run, 0400, FN, 3. It will then prompt you for the start and finish addresses of the block to be disassembled. After you tell it, the DREAM will print out the disassembled block.

BEWARE OF MACHINE CODE OR DATA IN THE PROGRAM, as this can produce some weird Mnemonics, as the DREAM attempts to decipher it. (For these sections of the program, use the normal PRINT FUNCTION (7) of your DREAMSOFT Eeprom. The first time you run the program, it sets up 00F0 so Key 8 to run it again. NOTE: If you wish to disassemble 0080-00FF, do not run 0400, FN, 3, or use Key 8. Run 040B, FN, 3, as the first few bytes of the program changes 00F0 to 040B, which will corrupt the program being disassembled.

0400	86	20	36	86	1C	36	CE	<u>04</u>	0B	DF	F0	CE	<u>0E</u>	74	BD	19
0410	04	BD	18	00	96	03	84	FE	97	03	96	05	<u>84</u>	FE	97	05
0420	DE	<u>04</u>	08	08	DF	<u>04</u>	BD	19	A4	8D	25	8D	23	DE	02	BD
0430	19	37	BD	1A	EE	DE	02	EE	06	BD	19	37	BD	1A	EE	8D
0440	14	DE	02	08	08	DF	02	9C	04	26	E0	8D	03	39	00	00
0450	86	02	7E	1A	F5	DE	02	EE	06	DF	28	DF	14	96	14	84
0460	0F	97	14	97	2E	96	29	44	44	44	44	97	2F	CE	<u>06</u>	36
0470	96	28	84	F0	08	08	80	10	24	FA	EE	00	6E	00	00	00
0480	D6	28	26	16	96	29	81	E0	27	0B	81	EE	27	02	26	11
0490	CE	<u>06</u>	99	20	14	CE	<u>06</u>	E5	20	0F	CE	<u>06</u>	EB	8D	0A	20
04A0	1E	CE	<u>07</u>	02	29	03	CE	<u>07</u>	06	7E	<u>05</u>	41	SD	0C	CE	<u>06</u>
04B0	84	20	F6	CE	<u>06</u>	91	8D	F1	20	05	CE	<u>06</u>	8A	8D	EA	DE
04C0	14	7E	19	37	8D	21	BD	<u>05</u>	57	CE	<u>06</u>	D9	8D	73	20	50
04D0	8D	15	BD	<u>05</u>	57	CE	<u>06</u>	DE	20	F2	8D	F9	8D	73	20	07
04E0	00	00	CE	<u>06</u>	C4	20	5A	CE	<u>06</u>	C8	20	55	20	74	8D	30
04F0	8D	65	CE	<u>06</u>	CF	20	4A	8D	<u>33</u>	8D	5C	20	E5	8D	F3	8D
0500	56	20	29	8D	04	8D	50	20	B6	CE	<u>06</u>	C9	20	33	8D	F9
0510	8D	45	8D	F5	CE	<u>06</u>	AA	8D	28	20	11	8D	EC	8D	38	CE
0520	<u>06</u>	D4	8D	1D	20	<u>06</u>	00	CE	<u>06</u>	A0	20	15	CE	<u>06</u>	BD	8D
0530	10	96	2E	7E	19	47	CE	<u>07</u>	0B	8D	06	20	C2	<u>08</u>	BD	19
0540	56	A6	00	81	04	26	F6	<u>39</u>	C6	R1	D1	29	27	3B	C6	9E
0550	D1	29	27	2C	7E	<u>04</u>	A1	CE	<u>06</u>	B9	20	E5	8D	C9	8D	00
0560	8D	F5	20	2D	8D	C1	8D	C4	01	01	8D	30	20	23	00	00
0570	8D	B5	8D	B8	8D	E1	20	29	8D	AD	8D	B0	8D	1E	20	21
0580	8D	A5	8D	AB	8D	D1	7E	<u>04</u>	E2	8D	90	8D	9F	8D	00	20
0590	F5	96	29	7E	19	3F	8D	94	8D	BD	20	90	CE	<u>06</u>	BF	20
05A0	A0	CE	<u>06</u>	BD	8D	98	96	2F	7E	19	47	8D	ED	8D	A8	20
05B0	E0	8D	<u>E7</u>	8D	R2	CE	<u>06</u>	A5	8D	87	20	D5	8D	D8	CE	<u>06</u>
05C0	RR	20	F5	8D	D5	8D	98	20	D8	SD	CB	CE	<u>06</u>	AE	8D	CF
05D0	20	CF	8D	C2	CE	<u>06</u>	B3	20	F5	8D	BB	CE	<u>06</u>	AA	20	EE
05E0	00	00	8D	B2	CE	<u>06</u>	CB	20	E5	CE	<u>06</u>	F8	8D	B1	96	29
05F0	84	0F	26	07	86	10	BD	19	3F	20	03	BD	19	47	CE	<u>06</u>
0600	F3	8D	9C	8D	95	CE	<u>06</u>	FE	20	C4	D6	29	C4	0F	27	B3
0610	5A	27	B6	5A	27	BC	<u>5A</u>	5A	27	BF	5A	27	C5	7E	<u>04</u>	A1
0620	CE	<u>06</u>	59	C6	09	A6	00	91	29	27	09	08	08	08	5A	26
0630	F4	7E	<u>04</u>	A6	EE	01	6E	00	<u>04</u>	80	<u>04</u>	BA	<u>04</u>	B3	<u>05</u>	50
0640	05	64	<u>05</u>	70	05	AB	<u>05</u>	BC	<u>06</u>	0A	<u>05</u>	78	<u>05</u>	03	<u>04</u>	AC
0650	05	B1	<u>05</u>	E9	<u>05</u>	48	<u>06</u>	20	08	07	<u>04</u>	EE	<u>04</u>	<u>04</u>	F7	15
0660	<u>04</u>	FD	18	<u>05</u>	36	1E	<u>05</u>	0E	29	<u>05</u>	18	32	<u>04</u>	C4	55	<u>04</u>
0670	D0	65	<u>04</u>	DA	00	43	48	49	50	20	38	20	44	49	53	4D

CHIP - 8 DISASSEMBLER (Cont)

0680	42	40	52	04	20	28	28	56	4F	04	47	4F	20	54	4F	20
0690	04	44	4F	20	53	55	42	20	04	52	45	54	55	52	4E	04
06A0	53	4B	46	20	04	52	4E	44	2E	04	20	28	20	04	20	4F
06B0	52	20	04	20	41	4E	44	20	04	20	3D	20	04	56	04	20
06C0	4E	45	20	04	48	45	59	04	4D	49	04	20	2D	20	04	54
06D0	49	4D	45	04	44	53	50	20	04	44	45	51	2E	04	56	4F
06E0	20	54	4F	20	04	45	52	41	53	45	04	43	41	40	40	20
06F0	4D	2F	43	20	41	54	20	04	53	48	4F	57	20	04	20	20
0700	20	04	4E	4F	50	04	53	54	4F	50	04	54	4F	4E	45	04

Shown below is a sample printout of all the Mnemonics generated by the program. (Do not try to run it as a program, as it will not work. It is only a demonstration of the Disassembler program's capabilities.)

The program can be relocated to X400 by changing all the underlined bytes.

0200	1234	GO TO 0234
0202	B234	GO TO 0234 + VO
0204	2345	DO SUB 0345
0206	00EE	RETURN
0208	3266	SKF V2 = 66
020A	4266	SKF V2 NE 66
020C	5AB0	SKF VA = VB
020E	9AB0	SKF VA NE VB
0210	E59E	SKF VS = KEY
0212	E5A1	SKF VS NE KEY
0214	6ABB	VA = BB
0216	C9FF	V9 = RND. FF
0218	74EE	V4 = V4 + EE
021A	8AB0	VA = VB
021C	8AB1	VA = VA OR VB
021E	8AB2	VA = VA AND VB
0220	8AB4	VA = VA + VB
0222	8AB5	VA = VA - VB
0224	F407	V4 = TIME
0226	F40A	V4 = KEY
0228	F415	TIME = V4
022A	F418	TONE = V4
022C	AFED	I = 0FED
022E	F41E	I = I + V4
0230	F129	I = DSP. V1
0232	F133	MI = DEQ. V1
0234	F155	MI = VO TO V1
0236	F165	VO TO V1 = MI
0238	00E0	ERASE
023A	0FED	CALL M/C AT 0FED
023C	CAB5	VA = RND. B5
023E	0000	NOP
0240	F000	STOP
0242	12FF	GO TO 02FF
0244	FF88	STOP
0246	88EE	NOP
0248	D895	SHOW 5 AT V8 , V9
024A	D980	SHOW 10 AT V9 , V8
024C	477A	SKF V7 NE 7A
024E	2020	DO SUB 0020

Many users have requested information on how to extend the buffer size of the ALPHA DISPLAY to utilize their expanded RAM. Others have criticised the lack of editing functions and the restricted character set. The original program was designed for easy enhancement and the commented listing gave clues as to how to go about it. Because of space restrictions, however, this listing wasn't published. So here are some solutions to these restrictions:-

1. Whether or not you have expanded RAM, the character set can be expanded. There are 32 bytes of free memory from 00C0 to 00DF which were reserved for 16 additional character patterns, each requiring 2 bytes and corresponding to the codes 30 thru 3F. These character patterns are formulated according to the CHIPOS Manual, under the heading "Displaying a 3 x 5 dot symbol".

2. Users with 2K or more of RAM may utilise the program patches given here to extend the buffer size. Values given here assume a 2K RAM, with the buffer starting at \$0380 giving a message in excess of 1100 (dec.) char-codes. Users with more than 2K should change the 'end-of-buffer-warning-beep-address' accordingly.

3. A user-written function may be implemented by placing a subroutine call (2mmm or 0mmm) at location 027A (replacing the 1208 instr. there). This subroutine will be invoked by the 'An' code. Codes '4n' and '5n' are also available to add extra functions, (e.g. control of peripheral devices, etc). See patches.

Increasing the size of the buffer for the Alpha Display makes it more likely to make an error when keying in a long message. Therefore, an "ALTER" function would be very handy to edit out errors during the recording process. This would be an ideal application for the unused control code, 'An', where 'n' is the number of codes (bytes) to be deleted from the buffer (n=0 would be taken to mean n=16). During recording, the An control code will cause the following events:-

1. The buffer pointer is backspaced by n bytes;
2. The program goes into playback mode temporarily to replay the last frame up to the point of deletion; (frames are separated by ERASE codes.);
3. The program returns to recording mode.

The Alter function is best implemented as a machine-code subroutine, by placing an appropriate CALL (0mmm) instruction at location 027A.

As a real bonus, the Alter function may be used to edit a previously saved message. This procedure is especially useful for appending new data onto an old message. Just proceed thusly:-

1. Load message from tape (if applicable);
2. Run Alpha Display in Playback; Wait for the point where alteration/appending is desired; Hit RESET;
3. Now use MEMOD to insert an 'AO' byte in the buffer at this point. (N.B. Buffer pointer is 001A,18 which contains the 2-byte address you're after.)
4. Run Alpha Display in playback again; when AO is reached, it will automatically switch to Record mode and let you insert new data (on top of any existing data, of course, if any.).

The computer memory dump for the program contains the above enhancements (except for extended character set). The following memory locations will be of benefit to those wishing to further modify the Alpha Display program.

(Don't forget to load data from 0080!)

RAM Locations of interest:-

001A,8 (2 bytes)	Buffer pointer (adrs of next char-code)
0380.....	Message buffer
02E7,8 (2 bytes)	Contains buffer start adrs (i.e. 0380)
02FE,F (2 bytes)	Near-end-of-buffer warning adrs; (normally 07C0 for a 2K system; or 0FC0 for a 4K system.)
0292,3 (2 bytes)	Jump to extra control function routine; (e.g. to implement 4x and 5x codes; in CHIP-8, must end with 1208 instr'n)
0260	Main control segment
0294	Mach. code subr to show 3 x 5 dot symbol.
02A5	M/c subr to shift V0 left 4 bits.
02AE	CHIP-8 subr to show cursor.
02C0	CHIP-8 subr for WAIT function.
02D4	M/c subr to reverse video.
02E6	M/c subr to initialize buffer.
02F1	M/c subr to store code in buffer.
0306	M/c subr to fetch code from buffer.
0310	M/c subr to implement ALTER function.
0342 - 0380	free space
00E0 - 00F0	free space

Improved ALPHA DISPLAY Memory dump:-

0200	F90A	02E6	6A01	6B01	4900	122E	22AE	F00A
0210	02A5	8100	F00A	8101	02F1	8010	0000	0000
0220	0000	3F01	122A	6580	F518	22AE	1238	0306
0230	0000	0000	0000	8100	62C0	8122	3100	1260
0240	4016	1250	4020	1254	0294	DAB5	7A04	1208
0250	A0F0	1256	A0F1	DAB5	7A06	1208	0000	0000
0260	8200	630F	64F0	8242	4260	12B8	4270	00E0
0270	4280	12C0	4290	02D4	42A0	0310	42B0	12E0
0280	8032	42C0	8A05	42D0	8B04	42E0	8B05	42F0
0290	8A04	1208	9630	810F	22B3	7EC1	9380	10CE
02A0	007E	7EC1	9896	3048	4848	4897	3039	7BFF
02B0	A0F6	DAB7	7B01	00EE	22AE	7A04	1208	0000
02C0	3900	1208	8032	02A5	8034	F015	F007	3000
02D0	1200	1208	CE01	0063	0008	8C02	0026	F839
02E0	00E0	6900	1202	CE03	8086	77A7	0008	DF1A
02F0	3996	31DE	1AA7	0008	DF1A	7F00	3F8C	07C0
0300	2603	7C00	3F39	DE1A	A600	08DF	1A97	3039
0310	9639	2725	D630	C40F	2602	C610	DE1A	095A
0320	26FC	86A0	09A7	0009	A600	8177	26F9	DF1A
0330	4F97	394C	973A	973B	39DE	1A09	DF1A	7C00
0340	3939							

NOTICE

'DREAMWARE', the source of CHIP8 and DREAM INVADERS, has now ceased operations. This is your last chance to order! (In the future, users who want either of these programs will have to buy a copy from someone who has it.) I hope to contribute further to DREAMer and I hope that micro-processing will continue to be a source of enjoyment for us all. Thank you for your support. (Mike Bauer)

D. A. TRABUCCO,

Have you ever put a Chip-8 program together, then after having a 'test run', realised that a '6XXX' instruction, (or something similar), had been left out? Then came the tedious shifting of statements and altering all the AMMM, 1MMM, 2MMM, 0MMM, and BMMM instructions.

Well, now those days are over, with this unique block shift program. Besides shifting a block of statements, (as little as 2 bytes forward,) it also alters the statements listed above TO SUIT THE NEW POSITION.

PLEASE NOTE, the statement is only altered if the address it refers to is WITHIN THE BLOCK BEING SHIFTED, and the statement is WITHIN THE SAME BLOCK, otherwise it remains unaltered.

Use the program as follows:-

Load source start address into 0100 (MSB) and 0101 (LSB)

Load source finish address into 0102 (MSB) and 0103 (LSB)

Load destination start address into 0104 (MSB) and 0105 (LSB)

RUN, by typeing 0080, FN, 3, and the program does the rest!

NOTES: If there is any data, and/or Machine Code Sub-routines in the block, they will be treated as Chip-8 statements and may possibly be corrupted. The program is fully relocatable, so if you have more than 1K available, put it say at 0480 - 0500. If you do not want to corrupt the display by using part of the display buffer, (0100 - 010D), use 0500 - 050D. (If you do change the workspace area, don't forget to change the instructions at 0080, CE 0100 to CE 0X00, ((CE 0500 if you use the example shown above)), then at 0097, FE 0102 to FE 0X02, at 009C, FF 0106 to FF 0X06, at 00A3, CE 0100 to CE 0X00.

WORKSPACE ASSIGNMENTS

0100 - 0101	Source Address	0106 - 0107	Source buffer register
0102 - 0103	End address	0108 - 0109	Destination buffer reg.
0104 - 0105	Destination	010A - 010B	Difference

010C - 010D Data buffer.

ADDR	INSTR	MNEMONIC	EXPLANATION
0080	CE 01 00	LDX #0100	Point to source Data
0083	A6 04	LDAA #\$4,X	Load MSB of destination
0085	E6 05	LDAB #\$5,X	Load LSB of destination
0087	E0 01	SUBB #\$1,X	LSB destination - LSB source
0089	A2 00	SBCA #\$0,X	MSB destination - MSB source
008B	A7 0A	STAA \$A,X	Store the difference,MSB
008D	E7 08	STAB \$B,X	Store the difference,LSB
008F	EB 03	ADDB #\$3,X	Get the destination, END + DIFFERENCE
0091	A9 02	ADCA #\$2,X	
0093	A7 08	STAA \$8,X	Store destination in buffer
0095	E7 09	STAB \$9,X	
0097	FE 01 02	LDX #\$102	Point to byte of data indicated by END
009A	09	DECX	Adjust pointer to point to first Chip-8 statement to be moved. (END - 2)
009B	09	DECX	
009C	FF 01 06	STX \$106	Save SCE pointer in buffer
009F	A6 00	LDAA #\$0,X	Get MSB of Chip-8 statement
00A1	E6 01	LDAB #\$1,X	Get LSB of Chip-8 statement
00A3	CE 01 00	LDX #0100	Reset pointer to buffers etc.
00A6	A7E7 0C	STAA \$C,X	Save Accumulator A in Data buffer
00A8	27 0E	BEQ 00E8	If Acc. A has #00, then Goto Shift 3.
00AA	94 F0	ANDA # F0	Mask to get Chip-8 code instruction
00AC	81 20	COMA # 20	
00AE	23 23	BLS 00D3	If it is 0MMM, 1MMM, 2MMM, then Tst.2

CHIP - 8 BLOCK MOVE. (Cont)

<u>ADDR</u>	<u>INSTR</u>	<u>MNEMONIC</u>	<u>EXPLANATION</u>
00B0	81 E0A0	COMA # A0	
00B2	27 1F	BEG 00D3	If it is AMMM, then Goto Tst.2
00B4	81 B0	COMA # B0	
00B6	27 1B	BEG 00D3	If it is BMMM, then test further Tst.2
00B8	A6 0C	LDAA #\$C,X	Restore Data to A and continue block move.
00BA	FE 01 08	LDX #\$0108	Get destination pointer from buffer
00BD	09	DECX	Decrement
00BE	09	DECX	Decrement
00BF	A7 00	STAA \$0,X	Store MSB of Chip-8 statement
00C1	E7 01	STAB \$1,X	Store LSB of Chip-8 statement
00C3	FF 01 08	STX \$0108	Save new destination in buffer
00C6	BC 01 04	COMX \$0104	
00C9	2E 03	BGT 00CE	If dest.buffer = destination, then Shift 4.
00CB	7E C3 60	JMP	Finish
00CE	FE 01 06	LDX #\$0106	Point to next source
00D1	20 C7	BRA 009A	Go back to adjust pointer. Shift 2.
00D3	86 0F	LDAA # OF	Mask off Chip-8 instruction bits
00D5	A4 0C	ANDA #\$C,X	
00D7	E1 03	COMB #\$3,X)Find if address specified is below
00D9	A2 02	SBCA #\$2,X)upper limit of block.
00DB	2A DB	BPL 00B8	If it refers above, then Shift 5.
00DD	86 0F	LDAA # OF)Now test low limit. Get MSB address
00DF	A4 0C	ANDA #\$C,X)again
00E1	E1 01	COMB #\$1,X)Find if address specified is above
00E3	A2 00	SBCA #\$0,X)lower limit of block.
00E5	2B D1	BMI 00B8	If it is below, then Shift 5
00E7	86 0F	LDAA # OF)It is within the block, so get address
00E9	A4 0C	ANDA #\$C,X)MSB again. (Acc. A)
00EB	EB 0B	ADDB #\$B,X	Add the difference to the address
00ED	A9 0A	ADCA #\$A,X	
00EF	E7 0D	STAB \$D,X	Store the new LSB in buffer
00F1	C6 F0	LDAB # F0	Get mask into Acc. B
00F3	E4 0C	ANDB #\$C,X	Mask off address portion of old instruction.
00F5	E7 0C	STAB \$C,X	Put it back in MSB buffer
00F7	AA 0C	ORAA #\$C,X	OR the modified address with instruction.
00F9	E6 0D	LDAB #\$D,X	Restore Acc.B with LSB of instruction.
00FB	20 BC	BRA 00B9	Go to main program. (Shift 3)

NOTES: The program is fully Machine Code and consists of the following segments:-
 0080 ; Start, 0083 - 008D ; Difference, 008F - 0095 ; Destination,
 0097 ; Shift 1, 009A - 00A8 ; Shift 2, 00AA - 00B6 ; Test 1,
 00B8 ; Shift 5, 00BA - 00CB ; Shift 3, 00CE - 00D1 ; Shift 4,
 00D3 - 00E5 ; Test 2, 00E7 - 00FB ; Modify.

WANTED.

Here is a selection of things that people have requested appear in the DREAMER. If you would like to try your hand at writing a program, or an article, but can not think of a subject, why not try one of the following?

- | | |
|---------------------------|---|
| - A CHESS program | - A FLIGHT SIMULATOR GAME |
| - DRAUGHTS | - Interfacing the DREAM to external devices |
| - An EPROM Programmer | - Morse Code Decoder. |
| - A LIGHT PEN | - Radio Amateur orientated programs |
| - More 'SERIOUS' programs | - A 'Western Gunfight' game. |

ADVERTISING.

If you would like some help, can offer some help, have something to sell, or would like to buy something, send it in to us with a fee of \$1-00, and we will print it in two newsletters. THIS OFFER ONLY APPLIES TO PRIVATE ADVERTISERS, and we would ask you to keep them reasonably short, something like the ones below. Commercial enterprises who wish to advertise in the DREAMER are invited to contact us for details of rates, etc.

FOR SALE. DREAM 6802 IN CASE. Case has Digitran keyboard fitted and connectors for Joystick Controller, Ext.keyboard, Power, R.F., Spare, and Tape IN/OUT. Joystick controller and Ext.keyboard supplied. Power supply in case, 2 tapes of programs, DREAMER, Issues 1 to 9, THE LOT FOR \$220-00, O.N.O.

ALSO, DICK SMITH MINI SCAMP, Full kit, with case, working, \$70-00. O.N.O.
JEFF BODNICK, [REDACTED]

IF YOU ARE HAVING TROUBLE getting your DREAM up and running or it has died on you and you cannot find out why, write with full details of troubles. Send stamped, addressed envelope for reply. If you think it is too hard for you to find the fault, send \$10-00, (to cover return insured postage and packing), and your DREAM, to SID MOORBY, [REDACTED]

IDEAS.

A LONG ONE

A neat way of providing better access to the EXPANSION BUS and I/O SOCKETS is to extend them, using WIRE-WRAP sockets.

With a bit of adjustment to the spacers holding up your DREAM board, the head of the wire-wrap sockets can be made to protrude through holes cut in the perspex cover. The wires from the keyboard, RAM expansion board etc., are easily soldered to the long pins of the wire-wrapping sockets. (well..... fairly easily.)

NOTE: Before you rush out and buy a handful of sockets, observe the following points:-

1. Not all 16 pin sockets (i.e. the ones you originally soldered on your DREAM board) will accept the chunky pins of most wire-wrapping sockets.
2. Although it sounds easy, cutting rectangular holes in perspex is no fun, and each slip leads to a scratch.
3. The centre of your DREAM board must be supported to stop it flexing when plugging in your Joystick, Directional Paddle, etc. The simplest method of doing this is to cut off a small piece of dowel and place it between the main board and the base of the case.

P.S. While you have got your DREAM apart, why not do those little jobs you have been meaning to do for some time..... like soldering pull-up resistors to the data lines, or adding LEDs to the Tape Input, etc.

R. STUART.

This is an improved version of the program published in DREAMER No.6, February 1981, on how to interface a Creed 75 Teleprinter to your DREAM.

The main reason for the changes is that I was using base-page scratch pads for the program, and had not realised how many people used the locations from 0080-00FF for programming. I have discovered that the bottom of the CHIPOS stack at 0040 up is quite good for transient scratchpad use.

For this revision, put the starting address of the block to be printed at 0030-31, the ending address at 0032-33. (These are Chip-8 variables, in case you want to get cute, and call the printer routine from within Chip-8 itself. When the program finishes, it will shove the printer magnet over to a non-noisy position and then stick in a WAIT state until you push RESET.

Creeds can be set up either as single or double acting. I.E., using either a spring return or without a spring. Experts on these monsters assured me that springless is best, hence my electrical design which shoves the armature forward and backwards by electrical currents in different directions. This is not the same as the usual 20 ma loop, which is merely off or on. The way that you will know that the current is going in the right direction is that while the computer is 'idling' the polarity is such that the Creed printhead is bouncing up and down with a gosh-awful racket. If you then deposit 'OF' into memory location 8012, the printer will quiet down....until you hit RESET.

My particular machine works well with a bit length set by a value of 0780 at location 04A4. Try increasing or decreasing this value in case your machine runs faster or slower.

PRINTER KEY ADDRESSES ARE:-

0400	Main. Controls whole routine
0406	Boundary calculations
0416	Go : Disables TV interrupts, calls lineout
0425	Outputs two spaces
042C	Converts Byte in ACCA to Baudot, outputs it.
043D	Outputs address in Index Register
044C	Outputs Carriage Return, Line Feed, Figures
045C	Outputs full line, invokes above three subroutines
0474	T.V. off
0482	Outputs ACCA, no conversion to Baudot
0494	Outbit, toggles PIA pins
04A1	Delay loop
04AC	Hex to Baudot conversion

0400	86F8	9431	9731	9633	840F	270A	7C00	3326
0410	037C	0032	20F0	8D5C	8D32	DE30	8D3E	9C32
0420	26FA	8D55	3E6D	0086	C88D	5739	3644	4444
0430	448D	7A8D	4D32	640F	5D73	6D46	39DF	4496
0440	44D6	4537	8DE6	328D	E38D	DA39	86D0	8D32
0450	56C4	8D2E	56F6	8D2A	7C00	4639	8DEE	DF46
0460	8DDB	C608	A600	8DC4	A601	8DC0	8DB7	0808
0470	5A26	F139	3636	BDC2	FEC6	OFF7	8012	397E
0480	C360	37C6	0744	8D0C	8D17	5A26	F80D	8D04
0490	8D0F	3339	3724	04C6	UF20	015F	F780	1233
04A0	39DF	42CE	3760	0901	26FC	DE42	39DF	40CE
04B0	04E5	4D27	0408	4A20	FAA6	0001	2B09	7D00
04C0	4827	0B8D	0C20	077D	0048	2602	8D0D	DE40
04D0	3936	86FE	8DAC	7FU0	4832	3936	36F6	EDA2
04E0	7C00	4E32	39EC	EEE6	C2D4	20EA	CECC	F046
04F0	725C	5242	5A3F	16DC	C1FF	3101	1632	C133

M. SPIETH,

Picture yourself in the cockpit of your Astro-Fighter, flying through space. You spot an enemy fighter from behind and try to manoeuvre yourself (using your Joystick), into position behind him. For a split second he flies through your centre screen. You hit the fire button instantly, and success is yours as you watch the four Photon torpedoes hurtle toward the fighter.

The Joystick is used as if you really are in a plane, and as you fly through space, you will see 'space-junk' shooting past. Key F is the 'Fire' key and the game ends after two minutes.

To start the game, run C000, FN, 3. The screen will remain blank, while you settle back and strap yourself into the pilot's seat. Then, press the 'Fire' button to launch yourself into battle.

HELPING HINT: If you find it too hard to hit the target, changing 0254 from 1266 to 0000 will allow you to fire continuously and make the game much easier.

0080	A24B	D121	D141	D561	D581	00EE	A24B	D131
0090	D421	D431	D121	00EE	DAB5	7A05	00EE	C002
00A0	C102	A040	F155	13EC	208C	6E00	10B4	208C
00B0	A3F9	F455	A2BC	02C1	03EF	02C1	4900	10C8
00C0	6603	F618	6900	7D01	3DC0	124C	A3FE	F0E5
00D0	70FF	3000	1234	A3F0	FC33	F265	00E0	6A08
00E0	6B0E	F029	2098	F129	2098	F229	2098	6000
00F0	F029	2098	F00A	6A00	00E0	1202	1200	2626
0200	10F4	6B00	A24B	DAB1	7A01	3A3F	1206	DAB1
0210	7B01	3B1F	120E	DAB1	7AFF	3A00	1216	DAB1
0220	7BFF	3B00	121E	6E00	A2BC	6A1F	6B0D	02C1
0230	6C00	6040	A3FE	F055	209E	611F	620F	6411
0240	6521	660F	6811	A3F0	F855	2080	4E01	1266
0250	620F	E29E	1266	6E01	6102	6201	631E	643D
0260	208C	A3F9	F455	A3F0	F865	2080	71FF	7501
0270	4D00	1284	0001	8205	C001	8404	C001	8605
0280	C001	8804	3100	128C	7D40	128E	2080	7D80
0290	A3F0	F855	3E01	10B4	A3F9	F465	208C	7102
02A0	7201	73FF	74FE	3120	10AE	D121	89F0	D131
02B0	89F4	D421	89F4	D431	89F4	10A8	1010	3806
02C0	3896	3A97	2E96	3B97	2FD6	05DE	2686	0197
02D0	10C4	0F26	0206	1037	DF14	A600	971E	7FB0
02E0	1FD6	2EC4	0727	0974	001E	7600	1F5A	26F5
02F0	062E	8D2E	961E	8D15	D62E	C808	8D24	961F
0300	8D0B	7000	2FDE	1408	335A	26CB	395F	D116
0310	2701	3916	E800	AA00	E700	1127	0486	0197
0320	3F39	4F97	1696	2F81	1E23	037C	0016	8108
0330	2603	7000	1684	1F48	4848	C13F	2303	7000
0340	16C4	3F54	5454	1B97	10DE	1C96	2E81	F823
0350	0796	1F84	7F97	1F39	8100	2607	961E	847F
0360	971E	3981	3822	0139	961E	84FE	971E	3996
0370	404A	9742	9641	4A97	433E	C63B	F760	1306
0380	61F7	8012	063F	F780	1386	FF37	FE97	FF7F
0390	8012	8621	B780	1206	4AB6	8012	4646	2403
03A0	7C00	FE46	2403	7D00	FF5A	26ED	96FE	800A
03B0	2C01	4F43	843F	97FE	96FF	800A	2001	4F43
03C0	843F	0101	D63B	DB43	8D0D	D73B	D63A	96FE
03D0	DB42	8D03	D73A	2981	2023	015C	8134	2301
03E0	5C81	1422	815A	810C	2201	5A39	6D00	00EE
03F0	9104	0716	173C	0B10	1701	1E0F	1021	3B00

BRUCE MITCHELL,

This is a game designed to teach lower primary school children compass directions and the use of co-ordinates.

A grid is drawn, (9 x 9), and somewhere on it is a lode of gold. To have a guess as to where it might be, enter the horizontal, then vertical co-ordinates for your chosen square. The co-ordinates are numbered 1 to 9 from the bottom left hand corner, where the white square is drawn.

Your next move will be made easier if you follow the directions which appear on the screen, but be careful of old mine shafts in the area!

0080	0650	DE30	A680	08DF	30DE	32A7	0008	DF32
0090	5AC1	2827	845D	26EA	390E	01D0	DF32	20E2
00A0	F785	B595	F7F0	9090	9090	8888	A8A8	F860
00B0	F080	F810	F060	2070	A820	2010	F810	2000
00C0	F8A8	A8AA	0300	0130	0350	0130	07DE	0609
00D0	DF06	8601	BDC3	B3DE	06BD	C3C8	8622	BDC3
00E0	E0DE	0608	DF06	BDC3	BDDE	06BD	C3C8	BDC2
00F0	C44D	2B05	BDC3	92A7	0020	D700	C0C0	0000
0200	0000	00E0	6800	6900	A0B1	D891	7801	3813
0210	120A	7902	6800	3914	120A	6800	6901	D891
0220	7802	3814	121E	6800	7802	3913	121E	13F4
0230	22CE	84C0	22CE	85C0	22CE	86C0	22CE	87C0
0240	5460	1248	9570	1238	22D6	A0BA	6800	6918
0250	D895	6806	22E8	8AC0	FC29	D895	A0B6	680E
0260	D895	6816	22E8	FC29	D895	6E12	80C4	7DFF
0270	9EC5	8BE0	8AA4	7AFF	A0B1	DAB1	6F20	22DE
0280	5A40	1288	9B50	13C6	5A60	1290	9B70	13D4
0290	A0A0	6820	D895	682C	R0A5	9B50	12A8	8B55
02A0	3F01	A0B0	D895	7805	A0AA	9A40	12C6	8A45
02B0	6E0E	3F01	FE29	D895	1206	0000	0000	0000
02C0	0000	0000	0000	6FFF	22DE	03A0	124A	23A8
02D0	8CC4	7001	00EE	0000	6F04	FF18	6F10	FF15
02E0	FF07	3F00	12E0	00EE	22D6	FC0A	4C00	12E8
02F0	6E0A	8DC0	8DE5	3F00	12E8	00EE	0000	0000
0300	0A0A	0A0A	E0EE	AEC0	0AAA	0AAA	808A	ARR0
0310	0EAA	0EEA	E0CA	RA00	04AA	0RRA	808A	ARR0
0320	04EE	0AA4	E08E	EA00	0ABB	BA83	B63B	A300
0330	0AAR	9282	A022	A280	0AAB	9382	B02A	A280
0340	0AAB	1282	A02A	A280	053A	9283	A03B	B620
0350	5750	5757	0774	4770	5550	5554	0454	4450
0360	7550	7757	0674	4750	2550	5554	0454	4450
0370	2770	5527	0457	7750	0CEA	B838	1A8B	B800
0380	0ARR	A828	22AA	1000	0AAA	A838	13BB	1000
0390	0AAA	A828	0AAA	1000	0CE5	2828	32AA	1200
03A0	8600	CE01	C07E	C07D	CC0F	6E09	8CE5	3F00
03B0	13A8	8CE4	00EE	23A8	FC29	D895	7804	7AFF
03C0	3A00	13B6	00EE	A0C4	23E4	A0AF	D897	7805
03D0	23B6	13E0	A0C8	23E4	23B6	A0BF	7806	D895
03E0	FF0A	1200	F365	00E0	03FD	CA07	7R01	6810
03F0	6910	00EE	A0FC	6914	D893	1230	007E	0000

IAGO FOR TWO

(0200 - 0400)

M. COLLINS

This program is taken from the old board game, REVERSI, and allows two players to compete against each other.

The starting turn is chosen randomly, and illegal moves will not be accepted. The object of the game is to capture your opponents pieces, (by reversing them) by flanking a continuous line of his pieces, in any directions, with your own pieces. Have fun learning the game by playing it.

0200	0000	0000	2260	2288	6202	6402	22AA	CE01
0210	2200	6508	8A60	4600	1300	8652	3600	1318
0220	2200	8B60	8652	3600	1318	132E	68F8	69FC
0230	236A	4700	1314	8CA0	8D80	22D4	360F	1316
0240	22AA	68F8	69FC	236A	4908	1252	23A2	23B2
0250	1246	23D2	22AA	8230	8244	4240	13DE	1300
0260	A3EC	6000	6D00	DCD4	7C04	3C20	1266	6000
0270	7D04	3D20	1266	6C00	6D10	A3F8	DCD4	DD04
0280	A3F4	DCD4	DCD4	00EE	A3F0	6C27	6D08	DCD4
0290	A3EC	DCD4	6D15	DCD4	A3F4	DCD4	00EE	F218
02A0	2302	72FF	3200	129E	00EE	8230	6533	6807
02B0	22B8	8240	6533	6614	A300	F233	A3CD	F165
02C0	F029	D565	7504	F129	D565	00EE	6C27	6D08
02D0	4E01	6D15	22F6	DCD4	6600	E6A1	12EC	7601
02E0	3610	120A	DCD4	6210	2302	12D6	DCD4	6210
02F0	F218	2302	00EE	3E00	12FE	A3F8	00EE	A3F4
0300	00EE	F215	F007	3000	1304	00EE	6501	7E01
0310	8E52	1210	22F6	DAB4	6206	229E	1210	A3FC
0320	1324	A3EC	6F00	DCD4	82F0	DCD4	00EE	8AA4
0330	8AA4	8BB4	8BB4	8CA0	8D80	2322	3201	1318
0340	22F6	DAB4	1220	6700	8CA0	8D80	2388	420F
0350	1368	2322	4201	1368	231E	82E5	4200	1364
0360	7701	1340	3700	00EE	6700	2372	3908	1346
0370	00EE	7804	4800	1382	3808	00EE	68FC	7904
0280	00EE	5890	00EE	1372	6208	8C84	8D94	65E0
0390	86C0	8652	3600	620F	86D0	8652	3600	620F
03A0	00EE	3E00	13AC	8374	8475	00EE	8375	8474
03B0	00EE	22F6	8CA0	8D80	2388	6204	F21E	DCD4
03C0	22F6	DCD4	77FF	3700	13B8	00EE	0000	0400
03D0	0000	3E00	13DA	7301	00EE	7401	00EE	6208
03E0	229E	22AA	6210	2302	22AA	13DE	0020	0000
03F0	2070	2000	7070	7000	2070	2000	5000	5000

IAGO FOR ONE

(0080 - 0400)

M. COLLINS,

This version allows one person to compete against the computer.
 The DREAM is a slow thinker, but don't be deceived - it is tricky!

The rules and key functions are the same as for 'IAGO for TWO'.
 Firstly, load the two player version, then add the following:-

0080	3E00	12CC	6100	601C	20A0	20B4	6108	6014
0090	20A0	20B4	6104	6018	20B4	20A0	660C	00EE
00A0	8A10	8B10	20DA	8A00	20DA	8B00	20DA	8A10
00B0	20DA	00EE	8A10	8B10	7B04	20DA	5B00	10B8
00C0	5A10	10C8	8A00	10B6	8A10	7A04	20DA	5A00
00D0	10CA	5B00	00EE	8B10	10C8	68F8	69FC	23EA
00E0	4700	00EE	8CA0	80B0	2322	3201	00EE	A3F0
00F0	DAB4	03CC	1236	82F0	00EE	54D5	F800	0000

Then, make the following changes:-

0210 to 2080, 02B8 to A0FD, 02BC to A0FE, 03CC to CE00,
 03CE to 5FDF, 03D0 to 2439.

DICE GAME

(0080 - 0400)

K. SEMRAD,

You may bet from 1 to 10 units. (Key 'A' = 10 units.) You then nominate OVER 7, (Key 'A'), or UNDER 7 (Key 'B'). The bank wins if a SEVEN is thrown. Clear the table for the next throw with Key '0'.

00A0	0000	0000	0000	0000	0000	E92E	4BDA	F6DE
00B0	F6DA	0000	F248	F6DE	C546	492E	0380	93DE
00C0	B8DE	F3CE	C546	C546	0000	F24E	0000	7000
00D0	0020	7020	0000	FEFE	FEFE	FEFE	FEFE	BEFE
00E0	FEFE	FAFE	FEFE	FEFE	FEFA	FEFE	BAFE	FEFE
00F0	BAFE	FEBA	FEFE	FEBA	FEFE	BAFE	BAFE	BAFE
0200	647D	6A00	6B00	6C1F	A0BD	DAB1	DAC1	7A01
0210	3A32	120A	6A00	6B01	6C31	DAB1	DCB1	7B01
0220	3B1F	121A	23AA	FE0A	3E00	1232	61A0	03F4
0230	1226	0308	F90A	390A	490B	1240	03F4	1234
0240	6A29	6B03	6C29	6D15	239E	3502	125A	7AFE
0250	2390	3F01	1278	2394	12BE	3500	126A	7BFE
0260	2390	3F01	1278	2394	7B02	7B02	2390	3F01
0270	1278	2394	7BFE	125E	239E	3502	128A	7CFE
0280	23EA	3F01	12A8	23EC	12B2	3500	129A	7DFE
0290	23EA	3F01	12A8	23EC	7D02	7D02	23EA	3F01
02A0	12A8	23EC	7DFA	128E	6103	03F4	23EC	2394
02B0	1248	7C02	23EA	3F01	12CC	23EC	12B2	7A02
02C0	2390	3F01	12CA	2394	1200	23EA	8200	8234
02D0	6123	8215	4F00	12EA	3200	12F8	2370	84E5
02E0	4F00	12FE	1336	00E0	1202	490A	12DC	2374
02F0	84E4	3F00	1324	12E4	490B	12DC	12EE	00E0

(See the bottom of Page 22 for 0300 to 0400)

R. KABZINSKI. S.A.

The computer produces sums using random numbers from 1 to 9, for answers up to a value of 9. Once the sum is displayed on the screen, the 'student' is allowed two tries at obtaining the correct answer, via the keyboard. If correct, a smiling face appears on the screen for 'reinforcement', and the score is incremented. If wrong twice, the computer displays the correct answer.

New sums are produced until key 'F' is pressed. When this is done, the computer displays the score, the number of sums attempted on the left, and the number of correct answers on the right. Tones are used during the running of the program for indication of progress.

0200	6500	6600	6702	680A	CB07	CD07	C102	C202
0210	81B4	82C4	8810	8C20	8BC4	8AB0	8B85	3F00
0220	1208	6B10	6C0A	F129	DBC5	6B17	A350	DBC5
0230	6B20	F229	DBC5	6B28	A355	DBC5	1204	A372
0240	DBC1	FA29	6B31	6C0A	DBC5	22FC	6D1E	22EE
0250	00E0	1254	6105	6206	6307	6415	6B16	6C17
0260	6D1D	6E25	A35A	D420	A36A	D4B5	A370	DD10
0270	A380	DD47	A288	DE30	A298	DEC2	7601	132A
0280	6B2F	6C10	A372	DBC1	6B31	6C0A	F429	DBC5
0290	6E2E	0320	6D10	22EE	DBC5	6B2F	6C09	A39A
02A0	DBC7	6E2E	0320	6D31	22EE	A39A	DBC7	77FF
02B0	3700	12F8	6B31	6C0A	FA29	DBC5	230E	6D6E
02C0	22EE	132E	6E00	6B2F	6C10	A372	DBC1	6D02
02D0	22EE	7E01	4E0A	120E	A372	DBC1	1206	F40A
02E0	340F	12E6	1336	7501	94A0	123E	1280	FD15
02F0	FD07	3D00	12F0	00EE	75FF	1204	6C00	6B02
0300	FB18	6D02	22EE	7C01	3C08	12FE	00EE	6C00
0310	6B04	FB18	6D06	22EE	7C01	3C03	1310	00EE
0320	F600	3ED7	21C6	407E	C2E5	6D30	22EE	00EE
0330	6B06	FB18	1204	00E0	6E05	1330	630C	640E
0340	6710	6B13	6926	6A28	6B2B	6C30	6D09	13A2
0350	2020	F820	2000	F800	F000	0107	1C30	60C0
0360	0606	C000	4060	2233	1918	0C06	0703	0100
0370	78FE	FF00	0000	0001	0100	3C30	0001	0386
0380	FC78	0103	87FE	F000	90E0	3C18	0C80	8C00
0390	0C08	1810	3060	6000	8080	FFFF	FFFF	FFFF
03A0	FF00	A3DA	D3E9	A3E4	DAE8	A3EC	DCE6	A3F2
03B0	F533	F265	F029	DD75	F129	D475	F229	D875
03C0	A3F5	F633	F265	F029	D975	F129	D875	F229
03D0	DC75	6DFF	22EE	00EE	1200	3C42	8181	8181
03E0	4R3C	0200	3844	8282	FE82	8282	0102	0488
03F0	5020	0000						

* * * * *

DICE GAME

(Cont)

0300	6A00	6B10	6001	031C	DAB5	7A04	7001	3012
0310	1306	F00A	300C	1312	00E0	1200	9630	CE00
0320	A87E	C198	00E0	6A08	6B10	6000	031C	DAB5
0330	7A04	7004	1306	F00A	3000	1336	00E0	1202
0340	137A	4602	1380	4603	1384	4604	1388	4605
0350	138C	1390	236E	87A0	4701	1394	4702	139A
0360	4703	13D0	6CFE	9C04	8014	13A6	13AA	CA07
0370	A0CC	1376	A0D0	670A	6633	D674	7605	A080
0380	FE33	F265	F129	D675	7604	F229	D675	00EE
0390	03D1	8300	8030	A0D6	F01E	DAB7	00EE	C507
03A0	3500	4501	00EE	6502	00EE	6703	6634	A3FD
03B0	F433	F265	F029	23C2	F129	23C2	F229	23C2
03C0	00EE	D675	7604	00EE	860A	913E	2202	973E
03D0	39B0	C132	8105	22F9	1648	4648	1097	3039
03E0	03D1	A0D6	F01E	DAB7	00EE	8201	A8D6	F01E
03F0	DCD7	B9EE	C640	9E01	9721	7EC2	E501	0105

C. FETHERS.

This program is intended to be similar to the motor bike games in some pinball parlours.

At the bottom of the screen, a motor bike with rider is displayed. At the top of the screen, Velocity, Distance, and Time are shown.

Player accelerates and brakes bike with Keys D and E. The wheels rotate and ground moves, with exhaust noises. You can change gears by releasing the accelerator briefly. If the bike accelerates too hard, it rides up on its back wheel. Obstacles appear in front of bike, which must be jumped over, otherwise bike crashes. (On crash, bike tips forward, and engine coughs and splutters.)

Key F (for 'foot') may be used to edge the bike backwards when it is stopped, if it is too close to jump over an obstacle. Do not press F unless the obstacle is visible.

After 120 miles, the program stops.

0080	8601	B780	12BD	C297	CE00	30D6	3C96	1826
0090	2C96	1781	0D26	156F	CA5C	D1FE	2304	D6FE
00A0	202E	6C0D	2A30	867F	973D	202A	810E	2607
00B0	6F0D	C002	2407	5F81	0F26	0297	376C	CA96
00C0	FA81	1923	0B17	8B50	C1A4	2302	86F4	97FE
00D0	6A0D	2A02	6F0D	179B	FF97	FF24	0886	40B7
00E0	8012	5A97	37D7	3C7E	C3E9	A3B2	601B	DB04
00F0	7B01	DB04	F018	6700	1236			
0200	6300	6D00	6C00	6BFF	6500	6901	6800	6700
0210	6400	232E	611E	A3E5	6000	D012	7008	3000
0220	121A	A0FA	F555	A000	6100	F918	F155	231C
0230	2300	239A	6E00	F007	6128	8015	3F00	1248
0240	231C	7301	231C	0000	3000	1254	2354	6901
0250	470F	10EA	95C0	125E	2300	85C0	2300	8000
0260	3800	1276	7008	80D5	4F00	3A14	1288	2332
0270	6801	2344	1288	3801	1288	70F8	80D5	3F01
0280	1288	2344	6800	2332	370F	4700	1236	23A0
0290	6700	2354	7902	6006	8902	2354	6000	611F
02A0	A3E5	D011	7008	3040	12A2	A3B2	611B	3BFF
02B0	12BC	C01F	3000	1236	6B3C	12C4	DB14	7BFF
02C0	4BFF	1236	DB14	3B0E	12D2	3801	1388	6803
02D0	1236	3B0B	4B0A	12FS	4B09	12F0	4B04	12EA
02E0	3B00	1236	23B6	232E	1236	2332	23B6	1236
02F0	2344	7AFF	2330	1236	2344	7AFF	2344	1236
0300	03DC	6600	2314	6000	F129	D605	7604	F229
0310	D605	00EE	A0FB	F033	F265	00EE	8030	6658
0320	F615	2314	F029	6F00	D6F5	7604	1306	6A14
0330	6800	6005	A3F1	D0AA	81A0	7105	A3FB	600D
0340	D015	1354	A3E7	6007	D0A6	81A0	6006	A3ED
0350	7106	D014	81A0	A3D4	F91E	6005	3800	136A
0360	7107	D012	600B	D012	00EE	4802	137C	7107
0370	6007	D012	71FC	600B	D012	00EE	7103	D012
0380	6009	7104	D012	00EE	2330	6901	23B6	F007
0390	CCFF	3059	138E	00E0	1200	8040	6633	1322
03A0	6020	9E04	3F01	00EE	239A	7401	3478	139A
03B0	239A	F0F0	F000	6005	6A14	A3C8	D0AA	6802
03C0	600D	A3D2	D0A2	1354	0131	7B9F	9D7F	1609
03D0	0906	8080	2000	0020	0040	4000	9635	4747
03E0	843F	9730	39FF	AAC0	C0BE	D9F9	F6FC	C848
03F0	300C	0C18	1E39	FF6D	9E9A	6180	8040	4080

HOW TO SUBMIT PROGRAMS

To remain in operation, we need a constant supply of new programs, and articles about the DREAM 6800. If you can write an article on modifications you have made to your DREAM, or the use you are making of it, or if you have written any games, or utility programs, we invite you to submit them to us for consideration. ALL CONTRIBUTORS OF PROGRAMS PRINTED WILL RECEIVE VOUCHERS FOR TWO FREE NEWSLETTERS. CONTRIBUTORS OF ARTICLES AND IDEAS PRINTED WILL RECEIVE FROM ONE TO THREE VOUCHERS, BASED ON THE GENERAL INTEREST CONTENT OF THE ARTICLE, AND THE AMOUNT OF WORK THAT HAS GONE INTO IT. Along with the listing for all programs submitted, we will need a tape recording, with at least twenty seconds of High and Low "leader" on it. We need a leader to align our tape heads, and tune the DREAM input port. To do this you first must record 20 Sec High tone, then 20 Sec Low tone. The High tone is normal leader, and can be recorded normally. To get the Low tone, load in the following Machine Code program.

```
0200    8640 Accumulator A = 40  
0202    B78012 Store in PIA output port.  
0205    20FE Branch back 2 bytes from 0207  
0207    0000
```

This will produce a continuous Low tone when run 0200, FN, 3. After 20 seconds press RESET to return to normal. Then load your program. We need the electronic copy so we can test the program and verify the listing BEFORE printing, to eliminate program errors and increase the enjoyment of other users.

We will not be able to enter into correspondence, but will print corrections or improvements where necessary. We will not be selling tapes.

Programs submitted for consideration should be typed, for clarity, and set out in the following format:-

- 1) Program name and memory location.
- 2) Your name and address. (If you do not wish to receive any correspondence from other users, omit your address.)
- 3) The program explanation. (Don't forget key functions)
- 4) The program listing, typed single space. (If in doubt, have a look at the way the programs in this issue have been typed, and copy the format)

Following the guidelines set out above lets us check out the programs submitted quickly and easily. If you do not have access to a typewriter, we will accept a handwritten listing, providing it is LEGIBLE, and accompanied by a tape. However, if we cannot read your writing, and the tape will not load, or has 'bugs' in it, there will be no way we can check the program, and it will not be considered.

That's all there is to it, so send us in your favourites, and don't forget, for each one we use, you get vouchers for two newsletters free of charge. Should you be a prolific programmer, and accumulate some surplus vouchers, or have already paid a subscription to the newsletter, we will redeem the vouchers at a rate of six vouchers for \$15-00.

PRICE STRUCTURE

The cost of this newsletter is \$3-00 per issue. An advance subscription is available at reduced cost. Please write for details of cost and length of time remaining in current subscription period.

BACK ISSUES. Copies of all newsletters from No.1, September, 1980, are available at a cost of \$4-00 each, from:-

N.S.W. 6800 USERS GROUP,
[REDACTED]

(Please add -10c to all CHEQUES sent from outside N.S.W., to cover Stamp Duty charged by N.S.W. Government. This is only required on cheques and does not apply to Money Orders etc.)
