

GM 816
MULTI-I/O

The Gemini MultiBoard Microsystem



MULTI-I/O

80-BUS MULTIPLE
INPUT/OUTPUT BOARD

Installation Manual
and User Guide

GM816
Issue 2
22/12/83

T A B L E O F C O N T E N T S

1. Introduction.....	2
1.1. Guarantee.....	2
2. Installation.....	3
2.1. I/O Port Addressing.....	3
2.2. LKS1.....	3
2.3. LKS2.....	4
2.4. Standard Port Addresses.....	5
2.5. Implementing /NASIO on a Nascom system.....	5
3. Connectors.....	6
3.1. PIO Connectors (PL2, 3 & 4).....	6
3.2. CTC connector (PL1).....	6
3.3. Expansion Connectors (PL 5 & 6).....	7
4. The Real-Time-Clock.....	9
5. Using-the Real-Time-Clock.....	9
5.1. Loading the Clock.....	9
5.2. Reading the Clock.....	11
5.3. Register Summary.....	12
5.4. Examples of Use.....	12
5.4.1. BASIC Set/Display Program.....	13
5.4.2. Machine Code Set Program.....	16
5.4.3. Machine Code Display Program.....	18

1. Introduction

The Gemini GM816 is an Input/Output board for use in Gemini MultiBoard and Nascom microcomputer systems utilising the 80-BUS/Nasbus bus structures. The board provides three Z80 PIOs, a Z80 CTC and a 58174 Real Time Clock with battery back up. Flexible I/O port decoding is provided to allow more than one board to be used in a single system.

A special feature of the GM816 is its ability to have a daughter board attached in a "piggy-back" fashion. This is made possible by an on-board I/O bus that provides the necessary buffered address, data and control lines along with a number of I/O port decodes provided by the main board logic. Available daughter boards include a proto-typing board and a dual-channel serial I/O board.

This manual contains all of the information necessary to implement the GM816 and provides programming examples for using the Real Time Clock. MK3881 PIO and MK3882 CTC manuals are supplied separately.

1.1. Guarantee

Your GM816 I/O board is guaranteed by the supplier (your Dealer) for one year from the date of purchase. However, being a system module, any faults attributable to the incorrect implementation of the board within your system will be fully chargeable as to both parts and labour. The guarantee extends as far as the original hardware as supplied and no work on GM816 I/O boards modified in any way will be carried out.

Any queries regarding the implementation and operation of your GM816 I/O board should be directed at your Gemini dealer.

80-BUS, MultiBoard and RP/M are trademarks of Gemini Microcomputers Ltd. Nasbus is a trademark of Lucas Logic Ltd. (Nascom Microcomputers Division). CP/M is a trademark of Digital Research, Pacific Grove, California. Nas-Sys is copyright (c) R. Beal, Surrey.

2. Installation

Carefully unpack your GM816 and examine it for any mechanical damage. In the event of any damage please inform your dealer immediately.

Your GM816 will have been shipped to you fully tested and working. However, to prolong the life of the clock back-up battery the 58174 clock chip has been removed. This should be inserted into the IC18 position on the board, taking care to ensure that it is inserted the correct way round and that all the pins enter the socket correctly. All that should now be required is for the board to be plugged into the bus. However, as there are a number of link options on the GM816 it should prove useful to read through this manual carefully first.

When plugging the GM816 into the bus please take great care, excessive force should not be required, any difficulty that is encountered will in all probability be due to the keyway of the edge connector not slotting accurately into the slot in the edge of the board. Ensure that the board is plugged in with the edge connector going in first and the correct way around, it should not be possible to plug the board in the incorrect way around because of the keyway. Power is connected to the board through the bus - refer to the 80-BUS or Nasbus specification for further details.

2.1. I/O Port Addressing

The two link blocks, LKS1 and LKS2, are used for setting up the Z80 I/O ports that the GM816 board's devices will occupy. The 58174 Real Time Clock (RTC) requires 16 I/O ports, and the CTC and three PIOs each require 4 I/O ports.

2.2. LKS1

LKS1 provides 16 decodes of 16 ports each. There are also 4 outputs from LKS1 to provide the necessary chip selects via LKS2 to the GM816s I/O devices.

Ports	A0 - AF	PSA	11	10	PS9	Ports	90 - 9F
"	B0 - BF	PSB	12	9	PS8	"	80 - 8F
"	C0 - CF	PSC	13	8	PS7	"	70 - 7F
"	D0 - DF	PSD	14	7	PS6	"	60 - 6F
"	E0 - EF	PSE	15	6	PS5	"	50 - 5F
"	F0 - FF	PSF	16	5	PS4	"	40 - 4F
To LKS2 Section B			17	4	PS3	"	30 - 3F
To LKS2 Section A			18	3	PS2	"	20 - 2F
To RTC Chip Select			19	2	PS1	"	10 - 1F
To /NASIO (bus line 12)			20	1	PS0	"	00 - 0F

Pin 20 provides a block decode onto line 12 of the bus. This line is /NASIO and is only required by a Nascom (see below).

Pin 19 provides a block decode for the RTC. The GM816 has been shipped with this pin connected to pin 3 (PS2). This places the RTC at ports 20H to 2FH in the I/O map.

2.4. Standard Port Addresses

The GM816 is supplied with the links set to provide the following port decodes:

Device	Ports
/NASIO	00 - 0F
CTC	10 - 13
PIO1	14 - 17
PIO2	18 - 1C
PIO3	1C - 1F
RTC	20 - 2F

2.5. Implementing /NASIO on a Nascom system

If the GM816 board is to be used with a Nascom then a /NASIO signal must be provided to the Nascom. This is because the Nascom I/O ports are not fully decoded.

With Nascom 1 the I/O internal/external link (LK1) should be set to external and, because of a decoding error on Nascom 1, the on-board PIO (IC 35) must be removed.

With Nascom 2 the I/O internal/external switch (LSW2/8) should be set for external operation.

There are a number of 80-BUS/Nasbus expansion boards that can supply a /NASIO signal, but only some boards provide a full decode. If there is more than one board with a /NASIO output in the system then only the board with the most complete implementation should supply the signal, and the other boards should have this line disconnected.

The following list shows a number of boards from the fullest implementation of /NASIO to the least:

Board	/NASIO implementation
EV IEEE488	Open collector, 4 or 8 port decode
Nascom I/O	"
Gemini I/O	"
" IVC	"
" FDC	"
" EPROM	"
Arfon Speech	128 port decode
Nascom RAM B	"
(with Page Mode)	"

Note: The /NASIO signal is only required by Nascom Microcomputers as the I/O on these boards is not fully decoded. In a system controlled by a Gemini GM811 or GM813 CPU board the /NASIO signal is not required and may be omitted.

Pin 18 provides a block decode to section A of LKS2 for the CTC and PIOs. The GM816 has been shipped with this pin connected to pin 2 (PS1). In conjunction with the linking of LKS2, this places the:

CTC at ports	10 - 13
PIO1 "	" 14 - 17
" 2 "	" 18 - 1B
" 3 "	" 1C - 1F

Pin 17 provides an alternative block decode to section B of LKS2 for the CTC and PIOs. The GM816 has been shipped with no connection to this pin.

2.3. LKS2

		/NMI (line 21 of bus)		/NMI		Interrupt output from RTC	
8	9	A1	Decode (Ports X0 - X7	7	CS1	"	Chip select for CTC
10	A2	X4 - X7	(X) from	5	CS2	"	"
11	A3	"	" X3 - XB	4	CS3	"	"
12	A4	"	" XC - XF	3	CS4	"	"
13	B1	Pin 18.	(Ports Y0 - Y3	2	B4	Ports Y3 - YB	Decode (Y) from
14	B2	"	" Y4 - Y7	1	B3	YC - YF	LKS1 Pin 17

3. Connectors

There are four I/O connectors on the GM816 board edge for the three PIOs (PL2, 3 & 4) and the CTC (PL1). In addition there are two connectors for expansion of the I/O board (PL5 & 6).

3.1. PIO Connectors (PL2, 3 & 4)

The PIO (Parallel Input/Output Controller) has two eight bit data ports, one known as A and one as B, so the I/O bits are numbered A0 to A7 and B0 to B7. A0 and B0 represent the least significant bits, A7 and B7 represent the most significant bits. In addition each port has two handshake lines, one for input and one for output.

Below are the details of the connectors on GM816. +5 volts and ground are provided on the connector to drive a small amount of external circuitry. However, users should beware of drawing excessive currents. Full details on the operation of the PIOs can be found in the MK 3881 PIO manual supplied.

B5	1	2	B4
B6	3	4	B3
B7	5	6	B2
ARDY	7	8	B1
/BSTB	9	10	B0
/ASTB	11	12	BRDY
A0	13	14	N.C.
A1	15	16	GND
A2	17	18	GND
A3	19	20	+5V
A4	21	22	+5V
A5	23	24	A7
A6	25	26	N.C.

Note : On the GM816 pins 14 and 26 are 'No Connection'. This is different to the Gemini GM811 and GM813 boards. These lines are freed to allow the user to run additional signals along the PIO cable if required.

3.2. CTC connector (PL1)

The CTC four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the Z80A CTC satisfy most common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

Bus clock	1	2	N.C.
N.C.	3	4	N.C.
ZC/TO2	5	6	N.C.
ZC/TO0	7	8	ZC/TO1
GND	9	10	CLK/TRGO
CLK/TRG1	11	12	CLK/TRG2
CLK/TRG3	13	14	N.C.
N.C.	15	16	+5V

3.3. Expansion Connectors (PL 5 & 6)

The Gemini GM816 I/O board is capable of accepting a daughter board "piggy-backed" onto the main board. There is an on-board I/O bus that provides the necessary buffered address, data and control lines along with a number of I/O port decodes provided by the main board logic.

PL5

+5V	1	2	+12V
-12V	3	4	POWER FAIL
-5V	5	6	WR
D7	7	8	CLOCK
D6	9	10	SEL
D5	11	12	BUSY
D4	13	14	ON
D3	15	16	M1
D2	17	18	IORQ
D1	19	20	RD
D0	21	22	IEI
DBDR	23	24	IEO
PSE	25	26	INT
PSD	27	28	PSA
PSC	29	30	PS9
PSB	31	32	PS8
A3	33	34	PS7
A2	35	36	PS6
A1	37	38	PS5
A0	39	40	PS4
A7	41	42	PS3
A6	43	44	PS2
A5	45	46	PS1
A4	47	48	PS0
GND	49	50	GND

The signals have the following specifications:

Signal name	To/from daughter board	Description
DO - D7	T/F	Z80 data bus.
AO - A7	T	Z80 address signals.
PS0 - PSE	T	Port select decodes. (Note PSF is present on PL6)
DBDR	T	When high the GM816s data buffer is enabled from the bus into the I/O board, when low data from GM816 is output to the bus.
Clock	T	Bus master clock.
RD	T	Z80 read signal.
WR	T	Z80 write signal.
IEI	T	Interrupt daisy chain input.
IEO	T	Interrupt daisy chain output.
INT	F	Z80 interrupt line.
IORQ	T	Z80 I/O request signal.

Signal name	To/from daughter board	Description
M1	T	Z80 M1 signal, gated with the bus RESET line.
Power Fail SEL	T F	Bus power fail warning line. Signal from daughter board indicating that on-board devices are selected.
BUSY	T	Signal indicating that devices on I/O board or daughter board are selected.
ON	F	Activates I/O board wait state generator.
+12v) + 5v) GND) - 5V) -12V)	T	Power supply lines for daughter board.

PL6 connector

The PL5 connector described above provides all the necessary signals for the user to implement any I/O requirement on the daughter board. PL6 provides the remaining bus address lines (A8 to A15) and the Z80 memory request signal (/MREQ) should the user wish to add memory devices to the board. Note that these signals are direct from the bus and any load should meet the relevant 80-BUS/Nasbus specification.

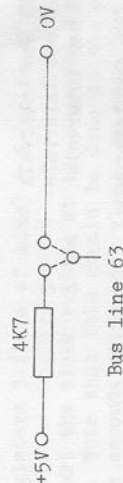
A15	1	2	N.C.
A14	3	4	N.C.
A13	5	6	N.C.
A12	7	8	N.C.
A11	9	10	N.C.
A10	11	12	N.C.
A9	13	14	PSF
A8	15	16	MREQ

4. The Real-Time-Clock

A trimmer capacitor (C5) is provided for fine adjustment of the clock. This should be adjusted over a period of time for minimum error.

Circuitry is provided on the GM816 to switch the read, write and chip select signals during power-on and power-off of the board. Should it be found that the clock data corrupts then it may be necessary to implement the bus power-fail warning line (bus line 63). This may be done in several ways:

1. A switch on bus line 63 to switch the line between +5V and ground.



The switch should be turned on prior to switching off the computer system, and switched off after the computer has been turned on again.

2. Some power supplies, especially switch mode types, have a TTL compatible Ready line that goes high to indicate that all voltages have reached their correct values, and goes low to indicate that one or more power lines are dropping. This line can be connected to bus line 63.

3. With linear power supplies a comparator can be fitted prior to the reservoir capacitors to detect a failing supply and provide an output to bus line 63.

5. Using the Real-Time-Clock

The MM58174 real time clock provides clock and calendar functions with a battery back-up facility. An interrupt timer is included which can be programmed to have one of three interrupt frequencies or alternatively data can be directly read from the clock registers by polling.

The internal registers are arranged as 4-bit nibbles, these occupying the lower 4 bits only of the 8 bit bytes which constitute the Z80 I/O ports for the clock. For registers which do not require all 4 bits (e.g. day of week uses only three bits) the unused bits are not recognised during a write operation and are logical '0' during a read operation. However, the upper nibble of the Z80 I/O port contains random data and must be ignored during read operations.

5.1. Loading the Clock

Data can be loaded as follows:

1. Switch off the clock by outputting 0 to register 14. This also has the effect of zeroing the ten, units, and tenths of seconds counts (these registers are read only and cannot be loaded directly).

2. Set non-test mode by outputting 0 to register 0. Test mode is used in production testing of the clock chip and for normal operation the clock chip must be in non-test mode.

3. Set interrupt mode. Interrupts are controlled by the interrupt latches in register 15. Register 15 in write mode enables the interrupt output and dictates the frequency of interrupts.

FUNCTION	DATA
No interrupt	0 or 8
Single interrupt at 0.5 sec intervals	1
Single interrupt at 5.0 sec intervals	2
Single interrupt at 60 sec intervals	4
Repeated interrupt at 0.5 sec intervals	9
Repeated intervals at 5.0 sec intervals	A
Repeated interrupt at 60 sec intervals	C

All interrupt frequencies are +/- 16.6 mS.

If a single interrupt mode has been selected the timer is reset at the completion of the selected timing period and must be set by software if a subsequent interrupt is required. Setting a repeated interrupt mode allows automatic repeated timer inputs starting after the next clock chip read following an interrupt status read. Interrupts should be initialised by applying the reset condition and reading register 15 three times.

Reading register 15 gives the interrupt status, D3 set to 1 (giving a value greater than 4) indicates that an interrupt has occurred, and D3 is then reset to 0 by the reading process. The next clock chip read automatically restarts the interrupt timer if in continuous mode.

INTERRUPT STATUS	REGISTER 15 DATA
No interrupts	0
0.5 sec	1 or 9
5.0 sec	2 or A
60 sec	4 or C

4. Load the date and time registers with the appropriate information.

Register 13 (write only) is the year status register. This is used to hold leap year information and is loaded as follows:

YEAR	DATA LOADED
Leap year	8
Leap year + 1	4
Leap year + 2	2
Leap year + 3	1

Any data loaded other than the values above can cause spurious operation. The year status is updated every year on the 31st December.

Registers 12 (tens of months) & 11 (units of months) are the month counters, the month being in the range 1 - 12.

Register 10 is the day of week counter and is in the range 1 - 7.

Registers 9 (tens of days) & 8 (units of days) are the days counters. These counters count up to 28, 29, 30 or 31 days depending on the month counters and year status register.

Registers 7 (tens of hours) & 6 (units of hours) are the hours counters. Both count in 24 hour mode.

Registers 5 (tens of mins) & 4 (units of mins) are the minutes counters.

5. Start the clock running at the required time by outputting 1 to register 14. Note that this should be done at the start of the required minute as the seconds count will always be zero (see paragraph 1 above).

5.2. Reading the Clock

The date and time may be obtained by reading the appropriate registers including the tens of seconds (register 3), units of seconds (register 2) and tenths of seconds (register 1). However, when a register has been updated all data outputs from the clock are set to a '1' indicating that the clock value has changed since it was last read. This data changed indicator is reset by any clock read operation. Upon reading back a data changed indicator (F Hex.) it is necessary to re-read not only that register but ALL the registers required, since the change may have affected more than one register (i.e. consider updating 23:59:59.9 on Friday 31st December!). These data changed indicators occur every tenth of a second so the software which is reading the clock must either guarantee to read all the registers it needs within that time or it must only attempt to read a register when it knows it cannot update (e.g. the software has up to 1 minute to read the units of minutes register immediately after it has just altered). This problem is particularly apparent when using a high-level interpreted language which tend to be slow anyway - a possible solution here is to use a machine code subroutine if the language itself is not fast enough.

Note that since only the lower 4 bits of the data bus are used by the chip the top 4 must be masked out since they are invalid. This can be achieved by ANDing the byte read back from the clock chip with 0FH or by MODing it with 16 (divide by 16 and the remainder of the division is the value required).

If a register tries to update during a read operation the data is prevented from updating and a subsequent read will return the data changed indicator (15 Hex.). This means that the clock could be slowed down by reading it very frequently as could happen if the software was sitting in a machine code loop reading the clock constantly.


```

550 '
560 ' Calculate the address of MC$ and POKE in the machine code program
570 MC=PEEK(VARPTR(MC$)+1)+256*PEEK(VARPTR(MC$)+2)
580 FOR J=0 TO 28: READ X$: POKE MC+J, VAL("&H"+X$): NEXT
590 '
600 ' POKE the address dependant bytes into the program
610 ADDR=MC+29: POKE MC+1, CLKIO
620 POKE MC+5, ADDR-INT(ADDR/256)*256: POKE MC+6, INT(ADDR/256)
630 POKE MC+15, ADDR-INT(ADDR/256)*256: POKE MC+16, INT(ADDR/256)
640 '
650 ' *****
660 ' ** START OF MAIN ROUTINE *****
670 ' *****
680 '
690 ' Clear any clock test and interrupt mode
700 OUT CLKIO+0, 0: OUT CLKIO+15, 0
710 '
720 ' Select time set and/or display
730 GOSUB 1280
740 INPUT "Set new time (Y/N) "; I$
750 IF I$ = "Y" OR I$ = "y" THEN 810 ELSE 1020
760 '
770 ' *****
780 ' ** SET TIME AND DATE *****
790 ' *****
800 '
810 GOSUB 1280
820 PRINT "Note: colons and leading zeros significant"
830 PRINT
840 INPUT "Enter time (hh:mm) "; I1$
850 INPUT "Enter day-of-week, date, month (d:dd:mm) "; I2$
860 '
870 ' Convert inputs into a binary output string
880 I$=CHR$(VAL(MID$(I1$,5,1))) + CHR$(VAL(MID$(I1$,4,1)))
890 I$=I$+CHR$(VAL(MID$(I1$,2,1))) + CHR$(VAL(MID$(I1$,1,1)))
900 I$=I$+CHR$(VAL(MID$(I2$,4,1))) + CHR$(VAL(MID$(I2$,3,1)))
910 I$=I$+CHR$(VAL(MID$(I2$,1,1))) + CHR$(VAL(MID$(I2$,7,1)))
920 I$=I$+CHR$(VAL(MID$(I2$,6,1))) + CHR$(YR) + CHR$(15)
930 '
940 ' Stop the clock, output the string and restart the clock
950 OUT CLKIO+14, 0
960 FOR J=1 TO LEN(I$): OUT CLKIO+J+3, ASC(MID$(I$,J,1)): NEXT
970 '
980 ' *****
990 ' ** READ TIME AND DATE *****
1000 ' *****
1010 '
1020 GOSUB 1280
1030 DAYS="SunMonTueWedThuFriSat"
1040 MTH$="JanFebMarAprMayJunJulAugSepOctNovDec"
1050 CALL MC: ' Read the clock
1060 '
1070 ' Get the data from the workspace and build an output string
1080 T$="The time is ---> "

```

```

1090 I=PEEK(ADDR+6)*10 + PEEK(ADDR+5): GOSUB 1330: T$=T$ + I$ + " "
1100 I=PEEK(ADDR+4)*10 + PEEK(ADDR+3): GOSUB 1330: T$=T$ + I$ + " "
1110 I=PEEK(ADDR+2)*10 + PEEK(ADDR+1): GOSUB 1330: T$=T$ + I$ + " "
1120 T$=T$ + RIGHT$(STR$(PEEK(ADDR)),1) + " "
1130 T$=T$ + MID$(DAY$,3*(PEEK(ADDR+9))-2,3) + " "
1140 I=PEEK(ADDR+8)*10 + PEEK(ADDR+7): GOSUB 1330: T$=T$ + I$ + " "
1150 I=PEEK(ADDR+11)*10 + PEEK(ADDR+10)
1160 T$=T$ + MID$(MTH$,3*I-2,3) + " " + YEAR$
1170 '
1180 ' Print the string and leave the system
1190 PRINT T$
1200 PRINT: PRINT: PRINT: PRINT: PRINT: PRINT: PRINT: PRINT
1210 SYSTEM
1220 '
1230 ' *****
1240 ' ** SUBROUTINES *****
1250 ' *****
1260 '
1270 ' Clear the screen and move half way down the screen
1280 PRINT CLS; TAB(20); "GM816 CLOCK DISPLAY/TIME SET ROUTINE"
1290 FOR J=1 TO 11: PRINT: NEXT
1300 RETURN
1310 '
1320 ' Convert a digit into string and add leading 0 if required
1330 IF I<=9 THEN I$="0"+RIGHT$(STR$(I),1): RETURN
1340 I$=RIGHT$(STR$(I),2): RETURN
1350 END

```

```

"Set" MACRO-80 3.4 PAGE 1
title "Set"
.z80
aseg
org 100h
true equ 1
false equ 0
.comment \
*****
* This routine sets the clock time to
* hh:mm where hh is the number of hours
* * and mm the number of minutes.
*
* For CP/M or RP/M - program starts at 100h.
* Assemble it into a file SET.COM and
* execute as SET hh:mm. There should be
* * only one space between SET and hh:mm.
*
* For Nas-Sys - The program starts at C80h.
* * Execute C80h and the cursor will appear
* * on a blank line type in hh:mm. There
* * should be no leading spaces before hh:mm.
* *****
\
;Define the operating system
cpm equ true
ife cpm
.phase 0c80h
endif
endif
if cpm
bdos equ 5 ;CP/M BDOS entry point
args equ 0082h ;command tail arguments
else
scal equ 18h ;Nas-Sys subroutine call
mret equ 5bh ;Nas-Sys return routine
inlin equ 63h ;Nas-Sys line input
endif
clock equ 20h ;base address of clock

```

```

"Set" MACRO-80 3.4 PAGE 1-1
;Initialisation
init: xor a ;zero it
out (clock+14),a ;stop the clock
out (clock+0),a ;set non-test mode
out (clock+15),a ;no interrupts
if cpm
ld hl,args ;pointer to arguments
else
rst scal ;get hours & mins
defb inlin
ex de,hl ;pointer to args in HL
endif
;Load the clock registers
ld c,clock+7 ;pointer to registers
call load ;load hours
call load
inc hl ;load mins
call load
call load
ld a,1
out (clock+14),a ;start clock
if cpm
ret ;and return to CCP
else
rst scal ;return
defb mret ; to Nas-Sys
endif
;Subroutine to load ASCII data at (HL) in register C
load: ld a,(hl) ;get ASCII
sub 30h ;into binary
out (c),a ;load the reg
dec c ;next register
inc hl ;next ASCII value
ret
end
PAGE S

```

```

"Set" MACRO-80 3.4
Macros:
Symbols:
ARGS 0082 BDOS 0005 CLOCK 0020 CPM 0001
FALSE 0000 INIT 0100 LOAD 011E TRUE 0001
No Fatal error(s)

```


"Time" MACRO-80 3.4

PAGE 1

title "Time"

0000

.280
aseg

org 100h

0001
0000true equ 1
false equ 0

.comment \

```

*****
* This program prints the clock time as
* hh:mm:ss.
*
* For CP/M or RP/M - program starts at 100h.
* Assemble it into a file TIME.COM and
* execute TIME to print up the time.
*
* For Nas-Sys - The program starts at C80h.
* Assemble it and execute C80h to print
* up the time.
*****

```

0001

;Define the operating system
cpm equ true

ife cpm

.phase 0c80h

endif

;Equates

0005

```

if cpm
  bdos equ 5 ;CP/M BDOS entry point
else
  rout equ 30h ;Nas-Sys output routine
  scal equ 18h ;Nas-Sys subroutine call
  mret equ 5bh ;Nas-Sys monitor return
  crlf equ 6ah ;Nas-Sys carriage return
endif

```

0006
0020nmrreg equ 6 ;number of registers to re;
clock equ 20h ;clock chip base address

0100 03 0109

jp start

"Time" MACRO-80 3.4

PAGE 1-1

0103

regs: defs 6 ;temp storage for registers

;Read in the clock registers

0109 0E 21
010B 06 06
010D 21 0103
0110 0C
0111 ED A2
0113 20 FB

```

start: ld c,clock+1 ;first register
      ld b,nmrreg ;no to read
      ld hl,regs ;start of storage
      inc c ;point to next port
      ini ;put value into (HL)
      jr nz, read ;do the rest

```

;Scan the values for any update indicators

0115 06 06
0117 21 0103
011A 7E
011B E6 0F
011D FE 0F
011F 28 E8
0121 77
0122 23
0123 10 F5

```

      ld b,nmrreg ;no to scan
      ld hl,regs ;start of storage
      scan: ld a,(hl) ;get value
            and 0fh ;mask top nibble
            cp 0fh ;update indicator?
            jr z,start ;yes, try again
            ;resave it
            ld (hl),a
            inc hl ;point to next one
            djnz scan ;and repeat

```

;Output the time to the console

0125 21 0108
0128 CD 0138
012B CD 014D
012E CD 0138
0131 CD 014D
0134 CD 0138

```

      ld hl,regs+nmrreg-1 ;end of storage
      call outnum ;hours
      call colon ;delimiter
      call outnum ;mins
      call colon ;delimiter
      call outnum ;secs

```

if cpm

ret ;and exit to CCP

else

rst scal ;output CR

defb crlf

rst scal ;return

defb mret ; to Nas-Sys

endif

0005

```

if cpm
  bdos equ 5 ;CP/M BDOS entry point
else
  rout equ 30h ;Nas-Sys output routine
  scal equ 18h ;Nas-Sys subroutine call
  mret equ 5bh ;Nas-Sys monitor return
  crlf equ 6ah ;Nas-Sys carriage return
endif

```

0006
0020nmrreg equ 6 ;number of registers to re;
clock equ 20h ;clock chip base address

0100 03 0109

jp start

```

;Subroutine to output register values
outnum: call outasc ;output hi byte
      dec hl ;set lo byte
      call outasc ;output it
      dec hl
      ret

```

;Subroutine to output A as a digit

```

outasc: ld a,(hl) ;get value
      add a,30h ;convert to ASCII

```

```

"Time" MACRO-80 3.4 PAGE 1-2
if cpm
    0144 5F ld e,a
    0145 0E 02 ld c,2
    0147 E5 push hl
    0143 C9 0005 call blos
    014B E1 pop hl
else
    rst cout
endif
;console output
;throw it out
;output it

```

```

014C C9 rst
;Subroutine to output a colon
if cpm
    colon: ll e,2
    014D 1E 3A ;ASCII colon
    014F 0E 02 ld c,2
    0151 E5 push hl
    0152 C9 0005 call blos
    0155 E1 pop hl
    0156 C9 rst
else
    colon: ll a,2
    rst cout
endif
;ASCII colon

```

```

"Time" MACRO-80 3.4 PAGE 3
end
end
Macros:
Symbols:
    0005 CLOCK 0020 COLON 014D CPM 0001
    0000 NMRREQ 0006 OUTASC 0141 OUTNUM 0133
    0110 REGS 0103 SCAN 011A START 0109
    0001 TRUE

```

No Data error(a)

THE COPYING OF THIS DOCUMENT IS
FORBIDDEN FOR ANY REASON WHATSOEVER
WITHOUT WRITTEN CONSENT FROM GEMINI
MICROCOMPUTERS LTD. 1984

© COPYRIGHT GEMINI MICROCOMPUTERS LTD. 1984



Gemini Microcomputers Ltd

18 Woodside Road Amersham Bucks HP7 0BH England.