Heurikon HK68/M120

User's Manual

HEURIKON CORP.

Microcomputers For Industry

Heurikon HK68/M120 - User´s Manual
Heurikon Corporation
Madison, WI

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The purpose of this manual is to document the features of the Heurikon HK68/M120 (tm) microcomputer board.

This manual covers the unique features of the HK68/M120 board. Although general information, such as MPU, MMU, SCSI, CIO, and DUSCC programming is discussed, more detailed information is available directly from the chip manufacturers.

Feel free to contact Heurikon Corporation (Customer Support Department) if questions arise. We are prepared to answer general questions as well as help with specific applications.

1.1 Disclaimer

The information in this manual has been checked and is believed to be accurate and reliable. HOWEVER, NO RESPONSIBILITY IS ASSUMED BY HEURIKON FOR ITS USE OR FOR ANY INACCURACIES. Specifications are subject to change without notice. HEURIKON DOES NOT ASSUME ANY LIABILITY ARISING OUT OF USE OR OTHER APPLICATION OF ANY PRODUCT, CIRCUIT OR PROGRAM DESCRIBED HEREIN. This document does not convey any license under Heurikon´s patents or the rights of others.

[1] HK68, HK68/M120 and Hbug-M120 are trademarks of Heurikon Corporation.

[2] This document was prepared using the UNIX nroff facility and the PWB/mm macros.

[3] UNIX is a trademark of AT&T Bell Laboratories.

2. HK68/M120 FEATURE SUMMARY

MPU — Motorola 68020 microprocessor chip; 12.5 Mhz; 16.67, 20 and 25 Mhz option; 32-bit internal architecture, 32-bit address and data paths; 32 address lines; 4 gigabyte addressing range; 256-byte Instruction Cache. (Ref: section 5)

FPP — 68881 Floating Point Co-processor. Implements the IEEE-P754 Binary Floating Point Standard. (Ref: section 7)

PMMU — Motorola 68851 chip (or equiv). Provides logical to physical address translation. Demand Paged Virtual Memory operation. (Ref: section 8) (Not present on the HK68/M12F.)

RAM — 256K to 4 megabyte capacity; One parity bit per byte; Uses 64K, 256K or 1024K x 9 DRAMs. Hardware refresh. (Ref: section 10)

EPROM — Four ROM sockets; 256 Kbyte total capacity. 8 or 32 bit data path. (Ref: section 10)

Multibus — 24-bit addressing (16 megabyte range); 16-bit data bus, compatible with 8-bit boards; Eight bus interrupts, bi-directional (via CIO); Master/Slave modes; On-card byte swap buffer. (Ref: section 11)

iLBX bus — High speed local memory expansion up to 16 Mbytes; Supports a secondary bus master. (Ref: section 11)

Serial I/O — Two serial I/O ports (Signetics 68562 Serial Communication Controller - DUSCC); Separate baud rate generators for each port; Asynchronous and synchronous modes; RS-232C interface, RS-422 option. (Ref: section 14)

SCSI — ANSI X3T9.2 compatible SCSI controller; supports up to 8 disk drive controllers or other devices; (Ref: section 15)

Centronics — Control I/F and eight-bit output port for Centronics-type printer. (Ref: section 16)

LEDs — Four user LEDs under software control; Four MPU/BUS status LEDs (Ref: section 12)

User Jumpers — Eight user definable jumpers. (Ref: section 12)

SBX — One 16-bit SBX connector; provides I/O or feature expansion. (Ref: section 17)

CIO — Zilog Z8536 Counter/Timer and Parallel I/O Unit; three 16-bit counter/timers; three parallel ports for on-card control functions. (Ref: section 13)

NV-RAM
Nonvolatile Static RAM; 256 x 4 configuration; Internal EEpROM; 100 year retention; 10,000 store cycle lifetime; For user definable functions. (Ref: section 10.7)

Mailbox
Allows remote interrupt of the HK68/M120 via specified Multibus addresses. (Ref: section 11.8)

RTC
Optional Real-Time Clock module for time-of-day maintenance. With battery backup. (Ref: section 18)

3. BLOCK DIAGRAM

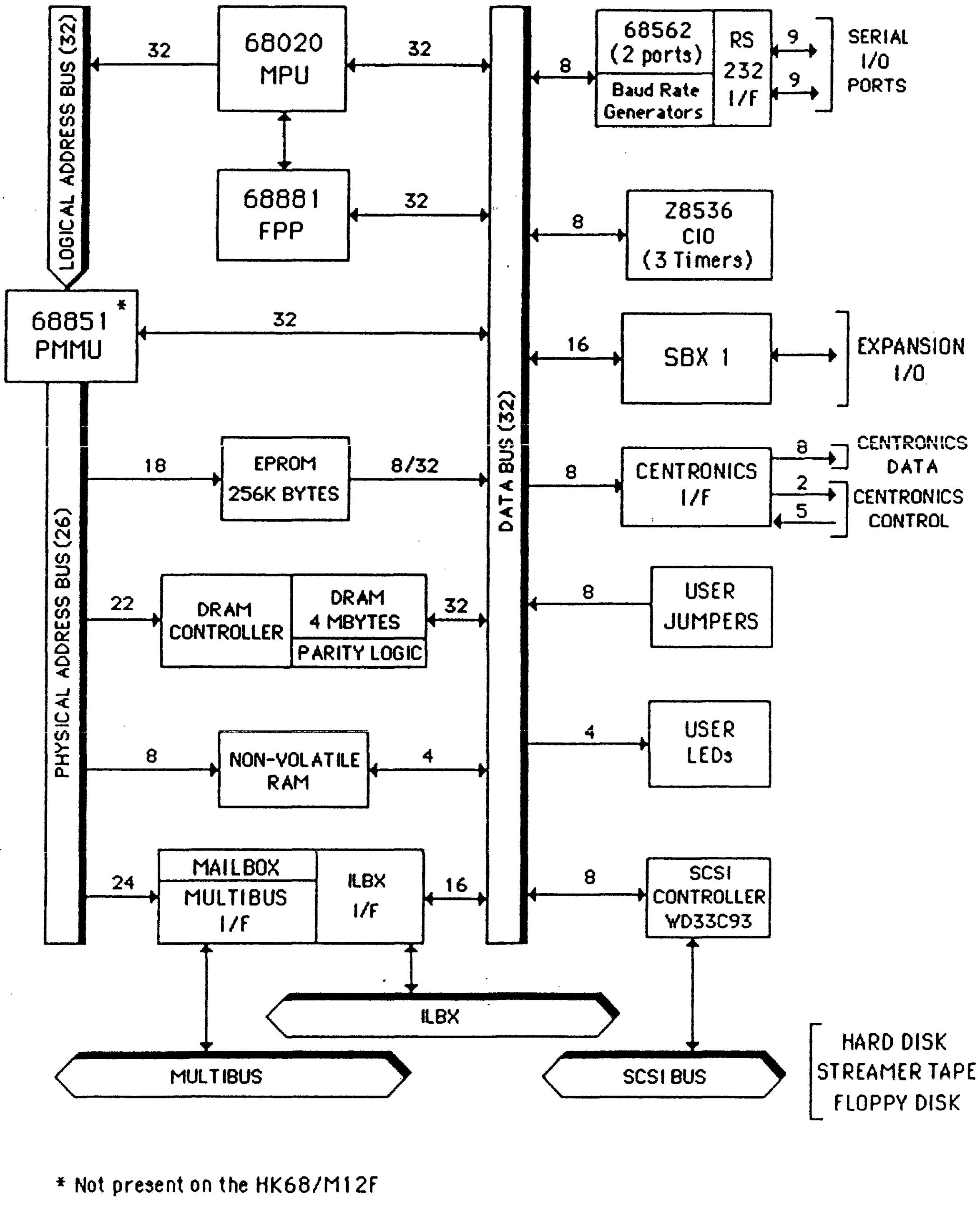


Figure 1. HK68/M120 Block Diagram

4. GETTING GOING

Here is what you need to get the Heurikon HK68/M120 "on-the-air":

- Heurikon HK68/M120 Microcomputer board
- Card cage and power supply
- Serial I/F cable (RS-232)
- CRT Terminal
- Heurikon Hbug-M120 monitor and bootstrap EPROM

4.1 Installation Steps

- CAUTION: All semiconductors should be handled with care. Static discharges can easily damage the components on the HK68/M120. Keep the board in an anti-static bag whenever it is out of the system chassis and do not handle the board unless absolutely necessary. Ground your body before touching the HK68/M120 board.

- CAUTION: High operating temperatures will cause unpredictable operation. Because of the high chip density, fan cooling is required for most configurations, even when cards are placed on extenders.

All products are fully tested before they are shipped from the factory. When you receive your HK68/M120, follow these steps to assure yourself that the system is operational:

[1] Visually inspect the board(s) for loose components which could be the result of shipping vibrations. Visually inspect the chassis and all cables. Be sure all boards are seated properly in the card cage. Be sure all cables are securely in place.

[2] Connect a CRT terminal to Serial Port B, via connector P6. If you are making your own cables, refer to section 14. Set the terminal as follows:

  - 9600 baud, full duplex.
  - Eight data bits (no parity).
  - Two stop bits for transmit data.
  - One stop bit for receive data.
  - If your terminal does not have separate controls for transmit and receive stop bits, select one stop bit for both transmit and receive.

[3] Connect AC power and turn the system on.

[4] Push the system RESET button. A sign-on message and prompt from the monitor should appear on the screen. If not, check your power supply voltages and CRT cabling.

[5] Now is the time to read the monitor manual and the operating system literature. Short course: To boot the operating system, insert a diskette and enter ´bf´ (for boot floppy) or ´bw´ (to boot from Winchester.)

[6] Reconfigure the jumpers, etc, as necessary for your application. See Section 20 for a summary of I/O device addresses and configuration jumpers.

4.2 Troubleshooting and Service Information

In case of difficulty, use this checklist:

[1] Be sure the system is not overheating.

[2] Inspect the power cables and connectors. If the HK68/M120 board has power, one of the four status LEDs (near P4) should be on.

[3] If the Hbug-M120 monitor program is executing, run the diagnostics, via command ´uc´ or ´um´.

[4] Check your power supply for proper DC voltages. If possible, look for excessive power supply ripple or noise using an oscilloscope.

[5] Check the chips to be sure they are firmly in place. Look for chips with bent or broken pins. In particular, check the EPROMs.

[6] Check your terminal switches and cables. Be sure the P6 connector is on properly. The cable stripe (wire #1) should be toward the center of the HK68/M120 board and the cable should flow toward the rear. The port B portion of the cable is on the pin #34 side. If you have made your own cables, pay particular attention to the cable drawings in section 14.

[7] Check the jumpers to be sure your board is configured properly. All jumpers should be in the "standard configuration" positions shown in section 20.3.

[8] After you have checked all of the above items, call us at (608)-271-8700 and ask our Customer Service Department for help. Please have the following information handy:

- The state of the Status LEDs (near P4).

- The monitor program revision level (part of sign-on message).
- The HK68/M120 p.c.b. serial number (scribed along card edge).
- The complete HK68/M120 model number, including option codes.
- The serial number of the Operating System.

If you plan to return the board to Heurikon for service, contact our Customer Service Department to obtain a Return Merchandise Authorization (RMA) number. Be prepared to provide the items listed above, plus your Purchase Order number and billing information if your HK68/M120 is out of warranty. If you return the board, be sure to enclose it in the anti-static bag, such as the one in which it was originally shipped. Send it prepaid to:

Heurikon Corporation
Factory Service Department
3201 Latham Drive
Madison, WI 53713

Please put the RMA number on the package so we can handle your problem most efficiently.

4.3 Monitor Summary

The HK68/M120 monitor and bootstrap program, Hbug-M120, is contained in one EPROM. It is intended to provide a fundamental ability to check the memory and I/O devices, to manually enter a program and to down-line load or bootstrap a larger program into memory. Advanced features and utilities may be loaded from media or via an operating system.

Refer to the Hbug-M120 manual for details on the commands and command formats.

5. MPU SUMMARY INFORMATION

This section details some of the important features of the 68020 MPU chip and, in particular, those items which are specific to the implementation on the Heurikon HK68/M120.

5.1 MPU Interrupts

The MPU can internally set an interrupt priority level in such a way that interrupts of a lower priority will not be honored. Interrupt level seven, however, cannot be masked off.

Level	Interrupt
7	Parity error, highest priority, non-maskable
6	CIO Interrupt (sub-priority: timer 3, port A, timer 2, port B, timer 1)
5	DUSCC Interrupt
4	SCSI Interrupt (autovectored)
3	Not assigned
2	Not assigned
1	Not assigned
0	Idle, no interrupt

Table 1. MPU Interrupt Levels

When an interrupt is recognized by the MPU, the current instruction is completed and an interrupt acknowledge sequence is initiated, whose purpose is to acquire an interrupt vector from the interrupting device. The vector number is used to select one of 256 exception vectors located in reserved locations in lower memory (see section 5.2 for a listing.) The exception vector specifies the address of the interrupt service routine.

The DUSCC and CIO devices on the HK68/M120 are capable of generating more than one vector, depending on the particular condition which caused the interrupt. This significantly reduces the time required to service the interrupt because the program does not have to rigorously test for the interrupt cause. Section 11.5 has more information on the HK68/M120 interrupt logic.

5.2 MPU Exception Vectors

Exception vectors are memory locations from which the MPU fetches the address of a routine to handle an exception (interrupt). All exception vectors are two words long (four bytes), except for the reset vector which is four words. The listing below shows the vector space as it appears to the Heurikon HK68/M120 MPU. It varies slightly from the Motorola MPU manual listing due to particular implementations on the HK68/M120 board. Refer to the MPU documentation for more details. The vector table normally occupies the first 1024 bytes of memory, but may be moved to other locations under software control. Unused vector positions may be used for other purposes (e.g., code or data) or point to an error routine.

Vector	Offset	Assignment
0	000	Reset: Initial SSP (Supervisor Stack Pointer)
1	004	Reset: Initial PC (Supr Program Counter)
2	008	Bus Error (Watchdog Timer, MMU Fault)
3	00C	Address Error
4	010	Illegal Instruction
5	014	Divide by Zero
6	018	CHK Instruction (register bounds)
7	01C	TRAPV Instruction (overflow)
8	020	Privilege Violation (STOP, RESET, RTE, etc)
9	024	Trace (Program development tool)
10	028	Instruction Group 1010 Emulator
11	02C	FPP or MMU Coprocessor not present
12	030	(reserved)
13	034	FPP or MMU Coprocessor Protocol Violation
14	038	Format Error
15	03C	Uninitialized Interrupt
16-23	040-05F	(reserved-8)
24	060	Spurious Interrupt, not used
25	064	Level 1 autovector, not used
26	068	Level 2 autovector, not used
27	06C	Level 3 autovector, not used
28	070	Level 4 autovector, SCSI Interrupt
29	074	Level 5 autovector, not used
30	078	Level 6 autovector, not used
31	07C	Level 7 autovector, parity error
32-47	080-0BF	TRAP Instruction Vectors (16)
48-54	0C0-0DB	FPP Exceptions (8)
55-63	0DC-0FF	(reserved-8)
64-255	100-3FF	User Interrupt Vectors (192)

Table 2. MPU Exception Vectors

Autovectoring is used for parity error and SCSI. Interrupts from all other devices can be programmed to provide a vector number (which would likely point into the "User Interrupt Vector" area, above).

The table on the following page gives suggested interrupt vectors for each of the possible device interrupts which could occur. Note that the listing is in order of interrupt priority, highest priority first (except for DUSCC internal priorities, which are adjustable).

Level	Vector	Device	Condition
7	31	Memory	Parity error autovectored interrupt
6	96	CIO	Timer 3
	79	CIO	External Interrupt (P5-11)
	77		SBX module interrupt INT 0
	75		SCSI Data Request
	73		Centronics Acknowledge
	71		Mailbox Interrupt
	69		SBX module interrupt INT 1
	67		SBX Data Request
	65		(SBX module present)
	98	CIO	Timer 2
	78	CIO	INT0 Multibus Interrupt 0
	76		INT1 Multibus interrupt 1
	74		INT2 Multibus interrupt 2
	72		INT3 Multibus interrupt 3
	70		INT4 Multibus interrupt 4
	68		INT5 Multibus interrupt 5
	66		INT6 Multibus interrupt 6
	64		INT7 Multibus interrupt 7
	100	CIO	Timer 1
	102	CIO	Timer, error
5	80	DUSCC	Port A, Receive character available
	81		Port A, Transmit buffer empty
	82		Port A, Rcv/Tx Status
	83		Port A, External/Status change
	84		Port B, Receive character available
	85		Port B, Rcv/Tx Status
	86		Port B, Transmit buffer empty
	87		Port B, External/Status change
4	28	SCSI	SCSI autovectored interrupt

Table 3. Suggested Interrupt Vectors

The suggested interrupt vectors for the CIO and DUSCC devices take into account that some of the bits of the vectors are shared, e.g., all CIO Port A vectors have five bits which are the same for all interrupt causes.

Each on-card device contains interrupt enable and control bits which allow the actual interrupt priority levels to be modified under program control by temporarily disabling certain devices. The internal DUSCC priorities are programmable .

Of course, fewer vectors may be used if the devices are programmed not to use modified vectors or if interrupts from some devices are not enabled.

If you want to use the suggested vector numbers in the above table, the proper values to load into the device vector registers are:

Device	Hex Value	Decimal Value
DUSCC (Ports A & B):	0x50	80
CIO, Port A:	0x41	65
CIO, Port B:	0x40	64
CIO, C/T vector:	0x60	96

Table 4. Device Interrupt Vector Values (Suggested)

Making your way through the Zilog CIO manual in search of details on the interrupt logic is quite an experience. We suggest you start with these recommended readings from the CIO technical manual:

Device	Item	
CIO	Z8536	Technical Manual Vector register: section 2.10.1 Bit priorities: section 3.3.2

5.3 Status LEDs

There are four status LEDs which give a visual indication of the MPU and bus status. These LEDs continuously show the state of the board as follows:

P5 Pin	LED	Name	Meaning
P5- 2	S	Supr	The MPU is in the supervisor state.
P5- 4	U	User	The MPU is in the user state.
P5- 6		n/c	
P5-10	B	Bus	Another Multibus master has control of the local bus.
P5- 8	H	Halt	The MPU has halted. (Double bus fault, odd stack address or the system reset line is active).
P5-odd		Vcc	

Table 5. Status LEDs (P5)

The output signals are low when true. Each is suitable for connection to a LED cathode. An external resistor must be provided for each output to limit current to 15 milliamps.

Two input signals are also provided on P5 for interrupt and reset.

P5 pin	Name	Function
P5-11	INTR*	Connected to CIO bit A7, and pull-up (Refer to section 13.1)
P5-12	Gnd	
P5-13	RESET*	When low, causes a local reset (Same as on-card RESET button)
P5-14	Gnd	

Table 6. Control Panel Interface (P5)

A recommended mating connector for P5 is Molex P/N 15-29-8148.

5.4 Co-Processors

The HK68/M120 supports the PMMU and FPP coprocessors. Both are described in more detail in the following sections.

Co-Proc ID	Device	Function	Reference Section
0	68851	Paged Memory Management Unit (PMMU)	8
1	68881	Floating Point Coprocessor (FPP)	7

Table 7. 68020 Coprocessor ID Codes

6. DMAC SUPPORT

DMA support on the HK68/M120 is provided by the MPU in conjunction with special on-card synchronizing logic. The high speed interfaces (SCSI and SBX) have a data ready signal which can be polled or can generate an interrupt (under software control). In addition, the MPU wait signal (DTACK) can be controlled by the data ready signals so the MPU does not need to poll between transfer cycles. These capabilities are referred to as a "pseudo-DMA".

Because of the 68020 internal instruction cache and the HK68/M120 high-speed system bus, the MPU-based pseudo-DMA performs as well as conventional DMA logic for most transfer operations.

The following DMAC facilities are available:

		--------Polled Capability--------	
Device	Interrupt Capability	Auto Wait	No Wait
SBX	MPU interrupt 6, via CIO	Special Map	Register Poll
DUSCC	MPU interrupt 5	-	DUSCC Register Poll
SCSI	MPU interrupt 4	Special Map	SCSI Register Poll

Table 8. Pseudo-DMAC Capabilities

Refer to the particular device section of this manual for more information on the pseudo-DMA interface for each device.

6.1 DMA Software Implementation Example

There are three parts to this example. The first two parts are the ´C´ potion of the pseudo-DMA; the third section is the machine code portion. Not all of the SCSI initialization logic is shown, since that is application dependent. This example is extracted from our SCSI UNIX device driver.

```
#include <sys/wd33C92.h> /* describes wd structure, scsidev */
#include <sys/cio.h>    /* describes CI */
#include <setjmp.h>

#define CIO     ((struct cdevice *)0x00fe9000)
#define CIO_APM         0x27    /* Port A pattern Mask Register */
#define CIO_MCC_PAE     0x04    /* Port A enable */

unsigned char rwflag;           /* read or write flag, for pseudo dma */
unsigned char *addr;            /* DMA data address */
unsigned char *a0save;          /* copy of a0 at bus error time */
int      *SDMA_jb;

wd_cmd(adrs)
char *adrs;
{
        ...                     /* send params to WD chip */
        wd->asr = 0x01;         /* control register
        wd->port = 0x80 | 0x08; /* dma data transfer, save successful */
                                /* ...completion intr until target disc */
        ctrl_op(CIO_APM, 0x20, OR);  /* turn on DRQ int */

        rwflag = rw;            /* save transfer direction */
        a0save = ++adrs;        /* DMA transfer adrs, do postincr as if */
                                /* ... we had a buserr */
        wd->asr = 0x18;         /* command register */
        wd->port = 0x09;        /* Select without ATN, start transfer */
        return;                 /* will get MPU intr 4 when SCSI done */
}

/*
 * We get here via interrupt vector 75, CIO bit A5, SCSI Data Request.
 * That interrupt will be the first DRQ; hardware synchronizes the rest.
 * A machine code program (not shown) saves and restores the registers
 */
scsi_drq()
{
        register struct scsidev *wd = ((struct scsidev *)0x00fe8000);
        jmp_buf jb;

        ctrl_op(CIO_APM, ~0x20, AND); /* turn off DRQ int via cio mask */
```

Figure 2. DMA Software Implementation Example (Part 1)

```
        CIO->ctrl = 0x08;        /* clear CIO int pending */
        CIO->ctrl = 0x20;

        SDMA_jb = jb;            /* save for longjmp in buserr */
        if (!setjmp(jb)) {       /* save state for later longjmp */
                /* we come here on the normal setjmp() return to do DMA */
                if (rwflag == Read) {        /* here comes the DMA... */
                        SDMA_in(a0save-1);   /* does not return */
                } else {
                        SDMA_out(a0save-1);  /* also does not return */
                }
        } else {
        /* we end up here after the buserr longjmp, at end of transfer */
                ctrl_op(CIO_APM, 0x20, OR);  /* re-enable DRQ interrupt */
        }
        /* done with DRQ intr, expecting level 4 autovector from SCSI */
}


/*
 * This routine allows a pattern mask register bit to be changed
 * without generating a spurious interrupt
 */
ctrl_op(reg , bits , op)        /* op = OR(==0) or op = AND(!=0)
register unsigned bits;
{
        short        s;
        register unsigned char *CIOCTRL = (unsigned char *)&CIO->ctrl;
        register unsigned char temp;

        s = spl7();                        /* turn MPU interrupts off */

        /* disable port A... */
        *CIOCTRL = CIO_MCC; temp = *CIOCTRL;
        *CIOCTRL = CIO_MCC; *CIOCTRL = temp & ~(CIO_MCC_PAE);

        /* fiddle with specifed CIO register... */
        *CIOCTRL = reg;
        temp = (op == OR) ? (*CIOCTRL | bits) : (*CIOCTRL & bits);
        *CIOCTRL = reg; *CIOCTRL = temp;      /* put the new value back */

        /* re-enable port A... */
        *CIOCTRL = CIO_MCC; temp = *CIOCTRL;
        *CIOCTRL = CIO_MCC; *CIOCTRL = temp | CIO_MCC_PAE;

        splx(s);        /* restore previous MPU intr level */
}
```

Figure 3. DMA Software Implementation Example (Part 2)

```
# This funny looking code matches UniSoft UNIX System V.2 assembler input.
        global  wdrxfer, wdwxfer, a0save, SDMA_jb, longjmp
buserr: # From vector 2.  Not shown: stack cleanup and register saving
        cmp.l   (0x14.w,%sp),&0x00fe7000 # test access (fault) address
        bne     otherfault   # fault not related to pseudo DMA
                             # Else, we were waiting for SCSI DRQ
                             # and got a SCSI Intr or watchdog.
        mov.l   %a0,a0save   # data adrs (plus one) where transfer stopped
        # The MPU did the postincr even though the mov.b failed
        mov.l   &1,-(%sp)
        mov.l   SDMA_jb,-(%sp)
        bsr     longjmp      # same as: longjmp(SDMA_jb,1);
        # we don´t return from a longjmp

# Machine code is used to implement the pseudo DMA to assure that we know
# which MPU register is used for the data adrs.  Buserr needs to know.
SDMA_out:mov.l  &0x00fe7000,%a1     # Transfer from mem to SCSI
        mov.l   (4.w,%sp),%a0       # get memory adrs

wloop%: mov.b   (%a0)+,(%a1)        # Multiple moves are used
        mov.b   (%a0)+,(%a1)        # to increase efficiency.
        mov.b   (%a0)+,(%a1)        # After the first pass, all
        mov.b   (%a0)+,(%a1)        # instructions are read from
        mov.b   (%a0)+,(%a1)        # the 68020 internal cache.
        mov.b   (%a0)+,(%a1)        # Thus, the bra.b doesn´t
        mov.b   (%a0)+,(%a1)        # consume bus time.
        mov.b   (%a0)+,(%a1)
        bra.b   wloop%

SDMA_in:mov.l   &0x00fe7000,%a1     # Transfer from SCSI to mem
        mov.l   (4.w,%sp),%a0       # get memory adrs

rloop%: mov.b   (%a1),(%a0)+        # We will leave these loops
        mov.b   (%a1),(%a0)+        # only on buserr, which will
        mov.b   (%a1),(%a0)+        # occur when the SCSI Interrupt
        mov.b   (%a1),(%a0)+        # line comes on (indicating end
        mov.b   (%a1),(%a0)+        # of transfer or error) or if
        mov.b   (%a1),(%a0)+        # the watchdog timer expires
        mov.b   (%a1),(%a0)+        # (indicating a transfer pause,
        mov.b   (%a1),(%a0)+        # due to an inter-record gap
        bra.b   rloop%              # or invalid address).
```

Figure 4. DMA Software Implementation Example (Part 3)

7. FLOATING POINT CO-PROCESSOR (FPP)

The HK68/M120 uses the MC68881 floating point processor chip. It runs as a coprocessor with the 68020. Heurikon can provide software support for the 68881, please contact the factory.

7.1 FPP Feature Summary

- Allows fully concurrent instruction execution with the main processor.

- Eight general-purpose floating-point data registers, each supporting a full 80-bit extended-precision real data format (a 64-bit mantissa plus a sign bit, and a 15-bit biased exponent).

- A 67-bit ALU to allow very fast calculations, with intermediate precision greater than the extended-precision format.

- A 67-bit barrel shifter for high-speed shifting operations (for normalizing, etc.)

- 46 instruction types, including 35 arithmetic operations.

- Fully conforms to the IEEE P754 standard, including all requirements and suggestions. Also Supports functions not defined by the IEEE standard, including a full set of trigonometric and logarithmic functions.

- Supports seven data types: byte, word, and long integers; single, double, and extended-precision real numbers; and packed binary coded decimal string real numbers.

- Efficient mechanisms for procedure calls, context switches, and interrupt handling.

FPP programming details are available in the 68881 technical manual.

7.2 FPP Bypass

The HK68/M120 will operate without the FPP chip. Simply unplug the FPP if it is not required. No wires or jumpers are needed.

If the Watchdog Timer is enabled (via the Memory Control Word), the software can determine if the FPP chip is installed. An attempt to access a non-existent FPP will result in a Watchdog timeout and a Bus Error, forcing a Line 1111 MPU Exception, vector number 11.

8. MEMORY MANAGEMENT CO-PROCESSOR (PMMU)

(Not present on the HK68/M12F.)

This section explains some of the relevant features of the 68851 PMMU chip. Refer to the PMMU technical manual for more details.

The PMMU operates as a coprocessor with the MPU.

The PMMU automatically enters a "transparent" mode following a system reset. Thus, all logical addresses and physical address will be the same. The PMMU must be programmed and enabled before any address translations will begin.

8.1 Function Code Definitions

The table below shows the MPU and FPP function codes which are generated for each memory reference. They indicate to the PMMU the particular type of reference being made, and are used to index into the PMMU Address Space Table (AST). Ultimately, the function codes determine the logical to physical mapping and the protection levels for the operation (e.g., write protect, user/supervisor space).

Hex	FC3	FC2	FC1	FC0	
0	0	0	0	0	(reserved)
1	0	0	0	1	User DATA
2	0	0	1	0	User PROGRAM
3	0	0	1	1	(reserved)
4	0	1	0	0	(reserved)
5	0	1	0	1	Supervisor, DATA
6	0	1	1	0	Supervisor, PROGRAM
7	0	1	1	1	CPU Space (FPP, PMMU)
8-F	1	x	x	x	Not used

Table 9. Function Code Assignments

8.2 PMMU Address Line Block Diagram

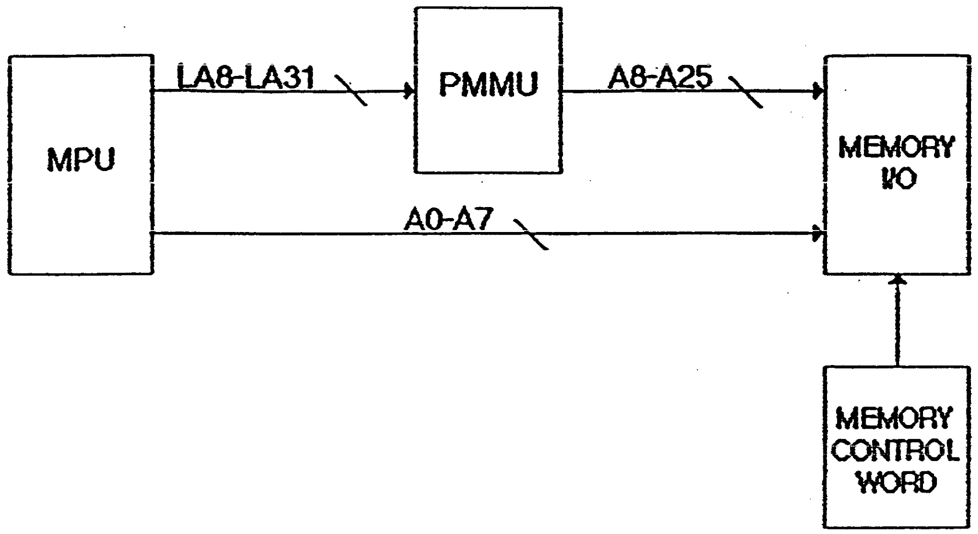


Figure 5. MPU Accesses to Memory or Bus

8.3 PMMU Bypass

The HK68/M120 will operate without an PMMU chip. If the PMMU is removed from the board, the following jumpers must be installed in its place. (A pre-wired bypass header is available from Heurikon.)

Signal	Connect Pins	Signal	Connect Pins
LA31	L3 and C5	LA30	M2 and A3
LA29	M3 and A2	LA28	L4 and B4
LA27	M4 and C4	LA26	N2 and B3
LA25	N3 and B2	LA24	L5 and D4
LA23	M5 and A1	LA22	N4 and C2
LA21	N5 and D3	LA20	L6 and D2
LA19	M6 and B1	LA18	N6 and C1
LA17	N7 and E3	LA16	N8 and E2
LA15	M8 and D1	LA14	L8 and E1
LA13	N9 and F3	LA12	N10 and F2
LA11	M9 and F1	LA10	L9 and G1
LA9	N11 and H1	LA8	N12 and H2
LBRI-O	A7 and A5	LAS	A4 and B5
LBGI-O	A6 and C6	PBR-PBG	B9 and A9

Table 10. PMMU Bypass Connections

If the Watchdog timer is enabled, the software can determine if the PMMU chip is installed. Any attempt to access a non-existent PMMU will result in a Watchdog timeout and thus a Bus Error, forcing a Line 1111 MPU exception, vector number 11.

8.4 Alternate Capabilities - MMB

In some applications, the MMB daughter board may be used instead of the 68851 PMMU. If you are using the MMB, it is addressed as follows:

Address (CPU Space)	MMB Register (read/write)
0008,0000	Root Pointer (RP)
0008,0004	Translation Control Register (TC)

Table 11. MMB Registers

The MMB registers are in the CPU address space. Both registers are 32 bits. Refer to the MMB manual for programming details.

Since the MMB is a piggy-back board, one or two extra card slots are required in a standard rack.

9. SYSTEM ERROR HANDLING

There are numerous events which could cause an error to occur. The responses to these events are carefully controlled.

9.1 Error Conditions

The following error conditions may arise during MPU cycles:

Condition — Meaning

RAM Parity

Incorrect parity was detected during a read cycle from on-card RAM memory. This may be due to a true parity error (RAM data changed,) or because the memory location was not initialized prior to the read and contained garbage.

Parity errors generate a level 7 autovector interrupt. A pointer to the parity error handling routine should be loaded at location 00007C. Parity checking cannot be disabled.

Watchdog Timeout

During an access, usually to the bus, no acknowledge was received within a fixed time interval defined by a hardware timer. (1.67 milliseconds.) This is usually the result of no device being assigned to the specified address. A timeout could also occur if an access from the bus is not terminated by the bus master or if a pseudo-DMA port is accessed and the respective device does not present a data request.

For an access to the bus, the memory cycle is terminated, the $\overline{\text{BERR}}$ (Bus Error) exception is taken by the MPU and execution resumes at the location specified by the exception vector.

If an access from the bus was in progress, no BERR exception occurs.

Double Bus Fault

Another bus error occurred during the processing of a previous bus error, address error or reset exception. This error is the result of a major software bug or a hardware malfunction. A typical software bug which could cause this error would be an improperly initialized stack pointer, which points to an invalid address.

A double bus fault forces the MPU to enter the HALT state. Processing stops. The HALT status LED will come on. The only way out of this condition is to issue a hardware reset.

MMU Fault
The MMU has detected a write violation or an undefined segment address. The memory cycle is terminated and the bus error exception is taken.

Divide by Zero
The value of the divisor for a divide instruction is zero. The instruction is aborted and vector 5 is used to transfer to an error routine.

Privileged Violation
A program executing in the user state attempted to execute a privileged instruction. The instruction is not executed. Exception vector 8 is used to transfer control.

Address Error
An odd address has been specified for an instruction. The bus cycle is aborted and vector 3 is used to transfer control.

Illegal Instruction
The bit pattern for the fetched instruction is not legal or is unimplemented. The instruction is not executed. Exception vector 4, 10 or 11 is used to transfer control.

Format Error
The format of the stack frame is not correct for an RTE instruction. The instruction is aborted and exception vector 14 is used to transfer control.

Line 1111 Emulator
The FPP or PMMU Coprocessor is not present and a coprocessor instruction was fetched. The instruction is not executed. Exception vector 11 will be taken.

FPP Exceptions
The FPP Coprocessor has detected a data processing error, such as an overflow or a divide by zero. The FPP causes the MPU to take one of eight exceptions in the range of 48 to 54.

As the above list indicates, there are two causes for a bus error exception. In order to determine the cause of a bus error exception, test the fault status bits in the MMU. If the MMU indicates no fault then the bus error was caused by the watchdog timer.

10. ON-CARD MEMORY CONFIGURATION

The Heurikon HK68/M120 microcomputer will accommodate a variety of RAM and ROM configurations. There are four ROM sockets for pROM, page addressable ROM or EEpROM, four SIP RAM positions, and a nonvolatile RAM. Off-card memory may be accessed via the Multibus or the iLBX.

10.1 ROM

At power-up or after a system reset, ROM is mirrored throughout the entire MPU address space. The MPU fetches the reset vector from location zero, which specifies the starting program counter and stack address values. RAM will be turned on, and the normal memory map will be activated, on the first access above address 00FA,0000. Thus, the reset vector may point directly to ROM (at base address 00FA,0000).

After RAM has been activated, the ROM base address is 00FA,0000.

Two data path widths, eight and 32 bits, are supported. Jumper J7 determines which width is used. In the eight-bit mode (J7 installed), only one 27512 ROM is required (or allowed); otherwise (J7 removed), four ROMs must be used and they must be of the same type. ROM access time must be 250 nsec or less.

10.1.1 8-bit Data Path

Eight-bit mode is selected by setting J7, J8 and J9 as follows:

Jumper	Position
J7	Installed
J9	J9-C
J8	J8-C

Table 12. ROM Jumpers, 8-bit Data Path

One 27512-type ROM must be in U23. The other ROM sockets, U24, U32 and U33, are not used. The MPU will automatically do multiple accesses to fetch instruction words and data. Logically consecutive bytes, however, do not occupy physically consecutive locations in the ROM. This is because the least significant two physical address lines, A1 and A0, are run to the most significant address inputs on the ROM (A15 and A14); a design dictated by the need to support the 32-bit width mode and still have only a few jumpers. The ROM is divided into four 16K segments. Each segment contains only those bytes associated with a particular value of A1 and A0. For example:

Physical Address	ROM Address
00FA,0000	0x0000
00FA,0001	0x4000
00FA,0002	0x8000
00FA,0003	0xC000
00FA,0004	0x0001
00FA,0005	0x4001
00FA,0006	0x8001
00FA,0007	0xC001
00FA,0008	0x0002
00FA,0009	0x4002
(etc)	(etc)

Table 13. ROM Address Translation, 8-bit Data Path

$$\text{ROM addr} = \frac{\text{PHYSICAL.addr \& 0xFFFC}}{4} + (\ (\text{PHYSICAL.addr \& 3}) * 16384\ )$$

Equation 1. ROM Addresses, 8-bit Data Width Mode

10.1.2 32-bit Data Path

In the 32-bit configuration, all four ROM sockets are used.

D31-D24	D23-D16	D15-D8	D7-D0
U23	U24	U32	U33

Table 14. ROM Chip Positions (32-bit Data Path)

To select the 32-bit width mode, remove J7. J8 and J9 are set according to the ROM type, as follows:

EPROM Type	ROM Capacity	Total Board Capacity	Jumpers
2764	8 Kbytes	32 Kbytes	J9-B J8-B
27128	16 Kbytes	64 Kbytes	J9-B J8-B
27256	32 Kbytes	128 Kbytes	J9-A J8-B
27512	64 Kbytes	256 Kbytes	J9-A J8-A
27513 Paged	64 Kbytes	256 Kbytes	J9-D J8-A
2864 R/W EEpROM	8 Kbytes	32 Kbytes	J9-D J8-open

Table 15. ROM Capacity and Jumper Positions (32-bit Data Path)

Electrically Erasable or paged pROMs may be used. An EEpROM allows specific addresses to be changed by writing to the ROM. When writing to the EEpROM, a delay must be provided by the software between write operations. For the 2864, this delay is 10 milliseconds.

Paged ROMs allow future growth of ROM capacity. A single device can contain multiple 16K byte pages. A specific page is selected by writing the page value to the ROM. For example, to select page three of a 27513, write 0x03 to address 00FA,0000.

10.2 On-Card RAM

EPROM must be turned off following power-up, as described in the previous section. After EPROM has been turned off, on-card memory may be turned on by setting the Memory Control Word. The following table describes the memory control word (located at FE6006, write only). Attention Hbug users: do not use the ´sw´ command; use ´fw´.

Bit	Name	Function (FE6006, write-only)
D15	BMAP3	Bus Map bit 3 (Slave mode, see section 11.4)
D14	BMAP2	Bus Map bit 2 (Slave mode, see section 11.4)
D13	BMAP1	Bus Map bit 1 (Slave mode, see section 11.4)
D12	BMAP0	Bus Map bit 0 (Slave mode, see section 11.4)
D11	LBXE	iLBX enable (See section 10.4)
D10	DOGD	Watchdog Timer Disable (See section 11.7)
D9	MEM1	On-card memory size select bit 1
D8	MEM0	On-card memory size select bit 0
D7	M/L7	Multibus or iLBX bus select (Eighth Mbyte)
D6	M/L6	Multibus or iLBX bus select (Seventh Mbyte)
D5	M/L5	Multibus or iLBX bus select (Sixth Mbyte)
D4	M/L4	Multibus or iLBX bus select (Fifth Mbyte)
D3	M/L3	Multibus or iLBX bus select (Fourth Mbyte)
D2	M/L2	Multibus or iLBX bus select (Third Mbyte)
D1	M/L1	Multibus or iLBX bus select (Second Mbyte)
D0	M/L0	Multibus or iLBX bus select (First Mbyte)

Table 16. Memory Control Word Bit definitions

The control word is set to zero at power-on or by a system reset. It is a write-only register. The bits MEM1 and MEM0 set RAM size according to the table below. The remaining bits will not affect on-card memory, however, they are set any time the memory control word is accessed. Bits M/L7 through M/L0 are described in the Bus Memory section, below.

MEM1	MEM0	Size Selected
0	0	off
0	1	256 Kbytes
1	0	1 Mbyte
1	1	4 Mbytes

Table 17. On-card RAM Size Selection

On-card memory always occupies the block size selected, starting at physical address 000000.

Of course, it is normally desirable to set the memory size according to the amount of memory on-card. See the section on memory sizing below.

The HK68/M120 can accommodate up to four 1 Megabit x 9 SIP RAM packages for a total of four megabytes of on-card memory. Two card slots are required for the four megabyte version. The following RAM configurations are possible:

RAM type	Quantity	Capacity
64K x 9 SIP	4	256 Kbytes
256K x 9 SIP	4	1024 Kbytes
1Meg x 9 SIP	4	4096 Kbytes

Table 18. On-card RAM Capacity

10.3 On-card Memory Sizing

The following algorithm can be used to determine the amount of on-card memory installed. This procedure takes advantage of "mirrors" which exist in higher addresses when the on-card physical memory size is less than the logical memory space. This information is normally used to set the memory size bits (MEM1 and MEM0) after power-up.

[1] Write 0300 (hex) to the Memory Control Word. This will turn on four megabytes of memory.

[2] Clear four megabytes of memory starting at location 000000.

[3] Restore MPU exception vector 31 (points to the parity handler).

[4] Write 5555 (hex) to location 000000.

[5] Read a word from 040000. If the value read is 5555 the board has 256 Kbytes of memory installed. If the value is zero, continue.

[6] Read a word from 100000. If the value read is 5555 the board has one megabyte of memory installed. If the value is zero, the board has 4 megabytes of memory.

10.4 Bus Memory

All physical addresses from the end of on-card RAM to the beginning of the EPROM at FA0000 are assumed to be off-card. Off-card accesses may be directed at either Multibus or iLBX bus depending on the setting of the Memory Control Word. Bits M/L0 through M/L7 select Multibus or iLBX bus as follows:

Bit	Setting	Function
M/L7	0	700000 - 7FFFFF accesses the Multibus
	1	700000 - 7FFFFF accesses the iLBX bus
M/L6	0	600000 - 6FFFFF accesses the Multibus
	1	600000 - 6FFFFF accesses the iLBX bus
M/L5	0	500000 - 5FFFFF accesses the Multibus
	1	500000 - 5FFFFF accesses the iLBX bus
M/L4	0	400000 - 4FFFFF accesses the Multibus
	1	400000 - 4FFFFF accesses the iLBX bus
M/L3	0	300000 - 3FFFFF accesses the Multibus
	1	300000 - 3FFFFF accesses the iLBX bus
M/L2	0	200000 - 2FFFFF accesses the Multibus
	1	200000 - 2FFFFF accesses the iLBX bus
M/L1	0	100000 - 1FFFFF accesses the Multibus
	1	100000 - 1FFFFF accesses the iLBX bus
M/L0	0	RAM top - 0FFFFF accesses the Multibus
	1	RAM top - 0FFFFF accesses the iLBX bus

Table 19. Bus Selection Control Bit definitions

Multibus and iLBX memory may be mixed as desired. But, before iLBX memory can be used, it must be enabled by setting bit D11 of the the Memory Control Word. For example, to set up one megabyte on-card, the second and fourth megabyte on iLBX, and the remaining memory on the Multibus, write a 0A0A (hex) to the Memory Control Word. If iLBX is disabled (default), the HK68/M120 is effectively disconnected from the iLBX bus. The HK68/M120 operates only in "master" mode on the iLBX bus.

Note: The Hbug-M120 program will automatically enable iLBX memory, if iLBX memory is present, and set the Memory Control Word. Refer to the Hbug-M120 manual for details.

There are two areas in the memory map where the Multibus and iLBX are accessible. The region between on-card RAM and FA0000 (discussed above) allows bus memory to be contiguous with on-card RAM. In addition, the entire Multibus and iLBX regions are accessible starting at 0100,0000 and 0200,0000, respectively. Section 11 describes the bus interface.

In systems using multiple HK68/M120 processors, each board can map its on-card RAM into different Multibus address spaces by use of the bus mapping PLE logic. This will allow all boards access to each other's RAM.

Bus I/O is mapped at address FF0000; see section 11.2 for more detail.

10.5 Physical Memory Map

See section 20.2 for an I/O device address summary.

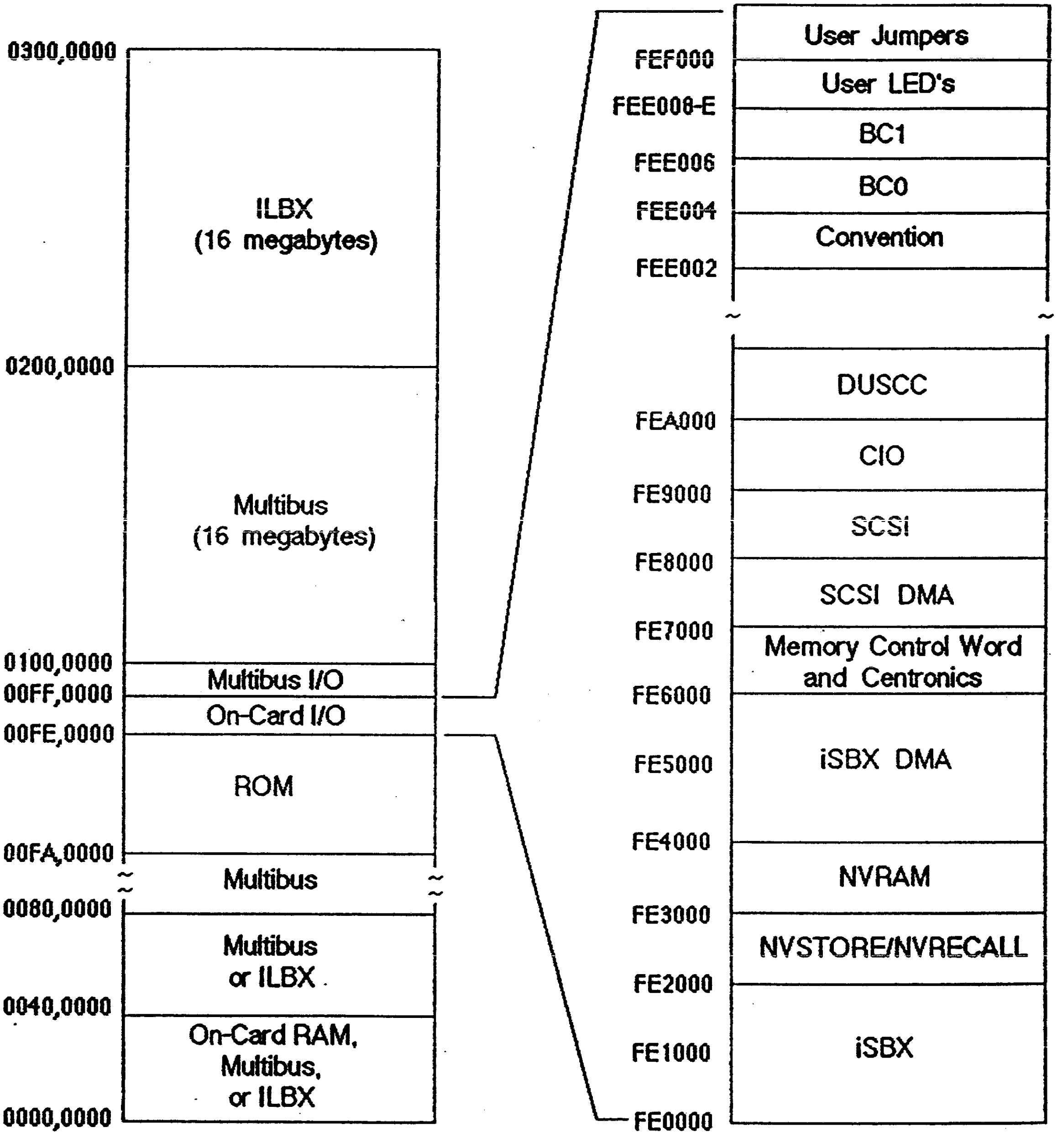


Figure 6. HK68/M120 Physical Memory Map

10.6 Memory Timing

The HK68/M120 memory logic has been carefully tuned to give optimum memory cycle times under a variety of conditions. Considerations have been given to these factors:

[1] The MMU, if present, delays the generation of stable physical addresses on MPU accesses. (Translation Time.)

[2] Typical access times for ROMs are 100 to 200 nanoseconds longer than RAM. Since most programs will be in RAM (or could at least be copied to RAM for execution), ROM timing need not be optimized.

[3] Dynamic memory refreshing must be fast enough that a lengthy (or infinite) bus access cycle will not cause loss of the RAM contents. If a long access from the bus to on-card RAM occurs, which would be terminated by the Watchdog Timer, refreshing must resume and a complete refresh cycle must be done before the maximum refresh time allowed by the RAMs expires. Refreshing operates normally during accesses from the Multibus which are redirected to the iLBX, and all accesses to the bus. The HK68/M120 uses a hardware refresh.

Depending on the RAM speed, extra clock cycles are inserted in memory references to synchronize the MPU with the MMU and memory. The number of extra clock cycles required (in addition to the cycles built into the instruction timing charts) for a (RAM) memory read are shown below.

Condition (120 nsec. RAM speed)	Extra Cycles (at 16.67 MHz)	Total Cycles (at 16.67 MHz)
MPU on-card RAM access, no MMU	1	4
MPU on-card RAM access, with MMU	2	5
pseudo-DMA on-card RAM access	1	4

Table 20. On-card Memory Cycle Timing (RAM)

As faster DRAMs become available, the number of wait states will be reduced, according to the following chart (assumes no MMU).

	---------------Required DRAM Speeds for---------------		
MPU Speed	Zero Waits (r/w)	Zero Waits (read)	One Wait (read)
20.0 MHz		60 nsec. (est)	110 nsec. (est)
16.67 MHz		75 nsec. (est)	135 nsec.
12.5 MHz	80 nsec.	100 nsec.	180 nsec.

The following chart can be used to estimate relative MPU/RAM performance based on the MPU speed, RAM access time and percentage of cache hits. The first column of figures is the performance value if there are no memory accesses, i.e., the cache is hit 100% of the time. The remaining columns show the performance figures for various RAM speeds. The "100% Cache Hits" column shows the maximum performance; the other columns show the minimum performance. The actual value depends on the actual cache hit ratio; your mileage may vary. The chart takes into account the number of wait states required to access the RAM.

Assumptions: No MMU installed. With an MMU, there will be an additional wait state for each RAM access.
All RAM cycles are reads; in some cases, there is an additional wait state for a write.

Computations: Performance Value = MPU.Clock.Rate / Cycles.per.access
Cycles.per.access = 3 + wait.cycles.per.access

	100%	---------RAM SPEED---(no cache hits)----------			
MPU Speed	Cache Hits	60 nsec.	100 nsec.	120 nsec.	150 nsec.
25 MHz	8.33	6.25	(5.00)	5.00	4.17
20 MHz	6.67	6.67	(5.00)	5.00	4.00
16 MHz	5.33	5.33	(4.00)	4.00	3.20
12.5 MHz	4.17	(4.17)	4.17	(3.13)	3.13

Table 21. Relative MPU/RAM Performance Figures

Since the performance figures are anchored at zero, the performance of a system with a value of eight will be twice that of a system with a value of four. The numbers in parenthesis e.g, "(5.00)", represent a cell in the chart which is better implemented (less costly) by using the next slower RAM speed.

The user must weigh the tradeoffs between performance and the costs of faster MPU or memory.

The HK68/M120 uses hardware logic to control refreshing of the dynamic memory. The refresh clock runs at is 76,800 Hz. Thus, one row of the RAM array is refreshed every 13 microseconds. Worst case conditions result in a speed penalty of about 1.5% to accommodate the refresh cycles.

Memory timing is controlled by jumpers J13 and J14 which select the proper delays for DRAM address multiplexing, RAS/CAS timing and DTACK response. These jumpers are factory set; please don´t fiddle with them.

10.7 Non-Volatile RAM

A unique feature of the HK68/M120 is its non-volatile RAM (NV-RAM), which allows precious data or system configuration information to be stored and recovered across power cycles. The RAM is configured as 256, four-bit words (low half of a byte). When the MPU reads a byte of data from the NV-RAM, the upper four bits of the value it receives are indeterminate. The NV-RAM is accessible as shown below.

Address	Mode	Function
00FE,30xx	R/W	Read/Write RAM contents (4 bits).
00FE,2000	Read	Recall RAM contents from Non-volatile memory.
00FE,2000	Write	Store RAM contents in Non-volatile memory. The 68020 "tas" (test and set) instruction must be used for this operation.

Table 22. Non-Volatile RAM Addresses

Physically, the NV-RAM (a Xicor X2212 or equivalent) consists of a static RAM overlaid bit-for-bit with a non-volatile EEpROM. The store operation takes 10 milliseconds to complete. Recall time is approximately one microsecond. Allowances for those delays should be made in software, since the memory hardware does not stop the MPU during the store or recall cycles. The chip is rated for 10,000 store cycles, minimum. During a store operation, only those bits which have been changed are "cycled". The use of a "tas" instruction helps prevent an unintentional store operation by an errant program or a power failure glitch.

At power-up, the shadow RAM contents are indeterminate. Do a recall operation before accessing the NV-RAM for the first time. Recall cycles do not affect the device lifetime.

The HK68/M120 monitor (Hbug-M120) and certain system programs use the NV-RAM. The exact amount reserved for Heurikon usage depends on the system. A major portion of the RAM, however, is available for customer use. Heurikon usage is summarized below (details are available separately).

Function
Magic Number
Checksum
Accumulated number of writes
Board type, serial number and revision level
Hardware configuration information
Software configuration information
System configuration information

Table 23. NV-RAM Contents (partial)

11. MULTIBUS CONTROL

The control logic for the Multibus (IEEE-796) allows numerous bus masters to share the resources on the bus. The control logic for the Multibus is divided into the following sections:

[1] On-card going off (access TO the bus)

[2] Off-card coming on (access FROM the bus)

[3] Bus Interrupts

[4] Data Convention control (relative byte locations)

11.1 Bus Control Signals

The following signals on connector P1 and P2 are used for the Multibus interface. For a complete listing of the P1 and P2 pin descriptions, refer to section 19.

11.1.1 P1, Primary System Bus

P1-13 BCLK/ Bus Clock. A 10 MHz clock generated by the highest priority master board on the bus. This signal is used to synchronize all bus requests and arbitration.

P1-15 BPRN/ Bus Priority In. A low level indicates that no higher priority master needs the bus.

P1-16 BPRO/ Bus Priority Out. A low level means that neither this board nor any higher priority board needs the bus.

BPRN/ and BPRO/ form a daisy chain for priority resolution. BPRO/ of each processor board is connected to the BPRN/ pin on the next lowest priority processor. BPRN/ of the highest priority board should be tied low by installing jumper J18 on that board.

P1-17 BUSY/ Bus Busy. A low level means that the bus is in use.

P1-18 BREQ/ Bus Request. A low level indicates that this board needs the bus. This signal may be used to implement a parallel priority arbitration scheme instead of a daisy chain. BREQ/ for each slot on the bus is independent of all other BREQ/ lines; i.e., this signal is not bussed.

P1-25 LOCK/ Bus Lock. This signal is used to prevent the target board from releasing the facilities between a pair of bus accesses. This is necessary to implement "test and set" types of instructions which use read/ modify/write cycles. If true (low) during an access FROM the bus, the HK68/M120 board will not release the on-card bus to the MPU between bus cycles, unless the Watchdog timer expires. During an access TO the bus, LOCK will be true whenever the MPU Address Strobe (AS) signal is on. Not all Multibus compatible boards support this function.

P1-29 CBRQ/ Common Bus Request. This signal is common for all processors in a system. A low level indicates that there is a bus request pending from a processor which is not already using the bus, regardless of priority. This signal allows a processor to maintain control of the bus, whether actively using the bus or not, until such time as there is another processor needing the bus. This method reduces the bus arbitration time in the absence of multiple bus requests, since the processor last using the bus can "keep" it until another board actually needs it.

P1-31 CCLK/ Constant Clock. The highest priority bus master provides this signal to the bus. The HK68/M120 provides a 9.83 Mhz clock signal.

P1-27 BHEN/ Byte High Enable. This signal, when true, indicates that a 16-bit bus operation is in progress. Otherwise, bus data transfers are 8-bit bytes, on the lower eight bits.

P1-23 XACK/ Transfer Acknowledge. At the completion of a bus operation, the target board (slave) generates this signal to indicate that the operation has been completed. Data is valid for a read or has been written for a write. XACK synchronizes all transfers over the bus and allows devices of various speeds to use the bus.

P1-14 INIT/ Initialize. This is the hardware reset line. It may be either an input or an output, as determined by the setting of jumper J17. When used as an output, the MPU can activate this signal by executing a ´reset´ instruction.

P1-19 MRDC/ Memory Read Command.

P1-20 MWTC/ Memory Write Command. These two signals control memory reads and writes. They indicate that the bus address is valid and, for writes, that the data bus is valid. The master processor waits for XACK/ before terminating the command.

P1-21 IORC/ I/O Read Command.

P1-22 IOWC/ I/O Write Command. The 64K of physical address space from 00FF,0000 through 00FF,FFFF maps into bus I/O commands. Reading a byte (or word) generates IORC/; writing data generates IOWC/. These signals are outputs only on the HK68/M120.

P1-35 to P1-42 Bus Interrupt lines INT0/ to INT7/. (8 lines) The bus supports eight interrupts. The HK68/M120 uses port B of the CIO to monitor these lines and interrupt the MPU when one is active. They may also be used as a general purpose parallel I/O port, if desired.

P1-33 INTA/ Interrupt Acknowledge. Not used.

P1-43 to P1-58 Bus Address lines ADR0/ to ADRF/. (16 lines)

P1-28, 30, 32, 34 Bus Address lines ADR10/ to ADR13/. (4 lines)

P2-55 to P2-58 Bus Address lines ADR14/ to ADR17/. (4 lines)

P1-59 to P1-74 Bus Data lines DAT0/ to DATF/. (16 lines) Note: DATF/ is the most significant data bit, as it should be, to allow communication with 16 bit I/O devices.

11.1.2 P2, Expansion Bus

P2-1 to P2-17 iLBX Bus Data lines DB0 to DB15. (16 lines)

P2-19 to P2-44 iLBX Address lines AD0 to AD23. (24 lines)

P2-46 iLBX ACK/ Acknowledge. At the completion of a bus operation, the target board (slave) generates this signal to indicate that the operation has been completed. Data is valid for a read or has been written for a write. ACK synchronizes all transfers over the iLBX and allows devices of various speeds to be used.

P2-47 iLBX BHEN Byte High Enable. A high level indicates that an access is being done on the high half of the iLBX bus. The value of AD0 determines whether or not the operation is an upper byte or a full word access.

P2-48 iLBX R/W Read/Write Control. A low level indicates that a write operation is being done onto the iLBX bus.

P2-49 iLBX ASTB/ Address Strobe. A low level indicates that the iLBX address is stable.

P2-50 iLBX DSTB/ Data Strobe. A low level indicates that the iLBX Data is stable (write) or that the slave device should place data on the data bus (read).

P2-51 iLBX SMRQ/ Secondary Master Request. This input signal indicates that the other bus master wants control of the iLBX bus.

P2-52 iLBX SMACK/ Secondary Master Acknowledge. This output signal indicates that the HK68/M120 has relinquished the iLBX bus and the other master board may use it.

P2-53 iLBX LOCK/ Bus Lock. This signal is not used by the HK68/M120. It will always be high.

P2-60 iLBX TPAR/ Parity bit. This signal is not used by the HK68/M120. It will always be high.

11.2 On-card going off (TO the Multibus)

This section applies to Multibus memory only (P1); not iLBX (P2). See section 10.4 for information on the iLBX bus.

When the MPU or DMAC makes a request for Multibus facilities, the arbitration logic takes over. If necessary, the requesting board enters a wait state until the bus is available (but only for the maximum time allowed by the Watchdog timer). When the requested operation is completed, the bus will be released according to the state of the two control signals, BC1 and BC0. These signals are under software control.

BC1	BC0	Bus release status
0	0	Release bus after every operation.
0	1	Release bus if any other board has a request for the bus. (Uses CBRQ/)
1	0	Release the bus only if a higher priority board has a request for the bus. (Uses BPRN/)
1	1	Never release the bus, once acquired. This state can be used to capture the bus.

Table 24. Bus Control Bits

The bus control bits are set (or reset) by writing a one (or zero) to following locations: (The default state at power-on is zero.)

Bit	Address (write-only)
BC1	FEE006
BC0	FEE004

Table 25. Bus Control Bit Addresses

Logical addresses aimed at the bus are mapped by the MMU in the same manner as on-card memory. Section 10.4 describes the bus memory configuration.

Although I/O requests from the bus are ignored, it is possible to generate an I/O command to the bus. The 64K physical addresses from 00FF,0000 through 00FF,FFFF are mapped to the bus as I/O commands. For example, to do an "OUTPUT" to a bus device with an I/O address of 48 (hex), do a ´move byte´ instruction specifying a physical destination of FF0048. Since a 64K-byte space is reserved for this function, eight or 16-bit device addressing is supported. If you want to guarantee that all bytes are transferred over the lower eight data lines, use byte mode instructions.

11.3 Off-card coming on (FROM the bus)

The conventional method of board assignment in the Multibus address space is to utilize a group of DIP switches or jumpers to specify a base address for each board. The Heurikon HK68/M120 uses a bus mapping PLE which monitors the Multibus for particular combinations of the upper eight Multibus address lines. The PLE may be programmed with up to eight different address space areas. Since the upper eight address lines are used, the board may be mapped into any 64K address block. By ignoring the state of some of the lower address lines, the size of the address space may be enlarged (up to eight megabytes). Refer to the "Bus Map" section, below.

Once a valid bus request has been detected, an on-card bus request is generated to the MPU. When the current cycle is completed, the MPU will release the on-card bus. The Multibus address and data are then gated on. The bus address lines are utilized as follows:

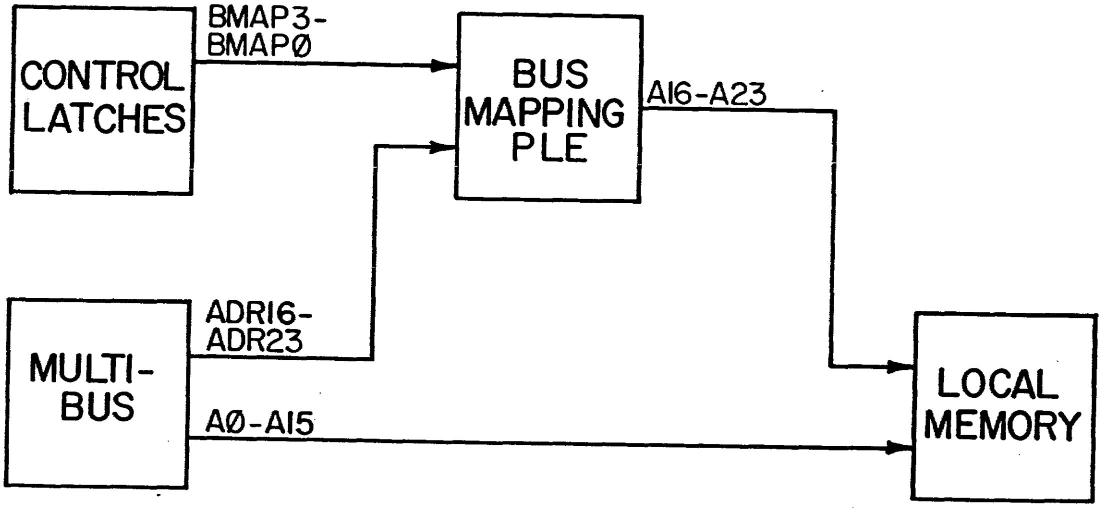


Figure 7. Memory Accesses from Multibus

Note that it is possible for another master to come on-card (from P1) and access memory on the iLBX bus (P2). iLBX memory is treated exactly like local memory. The HK68/M120 operates only as a "master" on the iLBX bus.

I/O commands FROM the bus are ignored. The HK68/M120 does not monitor IORC/ or IOWC/.

Lengthy memory cycles (hundreds of microseconds), originating from the bus, should be avoided. RAM refreshing is suspended during an access by the bus. When XACK is received from the HK68/M120 board, the master processor must terminate the bus request. If the Watchdog Timer is enabled (D10 of the Memory Control Word equals zero), any long access from the bus will automatically be aborted.

11.4 Bus Map (Slave Mode)

The MPU selects the address space by setting the four Bus Map Control lines, in the Memory Control word. The Memory Control word is defined in Memory Configuration, section 10.2. The bus mapping bits are defined in the following table:

BMAP- 3	2	1	0	Space	Standard Configuration
0	x	x	x	0-7	(Future)
1	0	0	0	8	1 Mbyte of memory at 000000
1	0	0	1	9	4 Mbytes of memory at 000000
1	0	1	0	10	4 Mbytes of memory at 300000
1	0	1	1	11	1 Mbyte of memory at 600000
1	1	0	0	12	1 Mbyte of memory at 500000
1	1	0	1	13	1 Mbyte of memory at 400000
1	1	1	0	14	1 Mbyte of memory at 300000
1	1	1	1	15	Accesses disallowed from the bus

Table 26. Bus Map - Off-card Coming On.

Note that the actual configuration may be changed by custom programming the bus mapping PLE (U80). The smallest block size which may be programmed is 64 Kbytes and the memory blocks must start on 64K boundaries. The PLE allows flexible control in that one PLE entry can specify multiple memory blocks at multiple addresses. See section 11.8 for more information.

11.5 Bus Interrupts

The eight Multibus interrupts are monitored and controlled by the CIO chip. By programming the CIO, certain combinations of the bus interrupt lines may be monitored, and a vectored interrupt to the MPU can be generated when the desired state is realized. Also, by selectively programming the CIO lines as outputs, bus interrupts may be generated under software control.

See "CIO USAGE," section 13, for details.

11.6 Bus Data Conventions

The Motorola data convention specifies that higher order bytes of a word are stored in lower address cells. This is opposite to the Intel convention. The following tables show how the two methods would store two 16-bit words (1234 5678):

	Motorola		Intel	
Address	D15-D8	D7-D0	D15-D8	D7-D0
000000	12			34
000001		34	12	
000002	56			78
000003		78	56	

Table 27. Multibus Byte Ordering Conventions

This can create some problems in systems using both types of processors. The Heurikon HK68/M120 has a bus convention control bit (at device address FEE002) which may be used to deal with these problems.

Address	Function (write-only)
FEE002	Set/Clear CONVENTION bit
	0 (false) = Normal
	1 (true) = Swap

If CONVENTION is false (default), memory devices on the bus appear to store bytes in Motorola style. If CONVENTION is true, Intel style is used for byte transfers. This signal also controls the translation of bus address bit ADR0 into Upper and Lower Data Strobes (UDS, LDS) during a byte access FROM the bus. If zero, a byte access from the bus with ADR0 true operates on the lower half of a word (Motorola convention). The CONVENTION bit affects only byte mode Multibus operations and is used to simply invert the sense of bus signal ADR0.

There are side effects from using either convention:

- Using Normal convention (Motorola style) will sometimes require software byte reordering when reading from Intel processors. This usually shows up when using an I/O device which deals with character streams. Character pairs read in word mode (two characters at a time) will appear swapped, however 16-bit numeric values read in byte mode will be okay. Many intelligent peripherals provide hardware or software byte reordering. However, if you are using one which does not, the program must exchange all of the even and odd bytes of a stream read in word mode. Heurikon uses the Normal convention for all of its system software.

- Using Swap convention (Intel style) precludes running 68020 programs that use Multibus memory. Although character streams will be correct in either word or byte mode, the MPU will get confused after writing a word numeric value to memory and trying to read one byte of that word. It will always read the wrong byte. For example, the MPU will read the LSB when the software expects the MSB. Intel style should only be used when it is necessary to communicate with an Intel-based intelligent device and no Multibus memory is required for the program running on the HK68/M120.

The charts on at the end of this section can be used to determine how various system configurations will operate. Don´t spend too much time trying to understand the bus data conventions unless you have a specific need; it is definitely headache material.

All MPU generated Multibus byte operations are conducted on the lower eight Multibus data lines, with DAT7/ being the most significant bit. For word transfers, DATF/ is the most significant bit, as per the Multibus specification. Caution: Some Multibus I vendors always put the MSB on DAT7/, even for 16-bit transfers!

11.7 Watchdog Timer

The HK68/M120 has a timer which monitors board activity. If the timer is enabled and if the on-card physical address strobe stays on for approximately one millisecond, the timer will expire. This will cause the current memory cycle to be terminated. The timer is disabled by setting D10 of the Memory Control Word to one.

See section 9.1 for more details on the watchdog timer.

11.8 Mailbox

The HK68/M120 mailbox logic generates a MPU interrupt if a Multibus access is detected within a certain address range. The address range is specified by the contents of the bus map PLE (U80). The interrupt is generated by the CIO. Jumper J19 is used to enable/disable the mailbox logic (install to enable).

The bus map PLE sets the base and size of the mailbox address range. Usually, the mailbox size is only 64K bytes, which is the smallest range possible. One general purpose PLE, part number M120-MAP-00, is detailed below. Other variations may be available; please consult the factory.

	Window	Window	Mailbox	Mailbox
Entry	Bus Base	Size	Bus Base	Size
0-7	(future)		(future)	
8	0x000000	1Meg	0x700000	64K
9	0x000000	4Meg	0x700000	64K
10	0x300000	4Meg	0x7D0000	64K
11	0x600000	1Meg	0x710000	64K
12	0x500000	1Meg	0x750000	64K
13	0x400000	1Meg	0x790000	64K
14	0x300000	1Meg	0x7D0000	64K
15	(off)	n/a	(off)	n/a

Table 28. Bus Mapping PLE (M120-MAP-00)

The on-card base address for all windows and the mailbox regions is 0x000000. The "window" region is an area defined for a standard Multibus slave access. For example, entry 12 specifies that the HK68/M120 is a Multibus slave in the region 0x600000 through 0x6fffff. Also, entry 12 defines the mailbox to be at Multibus addresses 0x740000 through 0x74ffff. A Multibus read or a write in that range will generate a mailbox interrupt. The HK68/M120 responds as a normal bus slave in the mailbox region (as long as J19 is installed).

The PLE has 12 inputs and 8 outputs. Four of the inputs are bus map selection bits. The others are the eight most-significant Multibus address lines. The input pin assignments can be found on the HK68/M120 schematic diagram.

Of the PLE outputs, one indicates when an address match has occurred for the mailbox, another signals an address match for a general slave access of the HK68/M120, and the remaining six specify the on-card memory base address for the access. Output pin allocations are detailed below.

Pin	Function	Polarity
15	slave access	Positive true
14	mailbox area	Negative true
17	on-card A21	Negative true
16	on-card A20	Negative true
13	on-card A19	Negative true
11	on-card A18	Negative true
10	on-card A17	Negative true
9	on-card A16	Negative true

Table 29. Mailbox PLE (U80 Pin Allocations

The mailbox interrupt signal is connected to CIO port A3. (Refer to section 13). The mailbox interrupt signal is a short pulse, so the CIO input must be programmed as a "ones-catcher".

Use this checklist when programming the HK68/M120 mailbox logic:

[1] During initialization, CIO registers 0x24, 0x25 and 0x27 must have bit D3 (0x08) set on. This specifies the proper polarity for the mailbox interrupt signal and enables the ones catcher.

[2] Set a MPU exception vector to point to your mailbox interrupt service routine. If the CIO vectors suggested in this manual are used, that will be vector number 71.

[3] Include code to save and restore the MPU registers.

[4] Reset the CIO interrupt in your interrupt service routine using the following code sequence:

```
#include "sys/m10.h"      /* defines (struct cio *)CIOADDR */
#include "sys/cio.h"      /* defines cio structure */
mailboxintr()       /* from vector 71, after register save */
{
    static int  zero = 0; /* avoids the clr.b instruction */
    CIOADDR->adata = zero;  /* clear port A data register */
    CIOADDR->ctrl = 0x08;   /* port A control register    */
    CIOADDR->ctrl = 0x20;   /* reset CIO interrupt */
    /* process the interrupt request here */
    return;                 /* restore regs and do rte */
}
```

Figure 8. Sample Mailbox Interrupt Service Routine

When used with UNIX, certain precautions are needed to assure correct CIO ones catcher operation. Avoid changing the CIO mask register when using a ones catcher, otherwise the mailbox interrupt pulse may be missed.

11.9 Relevant Jumpers - Bus Control

Jumper	Function	Position	Notes
J15	Bus BCLK Enable		Install on highest priority board
J16	Bus CCLK Enable		Install on highest priority board
J17	Bus INIT/ Select	J17-A	INIT/ is an Input
		J17-B	INIT/ is an Output
J18	Bus BPRN Enable		Install on highest priority board
J19	Mailbox Enable		Install to enable the mailbox logic

Table 30. Bus Control Jumpers

Item	Description
HK68/M120 Master	When the HK68/M120 is the master board.
Intel Master	When an Intel-type board is the master.
Target	Slave board type and configuration.
Word Mode	16-bit transfers
Byte Mode	8-bit transfers
CONV false	HK68/M120 CONVENTION bit is false (0).
CONV true	HK68/M120 CONVENTION bit is true (1).
[00,02,04,...]	The sequence of address values applied to the bus as the Master cycles through what it thinks are consecutive addresses.
upper	The higher order byte of a word.
lower	The lower order byte of a word.
even	The LSB address bit on the associated master, bus or target is false.
odd	The LSB address bit is true.
MASTER=upper,even	What the Master board thinks it is doing.
BUS=odd address	What the Multibus sees.
BUS=...,BHEN	Byte High Enable is on, else BHEN is off.
SLAVE=upper,odd	What the target board does.
[BYTES SWAPPED]	Upper and lower bytes of 16-bit numeric values appear swapped when read using a different mode (byte or word) than that used to write the values.
[BYTES OKAY]	Upper and lower bytes of 16-bit numeric values appear as the MPU expects.
[CHARS SWAPPED]	Character pairs appear to be swapped when read using a different mode (byte or word) than that used to write the characters.
[CHARS OKAY]	Characters always appear in the proper order, regardless of mode.
[ALL SWAPPED]	Character pairs and upper and lower bytes of 16-bit numeric values written in one mode appear swapped when accessed using a different mode.
[OKAY]	Character pairs and bytes of 16-bit numeric values appear as the MPU expects.

Table 31. Multibus Convention Logic - Key for Table 32

		HK68/M120 Master	HK68/M120 Master
	HK68/M120 Master	Byte Mode	Byte Mode
Target	Word Mode	CONV false	CONV true
	[00,02,04,...]	[01,00,03,02,...]	[00,01,02,03,...]

Intel
Memory
	MASTER=upper,even
	MASTER=lower,odd
	BUS=even adrs,BHEN
	SLAVE=upper,odd
	SLAVE=lower,even
	[BYTES OKAY]
	[CHARS SWAPPED]
		MASTER=upper,even
		BUS=odd address
		SLAVE=upper,odd
		[BYTES OKAY]
		[CHARS SWAPPED]
			MASTER=upper,even
			BUS=even address
			SLAVE=lower,even
			[BYTES SWAPPED]
			[CHARS OKAY]

HK68
CONV false
	MASTER=upper,even
	MASTER=lower,odd
	BUS=even adrs,BHEN
	SLAVE=upper,even
	SLAVE=lower,odd
	[OKAY]
		MASTER=upper,even
		BUS=odd address
		SLAVE=upper,even
		[OKAY]
			MASTER=upper,even
			BUS=even address
			SLAVE=lower,odd
			[ALL SWAPPED]

HK68
CONV true
	same as above
	[OKAY]
		MASTER=upper,even
		BUS=odd address
		SLAVE=lower,odd
		[ALL SWAPPED]
			MASTER=upper,even
			BUS=even address
			SLAVE=upper,even
			[OKAY]

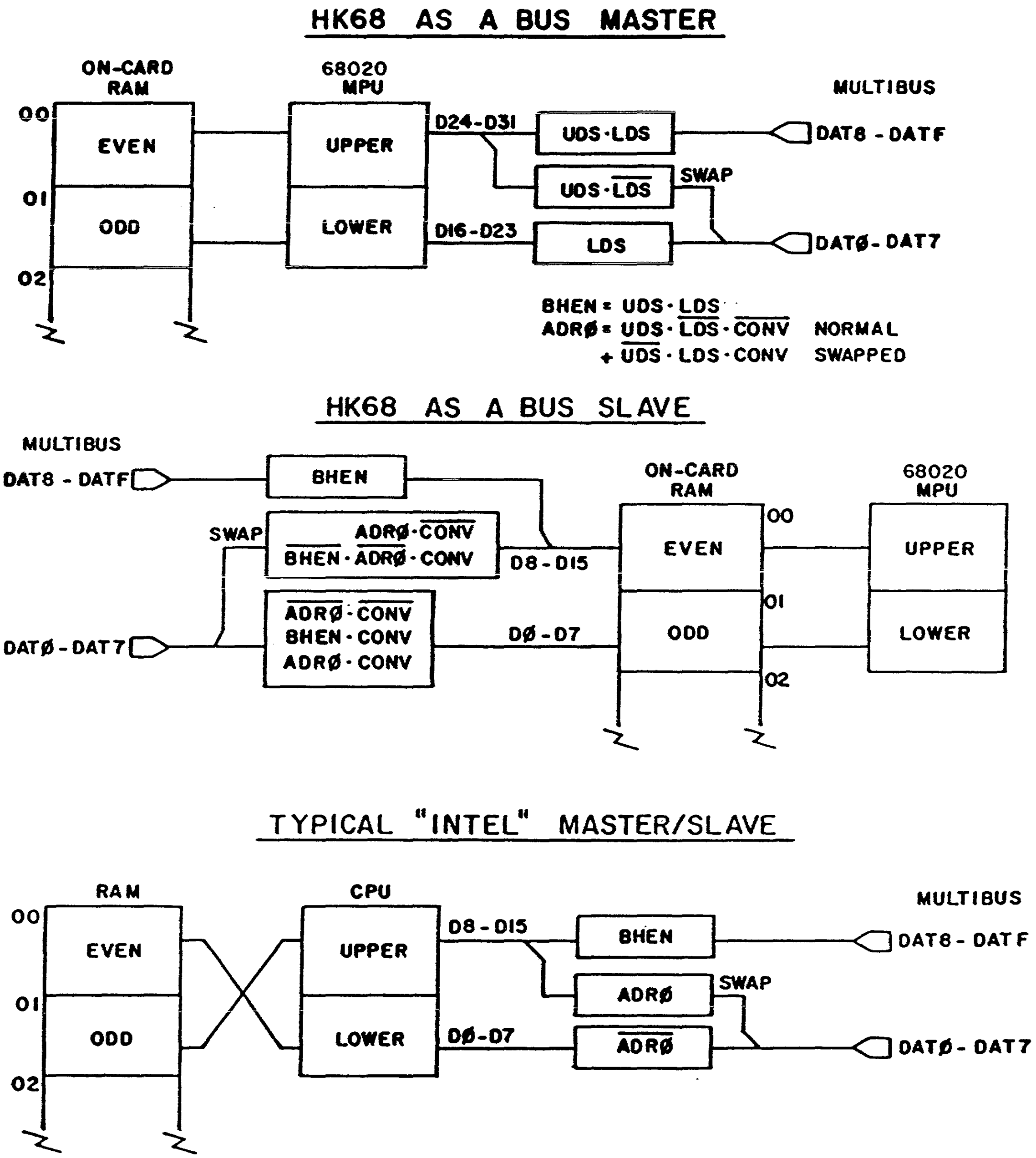
	Intel Master	Intel Master
Target	Word Mode	Byte Mode
	[00,02,04,...]	[00,01,02,03,...]

Intel
Memory
	MASTER=lower,even
	MASTER=upper,odd
	BUS=even adrs,BHEN
	SLAVE=lower,even
	SLAVE=upper,odd
	[OKAY]
		MASTER=lower,even
		BUS=even address
		SLAVE=lower,even
		[OKAY]

HK68
CONV false
	MASTER=lower,even
	MASTER=upper,odd
	BUS=even adrs,BHEN
	SLAVE=lower,odd
	SLAVE=upper,even
	[BYTES OKAY]
	[CHARS SWAPPED]
		MASTER=lower,even
		BUS=even address
		SLAVE=lower,odd
		[BYTES OKAY]
		[CHARS SWAPPED]

HK68
CONV true
	same as above
	[BYTES OKAY]
	[CHARS SWAPPED]
		MASTER=lower,even
		BUS=even address
		SLAVE=upper,even
		[BYTES SWAPPED]
		[CHARS OKAY]

Table 32. Multibus Convention Logic - Detail


HK68 AS A BUS MASTER

HK68 AS A BUS SLAVE

TYPICAL "INTEL" MASTER/SLAVE

Figure 9. Bus Data Interface - Logic Diagram

12. MISCELLANEOUS DEVICES

12.1 User Jumper Input Port

Jumpers J12-1 through J12-8 may be read by the MPU. They have no predefined function. Reading from location FEF003 will give the jumper settings as follows:

Jumper	Bit
J12-1	D7
J12-2	D6
J12-3	D5
J12-4	D4
J12-5	D3
J12-6	D2
J12-7	D1
J12-8	D0

Table 33. User Jumpers - Bit Definitions

Jumper State	Input Value
open	zero (0)
installed	one (1)

Table 34. User Jumpers - Input States

12.2 User LEDs

There are four LEDs located near the P6 connector whose meanings may be defined by the program.

LED Number	Address (write-only)
4	FEE00E
3	FEE00C
2	FEE00A
1	FEE008

Table 35. User LEDs - Addresses

Writing a zero turns the chosen LED on; writing a one will turn it off. At power-on or after a system reset, the LEDs will be ON.

13. CIO USAGE

The on-card CIO device performs a variety of functions. In addition to the three 16-bit timers which may be used to generate interrupts or count events, the CIO has numerous parallel I/O bits.

13.1 Port A Bit Definition

Port A controls the SBX interface and handles various interrupt signals. The function and polarity of the bits are described below. All bits should be programmed as inputs.

Bit	Function	Polarity	Interface	Reference
D7	External Interrupt	Negative True	P5-11	5.3
D6	SBX INTO	Positive True	P7-14	17
D5	SCSI Data Ready	Negative True	-	15
D4	Centronics Interrupt	Positive True	(P3-10)	16
D3	Mailbox Interrupt	Negative True	-	11.8
D2	SBX INT1	Positive True	P7-12	17
D1	SBX Data Ready	Positive True	P7-34	17
D0	SBX MPS/	Negative True	P7- 8	17

Table 36. CIO Port A Bit Definitions

13.2 Port B Bit Definition

The eight Multibus interrupts are monitored and controlled by port B of the CIO chip. By programming the CIO, certain combinations of the bus interrupt lines may be monitored, and a vectored interrupt to the MPU can be generated when the desired state occurs. Also, by selectively programming the CIO lines as outputs, bus interrupts may be generated under software control.

Bit	Function	Polarity	Interface
D7	Multibus interrupt 0	Negative True	P1-41
D6	Multibus interrupt 1	Negative True	P1-42
D5	Multibus interrupt 2	Negative True	P1-39
D4	Multibus interrupt 3	Negative True	P1-40
D3	Multibus interrupt 4	Negative True	P1-37
D2	Multibus interrupt 5	Negative True	P1-38
D1	Multibus interrupt 6	Negative True	P1-35
D0	Multibus interrupt 7	Negative True	P1-36

Table 37. CIO Port B Bit Definitions

Port B bits may also be programmed as external I/O lines for counter/timer channels 1 and 2. This feature could be used to count rapid external events or indicate count completion.

Internal priorities of the CIO place D7 as highest (D0 as lowest) for simultaneous interrupts from either port.

13.3 Port C Bit Definition

Port C on the CIO chip is not used by the HK68/M120. These lines are available for custom situations.

Bit	Function
D3	Uncommitted
D2	Uncommitted
D1	Uncommitted
D0	Uncommitted

Table 38. CIO Port C Bit Definitions

13.4 Counter/Timers

There are three independent, 16-bit counter/timers in the CIO. Each may be used as a counter in conjunction with the port B lines, which are connected to the Multibus interrupt lines; or as timers to implement real-time clocks for programmed delays. For long delays, timers 1 and 2 may be internally linked together to form a 32-bit counter chain. When programmed as timers, the following equation may be used to determine the time constant value for a particular interrupt rate.

$$TC = 2{,}457{,}600 \ / \ \text{interrupt rate (in HZ)}$$

Equation 2. CIO Timer constant

When the timer is clocked internally, the count rate is 2.4576 MHz. The HK68/M120 board uses a 19.6608 MHz clock oscillator as the system time base. The frequency tolerance specification is +/- 0.01%. If you are using the 19.6608 MHz clock as the CIO time base, the maximum accumulative timing error will be about 9 seconds per day, although the typical error is less than one second per day. Better long-term accuracy may be achieved via a power line (60 Hz) interrupt, using a bus interrupt or the Real-Time Clock (RTC) option (see section 18).

13.5 Register Address Summary (CIO)

Register	Address	Function
Port C, Data	FE9001	Unused
Port B, Data	FE9003	Multibus Interrupts
Port A, Data	FE9005	SBX Control, Mailbox
Control Regs	FE9007	CIO Configuration & Control

Table 39. CIO Register Addresses

13.6 CIO Initialization

The following figure shows a typical initialization sequence for the CIO. The first byte of each data pair in "ciotable" specifies an internal CIO register; the second byte is the control data. The specific directions of some of the PIO lines (e.g. SBX options) and interrupts need to be changed in the table, based on your application. An active low signal can be inverted (so that a "1" is read from the data port when the signal is true) by initializing the port to invert that particular bit. Refer to section 5 for information concerning CIO interrupt vectors.

```
char ciotable[] = {
0x00, 0x01, 0x00,/* reset, set chip ptr to reg zero */

            /* port A initialization */
0x20, 0x06, /* bit port, priority encoded vector */
0x22, 0xa9, /* invert negative true bits */
0x23, 0xff, /* all bits are inputs */
0x24, 0x08  /* mailbox one's catcher */
0x25, 0xc2, /* pattern polarity register */
0x26, 0x00, /* all levels (can't use transitions */
            /* in "or priority mode") */
0x27, 0xc0, /* pattern mask, enable two */
0x02, 0x41, /* set interrupt vector */
0x08, 0xc0, /* set int enable, no int on err */

            /* port B initialization */
0x28, 0x06, /* bit port, priority encoded vector */
0x2a, 0xff, /* bits inverting */
0x2b, 0xff, /* all bits are inputs */
0x2c, 0x00, /* normal input (no ones catchers) */
0x2d, 0xff, /* bit interrupt on a one */
0x2e, 0x00, /* no transition */
0x2f, 0x74, /* enable Int 5,3,2,1 */
0x03, 0x40, /* set interrupt vector */
0x09, 0xc0, /* set int enable, no int on err */

            /* timer 3 and other CIO initialization */
0x1e, 0x80, /* set mode to auto reload */
0x1a, 0xa0, /* high byte delay constant */
0x1b, 0x00, /* low byte delay constant */
0x04, 0x60, /* interrupt vector */
0x08, 0x20, /* clear any port A ints */
0x01, 0x94, /* enable timer 3, port a and port b */
0x0c, 0xc6, /* set interrupt enable and */
            /* gate command bit and trigger cmd bit */
0x00, 0x9e  /* master int enable and vector includes */
            /* status for timer 3, port A and port B */
};
```

```
struct cdevice {   /* CIO register structure */
    char dummy0;  char cdata;      /* port C */
    char dummy1;  char bdata;      /* port B */
    char dummy2;  char adata;      /* port A */
    char dummy3;  char ctrl; /* control port */
};
#define CIO ((struct cdevice *)0xfe9000)

cioinit()
{
    int i, t3intr();
    /* Don´t forget to set all CIO interrupt vectors. For example... */
    *(int(*))(0x60*4) = (int)t3intr; /* Timer 3 interrupt */
    i = CIO->ctrl;             /* assure register sync */
    CIO->ctrl = ciotable[0]; /* avoid a clr instruction */
    i = CIO->ctrl;             /* assure register sync */
    for (i = 0; i < sizeof(ciotable); i++)
        CIO->ctrl = ciotable[i];  /* send ciotable to CIO chip */
}

Aintr()    /* clear Port A interrupt */ /* one of 8 routines */
{   /* process port A interrupts here */
    CIO->ctrl = 0x08; CIO->ctrl = 0x20;
}

Bintr()    /* clear Port B interrupt */ /* one of 8 routines */
{   /* process port B interrupts here */
    CIO->ctrl = 0x09; CIO->ctrl = 0x20;
}

timer3()  /* clear Timer 3 interrupt, get here via t3intr */
{   /* process timer interrupt here */
    CIO->ctrl = 0x0c; CIO->ctrl = 0x24;
}
```

Figure 10. CIO Program Example (C Portion)

```
         .globl   t3intr%, timer3
# the vector at 0x60*4 points to this routine
t3intr%: movm.l   &0xFFFF,-(%sp)  # save registers
         jsr      timer3          # to C portion
         movm.l   (%sp)+,&0xFFFF  # restore registers
         rte
```

Figure 11. CIO Program Example (Assembly Code Portion)

13.7 CIO Programming Hints

[1] To maintain compatibility with 68010 programs, do not use the 68020 "clr.b" instruction to set a CIO register to zero. On the 68000 and 68010, that instruction does a "phantom" read of the port before it does the zero write. The read operation will upset the CIO internal register selection sequencer. Similarly, when using a high level language, do not set a CIO register value to the constant "0" because the compiler may use a "clr.b". Use a variable which is set to zero, or output the values from a lookup table. For example:

```
zero = 0;
*CIOcntrl = 0x20;
*CIOcntrl = zero;
```

[2] The ones catchers in a CIO port will be cleared whenever any bits are changed in the pattern mask register. Avoid changing the mask register if you are using a ones catcher. If this is not possible, a program that writes to the pattern mask register should first OR the CIO data register into a memory variable. Later, that memory value can be ORed with the CIO data register to find out what the data register would have been if the CIO had not cleared it. Routines which respond to a ones catcher interrupt must clear the corresponding bits in the memory value and the CIO data register. There will still be a critical period where a fast input pulse could be missed, even when using this scheme.

[3] If you get an unexpected interrupt from bit D0 of a CIO port, it may be because another enabled CIO input signal went false before the MPU initiated the interrupt acknowledge cycle. The use of a ones catcher may be appropriate to latch the input line.

[4] If you turn on a bit in the pattern mask register, that bit will generate an interrupt (if the port is enabled) even if the input signal is false. To prevent this, disable the port while adjusting the pattern mask register.

[5] The CIO may glitch the parallel port lines when a hardware reset is done, even if all lines are programmed as inputs. This may cause a problem in multi-processor systems because the glitches will produce spurious Multibus interrupts on other (operating) boards. To prevent this effect, disable the port (via software) prior to doing a board reset.

Refer to the Z8536 technical manual for more details on programming the CIO. Some people find the CIO technical manual difficult to understand. We encourage you to read all of it twice, before you pass judgement. Section 5.2 has a list of suggested readings from the CIO manual. Contact us (or Zilog) to obtain application notes.

14. SERIAL I/O

There are two RS-232C serial I/O ports on the HK68/M120 board. Each port may optionally be configured for RS-422 operation with a special interface cable, as detailed in section 14.5 Each port has a separate baud rate generator and can operate in asynchronous or synchronous modes.

14.1 RS-232 Pinouts

Data transmission conventions are with respect to the device. The HK68/M120 board is wired as a "Data Set." The connector pinouts are as follows:

Pin	"D" Pin	RS-232 Function	Direction
x	1	Protective ground	n/c on HK68/M120
P6- 1	14	x	
P6- 2	2	Tx Data	(from device)
P6- 3	15	Tx Clock	(from device)
P6- 4	3	Rcv Data	(to device)
P6- 5	16	x	
P6- 6	4	*Request To Send	(from device)
P6- 8	5	Clear To Send	(to device)
P6- 9	18	x	
P6-10	6	Data Set Ready	(to device)
P6-11	19	x	
P6-12	7	Signal Ground	
P6-13	20	*Data Terminal Ready	(from device)
P6-14	8	Data Carrier Detect (J1)	(from device)
P6-15	21	x	
P6-16	9	x	
P6-17	22	Ring Indicator (J1)	(from device)
x	10-13	x	
x	23-25	x	

Table 40. Serial Port A Pinouts

Pin	"D" Pin	RS-232 Function	Direction
x	1	Protective ground	n/c on HK68/M120
P6-18	14	x	
P6-19	2	Tx Data	(from device)
P6-20	15	Tx Clock	(from device)
P6-21	3	Rcv Data	(to device)
P6-22	16	x	
P6-23	4	*Request To Send	(from device)
P6-25	5	Clear To Send	(to device)
P6-26	18	(+5 via J4)	
P6-27	6	Data Set Ready	(to device)
P6-28	19	x	
P6-29	7	Signal Ground	
P6-30	20	*Data Terminal Ready	(from device)
P6-31	8	Data Carrier Detect (J2)	(from device)
P6-32	21	(+12 via J3)	
P6-33	9	(-12 via J5)	
P6-34	22	Ring Indicator (J2)	(from device)
x	10-13	x	
x	23-25	x	

Table 41. Serial Port B Pinouts

Note that the interconnect cable from P6 is split in such a manner that the "D" connector pinouts are correct for RS-232C conventions. Not all pins on the "D" connectors are used. Recommended mating connectors are Ansley P/N 609-3401CE and Molex P/N 15-29-8348.

Signals indicated with "*" have a default pullup resistor, controlled by J6. NOTE: The serial ports may appear to be inoperative if J6 is set to default "FALSE" and if the device connected to the port does not drive the DTR and RTS pins TRUE. The Hbug-M120 monitor software, for example, initializes the DUSCC channels to respect the state of DTR and RTS. The DCD and RI signals are routed to the CENT Status port by J1 and J2. See section 16.

14.2 Signal Naming Conventions (RS-232)

Since the RS-232 ports are configured as "data sets," the naming convention for the interface signals may be confusing. The interface signal names are with respect to the terminal device attached to the port while the DUSCC pins are with respect to the DUSCC as if it, too, is a terminal device. Thus all signal pairs, e.g., "RTS" & "CTS," get switched between the I/F connector and the DUSCC chip. For example, "Transmit Data," P6-2, is the data transmitted from the device to the HK68/M120 board; the data appears at the DUSCC receiver as "Received Data." For the same reason, the "DTR" and "RTS" interface signals appear as the "CTS" and "DSR" bits in the DUSCC, respectively. If you weren´t confused before, any normal person should be by now. Study the chart below and see if that helps.

DUSCC Signal	I/F Signal	Direction
Tx Data	Rcv Data	to device
Rcv Data	Tx Data	from device
Tx Clock	Rcv Clock	from device
Rcv Clock	Tx Clock	from device
RTS	DSR	to device
CTS	DTR	from device
DTR	CTS	to device
DCD	RTS	from device
	DCD	from device
-	Ring Ind.	from device

Table 42. Signal Naming Conventions

The DUSCC was designed to look like a "data terminal" device. Using it as a "data set" creates this nomenclature problem. Of course, if you connect the HK68/M120 board to a modem ("data set"), then the DUSCC signal names are correct, however, a cable adapter is needed to properly connect to the modem. (Three pairs of signals must be reversed.)

DUSCC Signal	P6 Pin #s	"D" Pin # at HK68/M120	"D" Pin # at modem	RS-232 Signal
x	x	1	1	Prot Gnd
Rcv Data	2 (or 19)	2	3	Rcv Data
Tx Data	4 (or 21)	3	2	Tx Data
DCD	6 (or 23)	4	6	DSR
RTS	10 (or 27)	6	4	RTS
DTR	8 (or 25)	5	20	DTR
CTS	13 (or 30)	20	5	CTS
	14 (or 31)	8	8	DCD
(Ring Ind)	17 (or 34)	22	22	Ring Ind
(Sig Gnd)	12 (or 29)	7	7	Sig Gnd

Table 43. RS-232 Cable Reversal

Summary: The HK68/M120 may be directly connected to a data "terminal" device. A cable reversal is required for a connection to a modem.

14.3 Connector Conventions

Paragraph 3.1 of the EIA RS-232-C standard says the following concerning the mechanical interface between data communications equipment:

> "The female connector shall be associated with...the data communications equipment... An extension cable with a male connector shall be provided with the data terminal equipment... When additional functions are provided in a separate unit inserted between the data terminal equipment and the data communications equipment, the female connector...shall be associated with the side of this unit which interfaces with the data terminal equipment while the extension cable with the male connector shall be provided on the side which interfaces with the data communications equipment."

Substituting "modem" for "data communications equipment" and "terminal" for "data terminal equipment" leaves us with the impression that the modem should have a female connector and the terminal should have a male. The Heurikon HK68/M120 microcomputer interface cables are designed with female "D" connectors, because the serial I/O ports are configured as data sets (modems). Terminal manufacturers typically have a female connector also, despite the fact that they are terminals, not modems. Thus, the extension cable used to run between a terminal and the HK68/M120 (or a modem) will have male connectors at both ends.

If you do any work with RS-232 communications, you will end up with many types of cable adapters. Double males, double females, double males and females with reversal, cables with males and females at both ends, you name it! We will be happy to help make special cables to fit your needs.

14.4 Port Address Summary

Name	Register	Port A	Port B
CMR1	Channel Mode 1	FEA001	FEA041
CMR2	Channel Mode 2	FEA003	FEA043
S1R	SYN/Sec Adrs 1	FEA005	FEA045
S2R	SYN/Sec Adrs 2	FEA007	FEA047
TPR	Transmitter Param	FEA009	FEA049
TTR	Transmitter Timing	FEA00B	FEA04B
RPR	Receiver Param	FEA00D	FEA04D
RTR	Receiver Timing	FEA00F	FEA04F
CTPRH	Counter/Timer Preset	FEA011	FEA051
CTPRL	Counter/Timer Preset	FEA013	FEA053
CTCR	Counter/Timer Control	FEA015	FEA055
OMR	Output and Misc	FEA017	FEA057
CTH	Counter/Timer High (R)	FEA019	FEA059
CTL	Counter/Timer Low (R)	FEA01B	FEA05B
PCR	Pin Configuration	FEA01D	FEA05D
CCR	Channel Command	FEA01F	FEA05F
TXFIFO	Transmitter FIFO (W)	FEA021	FEA061
RXFIFO	Receiver FIFO (R)	FEA029	FEA069
RSR	Receiver Status	FEA031	FEA071
TRSR	TX and Rcv Status	FEA033	FEA043
ICTSR	Input and C/T Status	FEA035	FEA045
GSR	General Status	FEA037	FEA077
IER	Interrupt Enable	FEA039	FEA079
IVR	Interrupt Vector (R/W)	FEA03D	FEA07D
IVRM	Interrupt Vector (R)	FEA03D	FEA07D
ICR	Interrupt Control	FEA03F	FEA07F

Table 44. DUSCC Register Addresses

All ports are eight bits. The RS-232 RI (Ring Indicator) or DCD (Data Carrier Detect) signals are read using the Centronics Status port at FE6000; refer to section 16.2.

14.5 RS-422 Operation

As an option, one or both of the serial ports on the HK68/M120 may be configured for RS-422 operation. To accomplish this, the on-card RS-232 interface chips are removed, and a special cable is used which includes the RS-422 interface logic. This option does not affect "D" pins 4 (RTS), 8 (DCD) or 22 (RI), which remain as RS-232 inputs.

To install the RS-422 option kit, follow these steps:

[1] Remove the RS-232 interface chips for the affected ports. These are U1 and U2 (for port A) and U3 and U4 (for port B).

[2] Install the special headers in the RS-232 interface positions. The headers are wired as follows:

75188 Outputs (U2, U3)	75189 Inputs (U1, U4)
2 to 3	1 to 3
4 and 5 to 6	4 to 6
8 to 9 and 10	8 to 10
11 to 12 and 13	11 to 13

Table 45. RS-422 Header Wiring

[3] Install jumper shunt J4. This supplies power to the RS-422 interface chips which are located on the cable.

14.6 Relevant Jumpers (Serial I/O)

Jumper	Function	Position
J1	Port A: RI, DCD Select	J1-A RI to Centronics Status, D6 J1-B DCD to Centronics Status, D6
J2	Port B: RI, DCD Select	J2-A RI to Centronics Status, D5 J2-B DCD to Centronics Status, D5
J3	+12 power to P6	
J4	+5 power to P6	
J5	-12 power to P6	
J6	RS-232 Status Default	J6-A (True) J6-B (False)

Table 46. Relevant Jumpers - Serial I/O

14.7 Serial I/O Cable Drawing

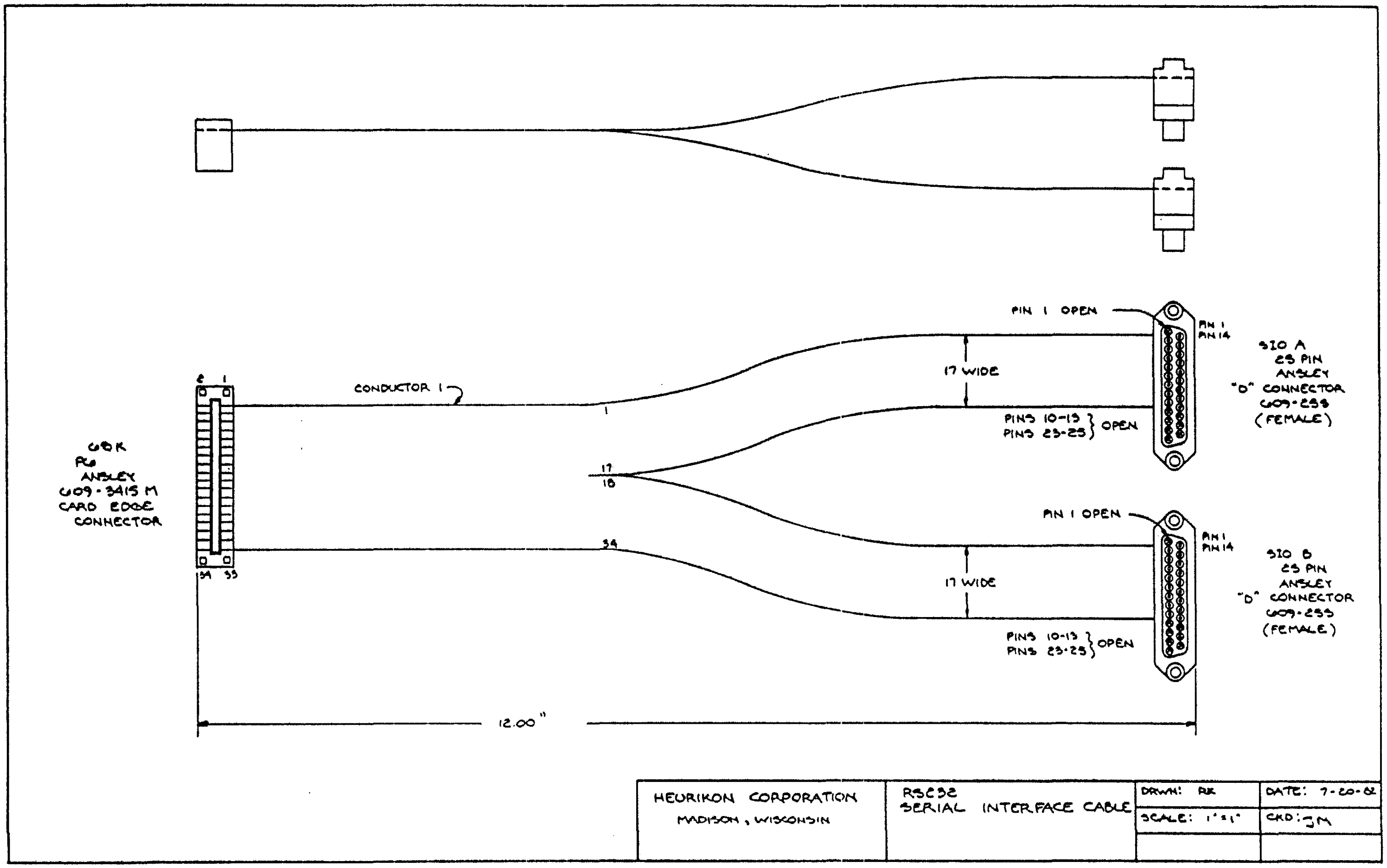


Figure 12. Serial I/O Cable Drawing

15. SCSI PORT

The HK68/M120 uses the Wester Digital WD33C93 chip to implement a Small Computer System Interface (SCSI) port. (Commonly called "Scuzzy".)

The SCSI port may be used to connect to a variety of peripheral devices. Most common are Winchester disks, floppy diskettes, and streamer tape drives.

This port is pseudo-DMA driven, transferring up to 1.5 megabytes per second. Actual rates depend on the characteristics of the attached controller and device being used.

Supported features and modes include:

initiator role
target role
arbitration
disconnect
reconnect
pseudo-DMA interface

15.1 SCSI Implementation Notes

The interrupt from the SCSI chip generates a level 4 autovector. See MPU Exception Vectors, section 5.2 for details. Data transfer functions can be handled in a polled I/O mode or by using the pseudo-DMA functions provided by on-card logic and the MPU.

The pseudo-DMA allows the SCSI data request signal to either interrupt the MPU (via the CIO) or to synchronize the MPU wait signal (DTACK) to the availability of data. The synchronization logic is enabled by accessing the SCSI data register in a special memory mapped area (at FE,7000). By so doing, the MPU will enter a wait state until data becomes available. This feature allows a block transfer of data to or from the SCSI interface to be initiated by an interrupt (when the first byte is ready) and continued through MPU control with minimal overhead.

The MPU will be released from the wait state when either the watchdog expires (indicating that there has been a pause in the data transfer as could happen between sectors) or when the WD33C93 interrupts the MPU (indicating the end of transfer or an error).

Refer to section 6 for a software example of this mode of operation.

15.2 Register Address Summary (SCSI)

Address	R/W	Size	Function
00FE,8000	W	8	Set Controller Address Register
00FE,8000	R	8	Read Auxiliary Register
00FE,8001	R/W	8	SCSI Controller Registers
00FE,7000	R/W	8	SCSI Data Register (pseudo-DMA)

Table 47. SCSI Register Address Summary

15.3 SCSI Port Pinouts

Pin number	Name	Function
Odd pins		Ground
P4- 2	DB0/	Data bit 0
P4- 4	DB1/	Data bit 1
P4- 6	DB2/	Data bit 2
P4- 8	DB3/	Data bit 3
P4-10	DB4/	Data bit 4
P4-12	DB5/	Data bit 5
P4-14	DB6/	Data bit 6
P4-16	DB7/	Data bit 7
P4-18	DBP/	Data parity bit
P4-32	ATN/	Attention
P4-34	Spare	
P4-36	BSY/	SCSI Bus busy
P4-38	ACK/	Transfer acknowledge
P4-40	RST/	Reset
P4-42	MSG/	Message
P4-44	SEL/	Select
P4-46	C/D/	Control/Data
P4-48	REQ/	Transfer request
P4-50	I/O/	Data movement direction

Table 48. SCSI Pinouts

Recommended mating connectors are Ansley P/N 609-5001CE and Molex P/N 15-29-8508.

The terminating resistors, RN14 RN15 and RN16 should be used only if the HK68/M120 is located at an end of the SCSI interface cable.

16. CENTRONICS PORT

This 8-bit parallel port is designed for direct connection to a Centronics compatible (printer) device. Since the handshake lines (STROBE and INIT) are under software control, this interface can be used as a general purpose output port.

16.1 Centronics Port Configuration

P3 Pin	Direction	Signal
P3- 9	Output	DATA8 (D7)
P3- 8	Output	DATA7
P3- 7	Output	DATA6
P3- 6	Output	DATA5
P3- 5	Output	DATA4
P3- 4	Output	DATA3
P3- 3	Output	DATA2
P3- 2	Output	DATA1 (D0)
P3- 1	Output	STROBE/
P3-10	Input	ACK/
P3-11	Input	BUSY
P3-12	Input	PE
P3-13	Input	SELECT
P3-31	Output	INIT/
P3-32	Input	ERROR/
P3-14		Gnd
P3-16		Gnd
P3-(19-30)		Gnd
P3-17		n/c
P3-18		n/c
P3-33		n/c
P3-34		n/c

Table 49. Centronics Pinout (Connector P3)

The falling edge of ACK/ is used to turn on the Centronics interrupt signal going to CIO bit A4. To clear the interrupt signal, read from the interrupt reset location, FE6006.

16.2 Control Port Addresses - Centronics

The Centronics interface logic uses the following physical memory addresses for data and control functions:

Address	Dir	Function
FE6000	W	Data Latch (see below)
FE6000	R	Status Port (see below)
FE6002	W	Turn STROBE on
FE6002	R	Turn STROBE off
FE6004	W	Turn INIT on
FE6004	R	Turn INIT off
FE6006	R	Reset ACK Interrupt

Table 50. Centronics Control Addresses

	FE6000 (Write)	FE6000 (read)
Bit	Data Latch	Status Port
D7	DATA8	x
D6	DATA7	RI/DCD A (see section 14)
D5	DATA6	RI/DCD B (see section 14)
D4	DATA5	ERROR/
D3	DATA4	SELECT
D2	DATA3	PE
D1	DATA2	BUSY
D0	DATA1	ACK/ (Negative true pulse)

Table 51. Centronics Data/Status Addresses

After power-on, the state of the Data Latch is indeterminate; STROBE and INIT will be false. The Data Latch is not changed by a board reset; however, STROBE and INIT will go false.

Follow this procedure when using this port for a Centronics printer:

[1] Wait for the printer BUSY signal to go false.

[2] Write the character to port FE6000.

[3] Turn STROBE on (write to FE6002).

[4] Turn STROBE off (read from FE6002). STROBE must be on for at least one microsecond.

[5] Wait for ACK (poll CIO bit A4, or wait for an interrupt via the CIO). The ACK signal at the Centronics status port (bit D0 of

FE6000) will be just a fleeting pulse.

[6] Reset the ACK interrupt signal by reading from FE6006.

[7] Repeat for the next character.

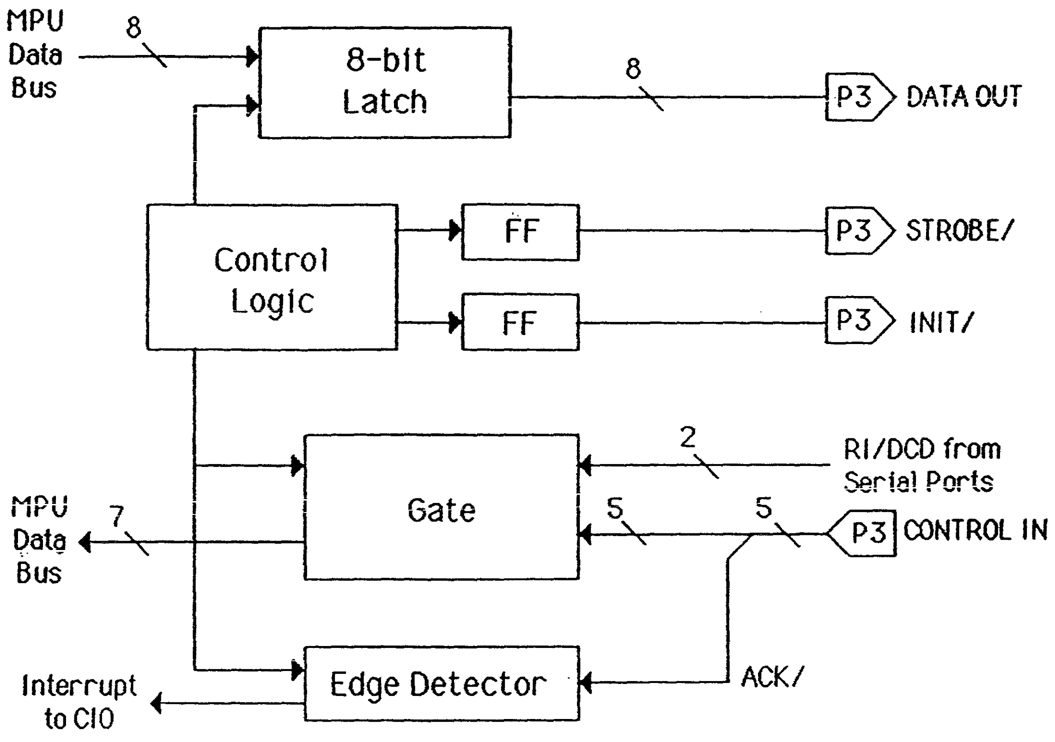


Figure 13. Centronics Interface - Block Diagram

16.3 Centronics Printer Interface Cable

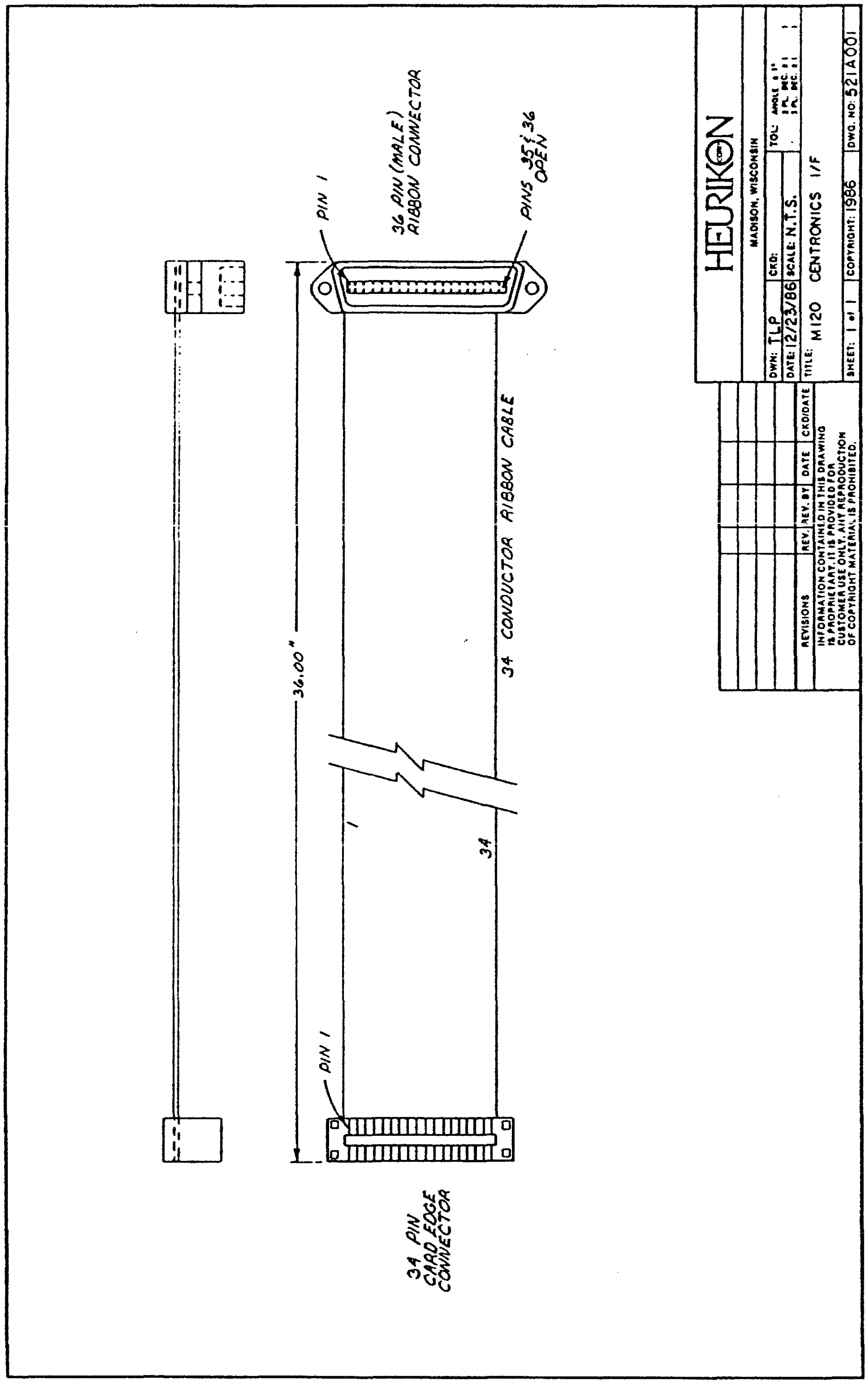


Figure 14. Centronics Printer Interface Cable

17. SBX EXPANSION I/O INTERFACE

The HK68/M120 board has provisions for one 8/16-Bit SBX module. This modules allows users to expand the I/O capabilities of the board by adding appropriate modules.

17.1 SBX Connector Pin Assignments

Pin	Function	Pin	Function	Notes (even pins)
1	+12	2	-12	
3	Gnd	4	Vcc	
5	RESET	6	CLK	10 MHz
7	A3 (MPU)	8	MPS/	(to CIO)
9	A2 (MPU)	10	n/c	
11	A1 (MPU)	12	INT1	(to CIO)
13	WR/	14	INT0	(to CIO)
15	RD/	16	WAIT/	
17	Gnd	18	Vcc	
19	D7	20	CE1/	(see below)
21	D6	22	CE0/	(see below)
23	D5	24	n/c	
25	D4	26	n/c	
27	D3	28	Opt1	to post J10
29	D2	30	Opt0	to post J11
31	D1	32	DACK/	
33	D0	34	DRQT	to CIO port A, D1
35	Gnd	36	Vcc	
37	D14	38	D15	
39	D12	40	D13	
41	D10	42	D11	
43	D8	44	D9	

Table 52. SBX Connector Pinout (P7)

The MPS (module present status) bit allows a program to determine if a module is installed. However, because MPS connects to the CIO, it is possible to use the MPS line as an interrupt or a general purpose pin, although such usage would not match the SBX specification.

Recommended mating connectors for SBX use are Viking P/Ns VSP01VH18A01 (8-bit) and VSP01VH22A01 (16-bit).

17.2 Device Address Summary (SBX)

The functions assigned to each port address are determined by the particular module attached to the port.

16-bit and 8-bit modules are accessed using different addresses as indicated by the tables below.

SBX Pins			Standard Address		Pseudo-DMA Address	
7	9	11	CE0	CE1	CE0	CE1
0	0	0	FE0001	FE1001	FE4001	FE5001
0	0	1	FE0003	FE1003	FE4003	FE5003
0	1	0	FE0005	FE1005	FE4005	FE5005
0	1	1	FE0007	FE1007	FE4007	FE5007
1	0	0	FE0009	FE1009	FE4009	FE5009
1	0	1	FE000B	FE100B	FE400B	FE500B
1	1	0	FE000D	FE100D	FE400D	FE500D
1	1	1	FE000F	FE100F	FE400F	FE500F

Table 53. SBX Address Summary - 8-bit Modules

SBX Pins			Standard Address	Pseudo-DMA Address
7	9	11	CE0 and CE1	CE0 and CE1
0	0	0	FE0000	FE4000
0	0	1	FE0002	FE4002
0	1	0	FE0004	FE4004
0	1	1	FE0006	FE4006
1	0	0	FE0008	FE4008
1	0	1	FE000A	FE400A
1	1	0	FE000C	FE400C
1	1	1	FE000E	FE400E

Table 54. SBX Address Summary - 16-bit Modules

Before accessing an SBX module, the program should test the module present bit (MPS) via CIO port A to be sure a module is plugged in. See section 13 for details.

The "Pseudo-DMA" addresses are used to synchronize the MPU to the SBX DRQT line. Accessing one of these addresses will cause the MPU to wait until the DRQT line is active before the data will be read or written. DACK will be issued when the data transfer occurs.

17.3 SBX Module ID Codes

SBX modules made by Heurikon have a five-bit device ID code. This value can be read in byte mode from address FE000F. Only the lower five bits are significant.

ID	Module
0x01	FD1793 Floppy Controller
0x02	Quad Serial, Z8530 Module
0x03	68881 FPP Module
0x0F	Rockwell MPCC (Serial)
0x10	NCR 5380 SCSI Module

Table 55. Heurikon SBX Module ID Codes

18. REAL-TIME CLOCK (RTC) - Optional Feature

As an option, one pROM can be fitted with a special socket which has a built-in CMOS watch circuit and a lithium battery (Dallas Semiconductor, part number DS1216E).

The module socket is installed in the first HK68/M120 pROM position (U23). It can be plugged into the existing socket (in which case the board profile is wider) or it may replace the standard pROM socket. The following table lists resulting board thickness values, depending on the installation method. The values include a standard pROM thickness.

Configuration	Component Height Above Board	Minimum Board Spacing
RTC module plugged into existing pROM socket:	.75 in.	.85 in. (2 slots)
RTC module soldered into HK68/M120 board:	.55 in.	.65 in. (2 slots)

Table 56. RTC module, physical effects

Only one card slot is required if the board is in the end slot. The RTC logic does not generate interrupts; a CIO timer channel is still used for that purpose. The RTC contents, however, may be used to check for long-term drift of the HK68/M120 system clock, and as an absolute time and date reference after a power failure. Leap year accounting is included. Heurikon can provide complete operating system software support for the RTC module.

The RTC module time resolution is 10 milliseconds. The RTC internal oscillator is accurate to one minute per month, at 25 degrees C.

The clock contents are set or read using a special sequence of ROM read commands, as detailed in the program example, below. The RTC module "monitors" ROM accesses and, if a certain sequence of 64 ROM addresses occur, takes temporary control of the ROM space, allowing data to be read from or written to the module. Writing is done by twiddling an address line, which the module uses as a data input bit. There are never any MPU write cycles directed to the pROM space.

Do not execute the module access instructions out of ROM. The instruction fetch cycles will interfere with the module access sequence.

```
#define WATCHBASE (unsigned char *)0xFA0000   /* ROM, 8-bit mode */
#define WR0_WATCH (unsigned char *)(WATCHBASE+0x8000) /* write 0 */
#define WR1_WATCH (unsigned char *)(WATCHBASE+0xC000) /* write 1 */
#define RD_WATCH  (unsigned char *)(WATCHBASE+1) /* read */

struct rtc_data {         /* D7 D6 D5 D4   D3 D2 D1 D0   range */
    unsigned char dotsec;/* --0.1 sec-- : --0.01 sec- ; 00-99 */
    unsigned char sec;   /* --10 sec--- : --seconds-- ; 00-59 */
    unsigned char min;   /* --10 min--- : --minutes-- ; 00-59 */
    unsigned char hour;  /*  A  0  B Hr : --hours---- ; 00-23 */
    unsigned char day;   /*  0  0  0  1 : --day------ ; 01-07 */
    unsigned char date;  /* --10 date-- : --date----- ; 01-31 */
    unsigned char month; /* --10 month- : --month---- ; 01-12 */
    unsigned char year;  /* --10 year-- : --year----- ; 00-99 */
}; /* "A" =  "0" for 00-23 hour mode, "1" for 01-12 hour mode */
   /* "B" =  MSB of the 10 hours value (if 00-23 hour mode) else
          =  "0" for PM or "1" for AM  (if 01-12 hour mode) */

rtc_wr(data)            /* set the real-time clock */
register unsigned char *data;  /* rtc_data pointer */
{
    register int i, bit;
    static unsigned char key[] = { /* the unlock pattern */
        0xC5, 0x3A, 0xA3, 0x5C, 0xC5, 0x3A, 0xA3, 0x5C };

    if ( data ) {
        rtc_wr(0);       /* send key pattern */
    } else { /* this is the unlock function */
        i = *RD_WATCH; /* reset */
        data = key;
    }
    for( i=0; i<8; data++, i++ )
        for( bit = 1; bit & 0xff; bit <<= 1 )
            ( *data & bit ) ? *WR1_WATCH : *WR0_WATCH ;
}

rtc_rd(data)           /* read the real-time clock */
register unsigned char *data;  /* rtc_data pointer */
{
    register int i, bit;

    rtc_wr(0); /* send key pattern */
    for( i=0; i<8; data++, i++ ) {
        *data = 0;
        for( bit = 1; bit & 0xff; bit <<= 1 )
            *data |= (*RD_WATCH & 1) ? bit : 0 ;
    }
}
```

Figure 15. Real-Time Clock, Example Software

This page has been intentionally left blank.

19. MULTIBUS INTERFACE

(See also "Memory Configuration," section 10, and "Multibus Control," section 11.)

The Multibus (and iLBX interface) consists of P1 and P2 address, data, and control signals. The following tables indicate which signals are used on each portion of the bus.

19.1 P1 (Multibus) Pin Assignments

P1 Pin	Signal	P1 Pin	Signal
1	Gnd	2	Gnd
3	Vcc	4	Vcc
5	Vcc	6	Vcc
7	+12	8	+12
9	(reserved)	10	(reserved)
11	Gnd	12	Gnd
13	BCLK/	14	INIT/
15	BPRN/	16	BPRO/
17	BUSY/	18	BREQ/
19	MRDC/	20	MWTC/
21	IORC/	22	IOWC/
23	XACK/	24	(reserved)
25	LOCK/	26	(reserved)
27	BHEN/	28	ADR10/
29	CBRQ/	30	ADR11/
31	CCLK/	32	ADR12/
33	(reserved)	34	ADR13/
35	INT6/	36	INT7/
37	INT4/	38	INT5/
39	INT2/	40	INT3/
41	INT0/	42	INT1/
43	ADRE/	44	ADRF/
45	ADRC/	46	ADRD/
47	ADRA/	48	ADRB/
49	ADR8/	50	ADR9/
51	ADR6/	52	ADR7/
53	ADR4/	54	ADR5/
55	ADR2/	56	ADR3/
57	ADR0/	58	ADR1/
59	DATE/	60	DATF/
61	DATC/	62	DATD/
63	DATA/	64	DATB/
65	DAT8/	66	DAT9/

Pl Pin	Signal	Pl Pin	Signal
67	DAT6/	68	DAT7/
69	DAT4/	70	DAT5/
71	DAT2/	72	DAT3/
73	DAT0/	74	DAT1/
75	Gnd	76	Gnd
77	(reserved)	78	(reserved)
79	-12	80	-12
81	Vcc	82	Vcc
83	Vcc	84	Vcc
85	Gnd	86	Gnd

Table 57. Pl (Multibus) Connector Pinout

The upper four address lines are located on the P2 connector. Refer to section 11 for signal descriptions.

19.2 P2 (iLBX) Pin Assignments

P2 Pin	Signal	P2 Pin	Signal
1	DB0	2	DB1
3	DB2	4	DB3
5	DB4	6	DB5
7	DB6	8	DB7
9	Gnd	10	DB8
11	DB9	12	DB10
13	DB11	14	DB12
15	DB13	16	DB14
17	DB15	18	Gnd
19	AD0	20	AD1
21	AD2	22	AD3
23	AD4	24	AD5
25	AD6	26	AD7
27	Gnd	28	AD8
29	AD9	30	AD10
31	AD11	32	AD12
33	AD13	34	AD14
35	AD15	36	Gnd
37	AD16	38	AD17
39	AD18	40	AD19
41	AD20	42	AD21
43	AD22	44	AD23
45	Gnd	46	ACK/
47	BHEN	48	R/W
49	ASTB/	50	DSTB/
51	SMRQ/	52	SMACK/
53	LOCK/	54	Gnd
55	ADR16/	56	ADR17/

P2 Pin	Signal	P2 Pin	Signal
57	ADR14/	58	ADR15/
59	(n/c)	60	TPAR/

Table 58. P2 (iLBX) Pin Assignments

ADR14/ through ADR17/ are part of the Multibus interface (carryover from P1). Refer to section 11 for signal descriptions.

19.3 Multibus Compliance Levels

Master:	D16	M24	I16	V0	L
Slave:	D16	M24		V0	L

19.4 Power Requirements

Voltage	Current	Usage
+5	6.5A, max	All logic
+12	1.0A, max	SBX, Reset Timing, RS-232 I/F
-12	0.5A, max	SBX, RS-232 I/F

Table 59. Power Requirements

NOTICE: Power dissipation is about 30 watts. The CIO, DUSCC, RS-232 I/F and some bus logic runs warm to hot. Fan cooling is required if the HK68/M120 board is placed in an enclosure or card rack. Fan cooling is also recommended when using an extender board for more than a few minutes.

19.5 Mechanical Specifications

Length	Width	Height (above board)
12.00 in.	6.75 in.	.490 (Normal)
12.00 in.	6.75 in.	.810 (with SBX module or 4Meg RAM)

Table 60. Mechanical Specifications

Standard board spacing is 0.6 inches. The HK68/M120 is a 10 layer board.

20. SUMMARY INFORMATION

20.1 Software Initialization Summary

This section outlines the steps for initializing the facilities on the HK68/M120 board. Certain steps must be performed in sequence, while others may be rearranged or omitted entirely, depending on your application.

[1] The MPU automatically fetches the reset vector following a system reset and loads the supervisor stack pointer and program counter. The reset vector is in the first 8 bytes of ROM.

[2] Turn off the low ROM mirrors and enable RAM by jumping to the ROM (at base address FA0000). (Reference: section 10.1)

[3] Recall the NV-RAM contents. (Reference: section 10.7)

[4] Determine RAM configuration. (Reference: section 10.3)

[5] Initialize the Memory Control Word to enable on-card RAM. (Reference: section 10.2) This step also sets the Bus Map Bits. (Reference: section 11.4)

[6] Set the Bus Control Bits as desired. (Reference: section 11.2)

[7] Clear on-card RAM to prevent parity errors due to uninitialized memory reads. (Reference: section 9.1)

[8] Load the 68020 Vector Base Register with the location of your exception vector table (usually 0).

[9] Initialize the exception vector table in RAM (at the selected base address.) This step links the various exception and interrupt sources with the appropriate service routines. (Reference: section 5.2)

[10] Initialize the MMU. (Reference: section 8)

[11] Initialize the CIO. (Reference: section 13.6)

[12] Initialize the serial ports. (Reference: section 14.4)

[13] Initialize the SCSI port. (Reference: section 15)

[14] Initialize the Centronics port. (Reference: section 16)

[15] Initialize the User LED port. (Reference: section 12.2)

[16] Read the SBX module ID code (Heurikon only), and initialize the module, as required. (Reference: section 17)

[17] Initialize off-card memory and I/O devices, as necessary.

[18] Read the User Jumpers, if desired. (Reference: section 12.1)

[19] Enable system interrupts, as desired. (Reference: section 5.1)

20.2 On-Card I/O Addresses

This section is a summary of the on-card port addresses. It is intended as a general reference for finding additional information about a particular device. Refer to section 10.5 for a pictorial description of the system memory map. All ports are on the physical address bus.

Hex Address	Type	Device	Reference Section
02xx,xxxx	R/W	iLBX	10.4
01xx,xxxx	R/W	Multibus	10.4
00FF,xxxx	R/W	Bus I/O	11.2
00FE,F003	R	User Jumpers	12.1
00FE,E008-E	W	User LEDs	12.2
00FE,E006	W	BC1	11.2
00FE,E004	W	BC0	11.2
00FE,E002	W	CONVENTION	11.6
00FE,A00x	R/W	DUSCC A,B	14.5
00FE,900x	R/W	CIO	13.4
00FE,800x	R/W	SCSI (normal)	15
00FE,7000	R/W	SCSI (pseudo-DMA)	15, 6
00FE,6006	W	Memory Control Word	10.2, 11.4
00FE,600x	R/W	Centronics	16
00FE,30xx	R/W	NV-RAM Data	10.7
00FE,2000	R	NV-RAM Recall	10.7
00FE,2000	W	NV-RAM Store (tas)	10.7
00FE,500x	R/W	SBX-P7 (pseudo-DMA)	17.2, 6
00FE,400x	R/W	SBX-P7 (pseudo-DMA)	17.2, 6
00FE,100x	R/W	SBX-P7 (normal)	17.2
00FE,000x	R/W	SBX-P7 (normal)	17.2
00FA,xxxx	R	pROM	10.1

Table 61. Address Summary

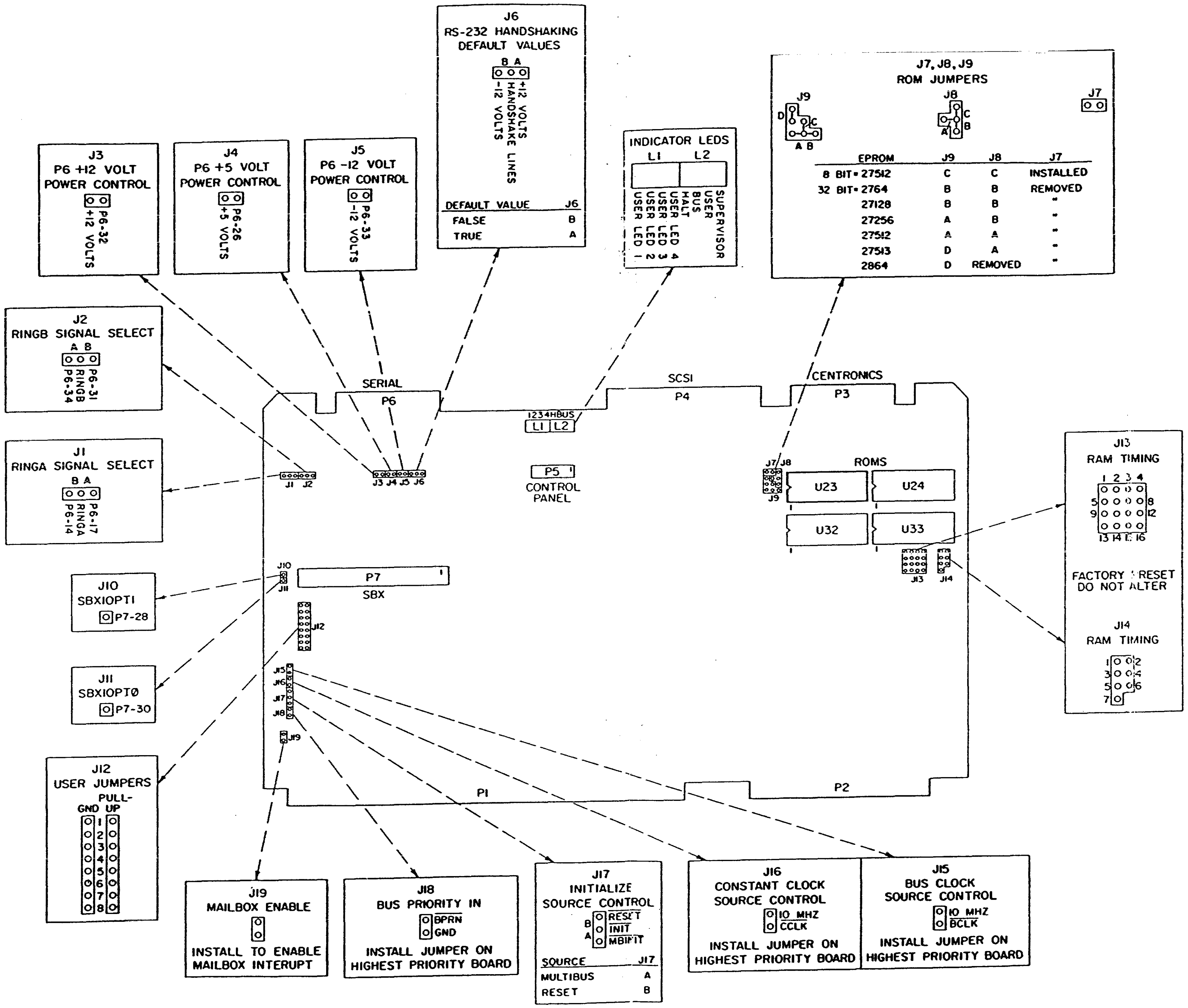
20.3 Hardware Configuration Jumpers

Jumper settings are detailed in the manual section pertaining to the associated device. This section can be used as a cross reference for finding additional information about the jumpers.

Jumper	Function	Reference Section	Standard Configuration
J1	Serial Port A RI, DCD Select	14.6	J1-A (RI)
J2	Serial Port B RI, DCD Select	14.6	J2-A (RI)
J3	P6 +12	14.8	removed
J4	P6 +5	14.8	removed
J5	P6 -12	14.8	removed
J6	RS-232 Defaults	14.6	J6-A (True)
J7	ROM width	10.1	Installed (8-bit)
J8	ROM type	10.1	J8-C (27512)
J9	ROM type	10.1	J9-C (27512)
J10	SBX Option 1	17.1	Open
J11	SBX Option 0	17.1	Open
J12	User Jumpers	12.1	removed
J13	Memory Timing	10.6	(Factory Set)
J14	Memory Timing	10.6	(Factory Set)
J15	B Clock (BCLK)	11.9	Installed
J16	C Clock (CCLK)	11.9	Installed
J17	INIT Control	11.9	J17-A (Input)
J18	BPRN	11.9	Installed
J19	Mailbox Enable	11.8	Removed (disabled)

Table 62. Jumper Summary

The HK68/M120 has been designed to have a minimum number of configuration jumpers. As many options as possible are under software control. The NV-RAM can be used to store board and system configuration information.



DEFAULT VALUE	J6
FALSE	B
TRUE	A

EPROM	J9	J8	J7
8 BIT = 27512	C	C	INSTALLED
32 BIT = 2764	B	B	REMOVED
27128	B	B	"
27256	A	B	"
27512	A	A	"
27513	D	A	"
2864	D	REMOVED	"

SOURCE	J17
MULTIBUS	A
RESET	B

REVISIONS	REV	REV. BY	DATE	CKD/DATE

INFORMATION CONTAINED IN THIS DRAWING IS PROPRIETARY. IT IS PROVIDED FOR CUSTOMER USE ONLY. ANY REPRODUCTION OF COPYRIGHT MATERIAL IS PROHIBITED.

HEURIKON CORP.

MADISON, WISCONSIN

DWN: KES	CKD:	TOL: 2 PL. DEC. ± ( ) ANGLE ± 1/2° 3 PL. DEC. ± ( ) BREAK ALL SHARP CORNERS
DATE: 1/28/87	SCALE: N.T.S.	
TITLE: M120 JUMPER LOCATIONS		
SHEET: 1 of 1	COPYRIGHT: 1987	DWG. NO: 521D002

21. READER COMMENT FORM

We would appreciate any comments you have concerning this guide. Please let us know if you have found any errors or feel that certain sections should be expanded. Thank you.

Name:________________________________ Title:________________

Company:_____________________________ Date:_________________

Address:_____________________________

City:_____________________________ State:________ ZIP:___________

Telephone: (   )-______-___________

Section | Comments

Mail to: Heurikon Corporation
3201 Latham Drive
Madison, WI 53713

HK68/M120 Rev 0

22. APPENDICES

22.1 Additional Technical Literature

Additional information is available on the HK68/M120 peripheral chips, either from Heurikon sales or directly from the chip manufacturers.

Device	Number	Document	Reference Section
MPU	68020	Motorola 68020 Spec	5
FPP	68881	Motorola MC68881 Spec	7
PMMU	68851	Motorola MC68851 Spec	8
CIO	Z8536	Zilog CIO Technical Manual	13
DUSCC	SCN68562	Signetics DUSCC Technical Manual	14
SCSI	WD33C93	WD33C93 Technical Spec	15
RTC	DS1216E	Dallas Semiconductor Clock Module	18

Table 63. Additional Technical Literature

22.2 HK68 Family - Feature Summary

	HK68	HK68	HK68	HK68	HK68	HK68
Feature	M10/ME	V10/VF	VE	V20/V2F	M220/M22F	M120
System Bus	MB-I	VME	VME	VME	MB-II	MB-I
Expansion	iLBX	-	-	VSB	iLBX-II/no	iLBX
MPU (68xxx)	010/000	68010	68000	68020	68020	68020
Speed, MHz	10/12.5	10/12.5	12.5	12.5	16.6	16.6
MMU/PMMU	68451/no	68451/no	-	68851/no	68851/no	68851
FPP	(SBX)	68881	(SBX)	68881	68881	68881
DMAC (68xxx)	450/440	450/440	68440	-	4 Chnl/opt	-
ROM	128-KB	128-KB	256-KB	128-KB	256-KB	256-KB
skts	2 Skts	2 Skts	4 Skts	2 Skts	4 skts	4 skts
width	16 bits	16 bits	16 bits	8 bits	8 or 32	8 or 32
EEpROM	-	-	yes	yes	yes	yes
On-card RAM	1 Meg	4 Meg	4 Meg	4 Meg	4 Meg	4 Meg
Parity	yes	yes	yes	yes	yes	yes
NV-RAM	-	-	-	1-Kbit	1-Kbit	1-Kbit
SCSI	5380/(SBX)	5380/opt	(SBX)	-	33C93/opt	33C93
Control	CIO	CIO	CIO	MFP	CIO	CIO
Serial ports	4/2 (SCC)	2 (SCC)	2 (SCC)	1 (MFP)	2 (SCC)	2 (DUSCC)
Streamer I/O	yes	(SCSI)	-	-	(SCSI)	(SCSI)
Parallel I/O	16/36 bits	-	-	-	-	8 bits
Mailbox	option	std	std	std	Messages	std
SBX	2	-	1	-	1	1
TOD Clock	option	option	option	option	option	option
User LEDs	4	4	4	1	3	4
Status LEDs	5	5 (ext)	5 (ext)	7 (4ext)	4	4
User Jumpers	8	8	8	-	-	8

Table 64. HK68 Family Feature Summary

INDEX

Heurikon Corporation
3201 Latham Drive
Madison, WI 53713
(608) 271-8700