PART NUMBERS
RMS121, 122, 141, 142

Rockwell

BUBBLE MEMORY PRODUCTS
DATA SHEET

SINGLE-BOARD BUBBLE MEMORY SYSTEM

OVERVIEW

The RMS family is a single-board bubble memory system containing the controller and all necessary support circuitry. The module is electrically compatible with the 6500 or 6800 bus. The system is contained on a single 6" by 9.75" module and, through options, may contain one or two one-megabit devices or one or two 256K-bit devices (30,840 to 262,656 bytes). The options are selected by two jumpers on the module.

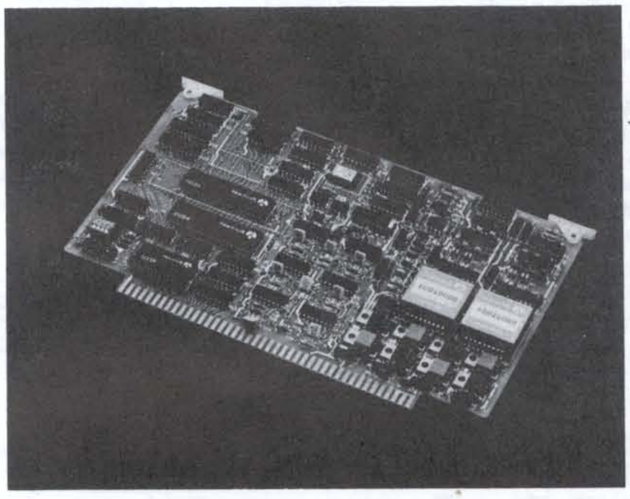
There are four modules in the RMS family:

RMS121 is a 32K-byte single-board bubble memory system
RMS122 is a 64K-byte single-board bubble memory system
RMS141 is a 128K-byte single-board bubble memory system
RMS142 is a 256K-byte single-board bubble memory system

The host system communicates with the RMS via programmed I/O at a rate determined by the number of devices and the location of the redundant loops. For a system with two devices, the average data rate for a block (once it has been accessed) is 22K bytes/second. The maximum data rate (when no redundant loops are encountered) is 25K bytes/second. The access time for a system ranges from 0.84 ms to 22 ms, depending on present block location and the type of bubble device.

The RMS has provision for a file management or utility PROM which, when connected to the system, becomes a part of the memory space for the host computer.

The RMS firmware employs a checksum algorithm for error detection. All data bytes that are to be written in a block are added together, and the resulting eight bits are written into the last location of each block. Upon reading a block of data from the bubble memory, each byte read is also summed. This result is compared to the checksum byte at the end of the block. The error status is loaded into the controller following each block access.



FEATURES

- Board user spectrum
  - One or two RBM256 256K-bit devices
  - One or two RBM411 one-megabit devices
- 32K to 256K bytes
- 100 kHz field rate
- Plug-in compatibility with System 65, AIM 65, 6500 and 6800 systems
- Byte-oriented data transfers
- Extensive command structure
- RCS151 or RCS152 controller
- Programmed data transfer
- Operating system on board
- Checksum error detection
- Data buffering and formatting
- Read/write coordination
- Redundancy control
- All analog circuitry
- Power-fail memory protect circuitry
- 6" × 9.76" pc board size
- 0°C to 50°C operating
- −40°C to +125°C non-operating data retention

REFERENCE DOCUMENTS

The following Rockwell documents are recommended for use with the information in this data sheet:

- SYSTEM 65 User's Manual, Document No. 29650N35
- AIM 65 User's Guide, Document No. 29650N36
- RMS File Management/Utility Program Reference Guide, Document No. 514
- R6500 Hardware Manual Document No. 29650N31

SINGLE-BOARD BUBBLE MEMORY SYSTEM

Document No. BD-11
Rev. 3, July 1980

1800 US 32K.

The Memory Module can be divided into three functional areas: System Interface, controller, and bubble interface. The central unit of the module is the RCS151 or RCS152 NMOS controller. It contains a microprocessor bus interface, an internal microprocessor system (ROM, RAM, CPU and I/O) to decode commands and control the operational sequences and a complete bubble memory control and timing generator.

A 4K-byte slice of the host system main address space selected by a DIP switch allows the user to choose the memory assignment for the RMS. Multiple modules may be used in a system by proper application of the address space assignments.

6500 Bus Structure

6500-based systems are organized around two primary buses. Each bus consists of a set of parallel paths which can be used to transfer binary information between the devices in a system. The first bus, the Address Bus, is used to transfer the address generated by the processor to the address inputs of the memory and peripheral interface devices. The Address Bus consists of 16 lines, allowing the processor to access (Read or Write) up to 65,536 memory words, registers, etc.

The Data Bus consists of an 8-bit bidirectional data path. These lines transfer data from the processor to the selected memory location during a Write operation, and transfer data and instructions from memory into the processor during a Read operation.

The direction of the data transfers is controlled by the Read/Write (R/$\overline{W}$) line on the processor. This line performs the write enable function. As long as the R/$\overline{W}$ line is high (2.4 Vdc), all data transfers will take place from memory to the processor (Read operation). This line will go low only when the processor is going to Write data out to memory. All transitions on this line occur during Phase 1 time of the clock, concurrent with the address lines.

As in most microcomputer systems, the timing of all data transfers is controlled by the system clock. A "Phase 1" clock pulse is the negative portion of B02 during which the address lines change, and a "Phase 2" clock pulse is the positive pulse during which the data is transferred. The timing of the signals on the Address Bus, Data Bus, and R/$\overline{W}$ line is shown in the R6500 Bus Timing section. All signal transitions are specified with respect to the Phase 1 and Phase 2 clock signals. In particular, the address lines and the R/W line will stabilize during Phase 1, and all data transfers will take place during Phase 2.

The specific timing specifications for operating at a 1-MHz clock rate are also given under R6500 Bus Timing. Note that the sequence of operations will be the same for all processors. However, these timing specifications will change for devices which are specified to operate faster than 1.0 MHz. The address is guranteed to be stable 300 ns after the leading edge of Phase 1, and the data must be stable 100 ns before the trailing edge of Phase 2. At 1.0-MHz operation, this allows the memory devices approximately 575 ns to make data available on the data bus. Although there are many factors which determine the actual data and address generated within the system, it is important to keep in mind that the basic timing shown does not change. These figures specify the system bus discipline which applies to <u>all</u> R6500 processors and support chips.

One of the basic system control functions is the system RESET signal. Whether this signal is generated automatically by external power-on circuitry or manually from a switch, the system components must obey a fixed set of rules to assure proper system operation. During power-up and power-down, B RESET (Pin 5) must be held low until the supply voltages have stabilized or been removed. This procedure ensures that the peripheral pins will remain in a known state until the entire system is initialized and the processor is ready to assume control of the output lines.

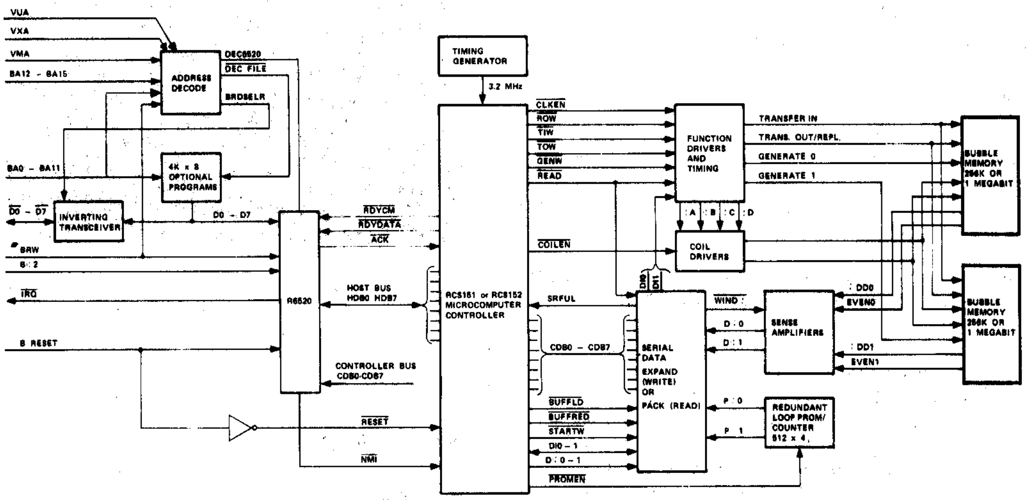


Figure 1. Single Board Bubble Memory System

Interface Signals and Data Flow

The first part of Table 1 describes the System bus signals used by the RMS. The arrows indicate the direction of signal flow. The second part of this table describes the other interfaces internal to the RMS between the "interface" and "controller" functions.

Data Flow

The heart of the RMS controller is an RCS151 or RCS152 one-chip microcomputer and one R6520 Peripheral Interface Adaptor (PIA). The Controller sets up automatic handshanking in the R6520 to communicate commands and control data to the host. Since the controller has no DMA capability the data to be stored in or retrieved from the bubbles is transferred by a byte-for-byte basis. The next two sections describe the Write and Read Data Flow.

Write Data Flow

In a typical Write sequence, the first byte from the host is received by the controller through PORT A of the R6520. Once the write sequence is received by the controller, the host must send the exact number of bytes to fill an entire block of data. The controller signals to the host to send each byte by lowering the $\overline{\text{RDYDATA}}$ line.

The first byte put into the data stream to the bubbles by the controller contains the block address. Every byte after that is sent by the host at approximately 1 byte every 40 microseconds depending when the $\overline{\text{RDYDATA}}$ line goes low. The exact data rate depends on the spacing of the defective loops. The last byte sent by the controller is the checksum byte which is being calculated as the data is being sent.

Read Data Flow

For the Read sequence, the data path operation is exactly opposite to the Write sequence. Data is received from the bubbles serially, shifted into a shift register and received by the controller in parallel. The controller reads the first byte as the block address and strips that off before sending the rest of the data to the host. This data is send through PORT A of the R6520 where the host receives it byte by byte when the $\overline{\text{RDYDATA}}$ goes low. The host must be ready to receive the data at approximately 1 byte every 40 microseconds. The last byte received from the bubbles is also stripped off the data stream and is used by the controller for a checksum. If there is a discrepancy, the checksum error flag is set in the status register to tell the host of such an error.

Checksum

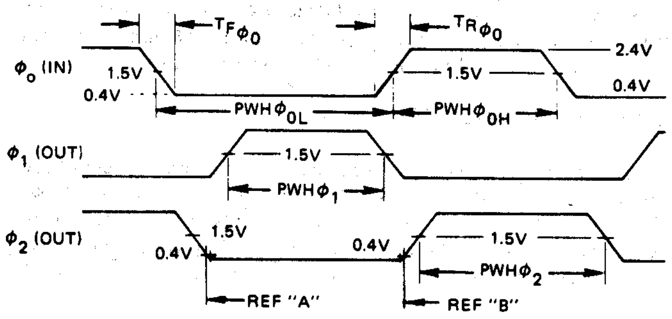
The checksum is used to detect errors in each block and is the last byte of the data stream. It is computed by adding every byte in a block including the first byte (the address field) and using the result of that addition as the last byte of every block. On a write operation, the controller calculates the checksum and writes the result to the bubbles. On a read operation, the controller calculates the checksum and compares the result with the last byte read. Any discrepancy between these two results is flagged in the status register as a checksum error.

Table 1. RMS Interface Signal

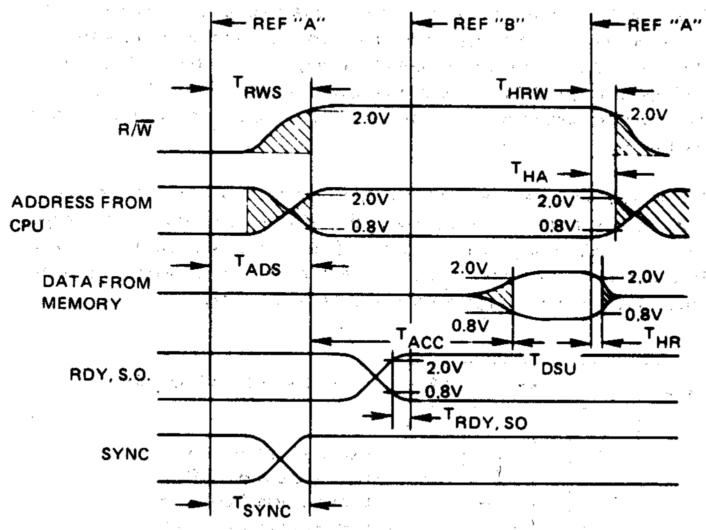
SYSTEM RMS SIGNALS		
AB <0:15> F ←→		These bidirectional address lines inform the RMS when it is being addressed. During data transfers they provide the address of the data byte being transferred. The lines are TTL compatible.
DB <0:7> ←→		These bidirectional data lines are used to transfer data to/from the RMS. The data lines are inverted and are TTL compatible.
BØ2 →		This is the Ø2 clock input from the host system.
BR/W →		The bus read/write line is used to indicate the direction of transfer. During operation this line is controlled by the host system.
VMS, VUA, VXA →		These signals are used in lieu of internally-connected BØ2 for strobing the address in EXORciser and Micro Module systems.
IRQF ←		The System interrupt request line can be used to interrupt the system when a command is completed or an error occurs.
RST →		This line resets the controller, the R6520, and other digital circuitry.
R6520 INTERFACE/CONTROLLER SIGNALS		
Input (from Controller)		
$\overline{\text{ACK}}$	—	a low transition indicates the host computer has put a command word or data word (Write) on port A or that the host has read the data word from port PA (Read).
Output (to Controller)		
$\overline{\text{RDYCM}}$	—	set low to indicate the controller is ready to receive a command word.
$\overline{\text{RDYDATA}}$	—	set low to indicate the controller is ready to receive a data word (Write). Note—a data word must be on port A within 35µs of the low transition of this signal. The controller will read from port A after 35µs even if the host has not loaded new data into port A. This signal is also set low to indicate the controller has output a word to port A (Read). The host must read the word from port A within 35µs.
Data Ports		
PA0-PA7	—	Used to transfer command words to the controller as well as data words to and from the controller.
PB0-PB7	—	Used to transfer status from the controller to the host. The status is available on port B after each command has been executed.

R6500 BUS TIMING

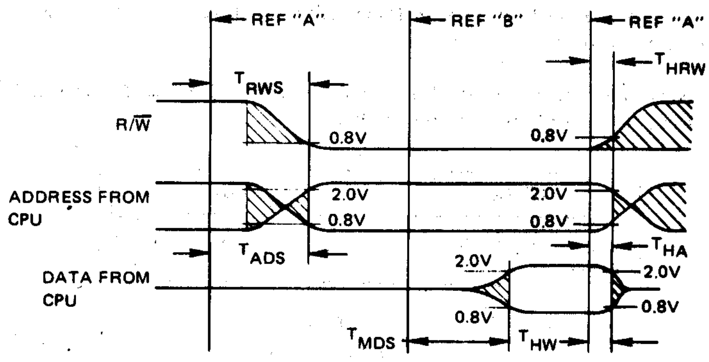
Clock Timing



Timing for Reading Data from Memory or Peripherals



Timing for Writing Data to Memory or Peripherals



Clock Timing

Characteristic	Symbol	Min	Typ	Max	Units
Cycle Time	T_{CYC}	1.0	–	10.0	µs
$\phi_{1(OUT)}$ Pulse Width (measured at 1.5V)	PWHϕ_1	430	–	–	ns
$\phi_{2(OUT)}$ Pulse Width (measured at 1.5V)	PWHϕ_2	430	–	–	ns
$\phi_{1(OUT)}$, $\phi_{2(OUT)}$ Rise, Fall Time (measured at 0.8V to 2.0V) (Load = 130 pF + TTL)	T_R, T_F	–	–	25	ns
Delay Time Between Clocks (measured at 1.5V)	T_D	5	–	–	ns

Read/Write Timing **

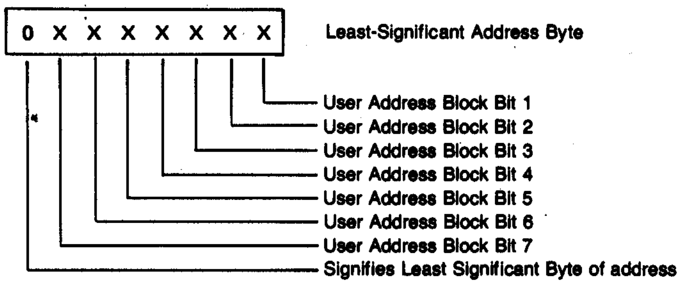
Characteristic	Symbol	Min	Typ	Max	Units
Read/Write Setup Time from R650X	T_{RWS}	–	100	300	ns
Address Setup Time from R650X	T_{ADS}	–	200	300	ns
Memory Read Access Time	T_{ACC}	–	–	575	ns
Data Stability Time Period	T_{DSU}	100	–	–	ns
Data Hold Time	T_H	10	30	–	ns
Enable High Time for DBE Input	T_{EH}	430	–	–	ns
Data Setup Time from R650X	T_{MDS}	–	150	200	ns

**** Load Conditions = 1 TTL Load + 130 pf**

ADDRESSING

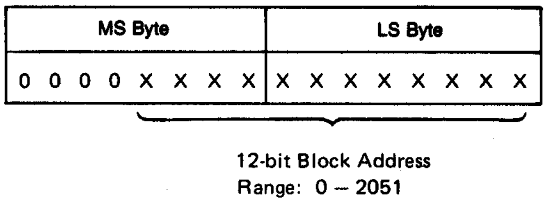
The term "block" refers to one bit in the same relative position from each of the loops (on one or more devices) which are accessed in parallel during a Read or Write. A block of data is sent to the RMS serially from each bubble device, using the defective loop PROM as a shift clock to compact the data. The RMS separates the data stream into bytes which are delivered to the System through Port A of the R6520.

The data stream is partitioned into an address field, a data field and a checksum field. The first byte contains the block address, which is used to determine the current location during power-on reset, and as a check after each subsequent Read operation. The checksum, the last byte of each block, is used to detect data errors. Table 2 gives the number of data bytes for each device.

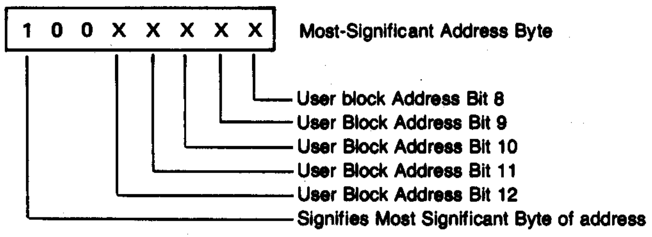


A unique address is stored in bubble memory for each block. The first byte in each block holds that block's address. This address is inserted for the Erase/Address Load and by every Write operation. In the Read mode, it is stripped off before the data is sent to the user.

The number of blocks depends on the device as indicated in Table 2. As shown below, there are 12 bits in the user block address.



The block address actually stored by the RMS (Table 3) differs from the user block in four ways: 1) it is a one-byte address, as opposed to two bytes: 2) each block contains either the MS Byte or the LS Byte, so the controller must read at least two blocks to obtain the block address in restart; 3) Bit 7 signifies whether the address if MS Byte or LS Byte; and 4) since, for a four-micron system, there is an odd number of blocks (1025), the last block is represented as all FFs in the address byte.


1 0 0 X X X X X
Most-Significant Address Byte
User block Address Bit 8
User Block Address Bit 9
User Block Address Bit 10
User Block Address Bit 11
User Block Address Bit 12
Signifies Most Significant Byte of address

If the above byte has the value 11111111, then the address is assumed to be 1025 of the four-micron system.

Upon initial power-up, the controller is put in a reset mode, to establish address positioning. This address is stored in the controller RAM. The controller establishes address positioning by two or three blocks, depending on the previous address positioning. For example, if prior to the initial read the controller is positioned on an MS Byte block, then two blocks are read (one with MS Byte and the following, which is an LS Byte block address). If an LS Byte block does not follow an MS Byte block, a block address error is flaged to the host; the same applies if an MS Byte block does not follow an LS Byte block. The only exception is block number 1024 for a four-micron device.

If the bubble memory is positioned on an LS Byte block address, the controller ignores the address and reads the two following addresses (which are MS Byte followed by LS Byte block).

With the present block address known, the controller converts the one-byte address to a two-byte user block address format, and stores that address in RAM. Before reading or writing to the bubbles, the controller converts the two-byte user address to a one-byte RMS block address, which is used for that bubble function. Since the address is in nonvolatile bubble memory, along with the data, power shut down of the memory does not affect operation. Read operations are non-destructive and also do not affect the block address in memory.

As shown in Table 2, every configuration has a different number of redundant loops. Redundant loop information is stored in a PROM on the RMS board. If a device has fewer redundant loops than the number listed in Table 2, the controller simply does not use these extra "good" loops, and writes zeros into these loops on a Write operation.

FUNCTIONAL OPERATION

The RMS sets up several registers in RAM which are used to calculate and store the block address. A program stored in controller ROM interprets and executes the RMS command set and provides the basic timing for the RMS data transfers. This section describes the registers, command set, and types of errors which are detected. Figure 2 shows a RAM map.

RMS Data Register

The RMS data registers reside in the controller RAM and are initialized by the power-on reset sequence and the Restart command. The user has access to the registers.

REGISTER	FORMAT	CONTENTS
Block Address Reg.	15 8 7 0	Block to be accessed
Secondary Block Address Reg.	15 8 7 0	Base address
# of Blocks Register	7 0	# of blocks to be transferred
Block Counter	7 0	Multi-block Read/Write
Status Register	7 0	Error Conditions & Configuration
Command Register	7 0	Command code
Block size	7 0	# of parallel bubble devices

Block Address Register (0B, 0C)

The value in this register at the start of each Read or Write determines the block to be accessed. After each Read or Write, the block address is incremented to allow the host access to sequential blocks without explicitly changing the block address. The block address register can be loaded by the Read, Write, Load Registers, and Load Registers & Execute commands. The contents of the block address register can be read by the Read Registers command.

Secondary Block Address Register (0D, 0E)

This register is not incremented after each Read or Write and can, therefore, be used as a base address for subsequent operations. The Load Registers and Load Registers & Execute commands can be used to load this register.

of Blocks Register (0A)

This register specifies the number of blocks of data to be transferred during Read and Write operations. The # of blocks register can be loaded by the Read, Write, Load Registers and Load Registers & Execute commands and can be read by the Read Registers command. The value contained in the register is equal to the number of blocks to be transferred minus one; thus from 1 to 256 blocks may be specified.

Status Register (07)

At the completion of each command (after the $\overline{\text{RDYCMD}}$ line goes low), the contents of the status register are available on port B. The assignments for each bit in the status word are shown below. A "1" indicates the error condition exists. Bits 4, 3 and 2 are error conditions. Bits 0, 1 and 7 define the bubble type and capacity.

7	6	5	4	3	2	1	0

1 = 4 micron
0 = 2 micron
	Not Used	Not Used	Invalid Command Code Error	Check-sum Error	Block Address Error	device type
						0 = 256K
						1 = 1M
							# devices
							0 = 1 device
							1 = 2 device

Status Register Bit Assignments

Command Register (06)

The command register is used to store the command code until the operation is completed.

Block Counter (09)

The block counter is used to count blocks during multi-block Read and Write operations.

Block Size Register (08)

The block size register indicates the block size of each controller block. The register is updated during each Restart command and during hardware reset.

Table 2. Data Stream Definition

CONFIGURATION		CONFIG. LINES		LOOP ALLOCATION						AVAILABLE NO. OF BLOCKS
NO. OF DEVICES	DEVICE	PA2	PA1	MAX. REDUNDANT	ADDR.	CHECKSUM	DATA	NOT USED	TOTAL	
1	RBM 256	0	0	10	8	8	256	0	282	1025
2	RBM 256	0	1	36	8	8	512	0	564	1025
1	RBM 411	1	0	44	8	8	512	0	572	2052
2	RBM 411	1	1	98	8	8	1024	6	1144	2052

Table 3. RMS Block Address Format vs. User Block Address Format

BLOCK #		USER BLOCK ADDRESS (HEX.)	RMS BLOCK ADDRESS (HEX.)
DEC.	HEX.		
0	0	0000	80
1	1	0001	00
2	2	0002	80
3	3	0003	02
4	4	0004	80
5	5	0005	04
6	6	0006	80
			. . .
126	7E	007E	80
127	7F	007F	7E
128	80	0080	81
129	81	0081	00
130	82	0082	81
131	83	0083	02
			. .
1022	3FE	03FE	87
1023	3FF	03FF	7E
1024	400	0400	88
1025	401	0401	00
			. . .
2048	800	0800	90
2049	801	0801	00
2050	802	0802	90
2051	803	0803	02
		Special case: 256K four-micron device block 1024	
1024	400	0400	FF

00	**CHKSUM** STORED CHECKSUM	**RDCHKS** READ CHECKSUM	**OUTADD** ADDR TO BUBBLES	**INADD** ADDR FROM BUBBLES	**PNDCMD** PENDING COMMAND	**STRAP** STRAP FOR CONFIGUR.	**COMMAND** CURRENT COMMAND	**STATUS**
08	**BLKSIZ** BLOCK SIZE	**BLKCTR** BLOCK COUNTER	**NBLKS** # OF BLOCKS	**BLKAU** BLOCK ADDR (U)	**BLKAL** BLOCK ADDR (L)	**BLKA2U** 2ND BLOCK ADDR (U)	**BLKA2L** 2ND BLOCK ADDR (L)	**CBLKAU** CURRENT ADDR (U)
10	**CBLKAL** CURRENT ADDR (L)	**ERS FLG** FLAG TO INCREMENT ERSVAL	**ERS VAL** VALUE FOR ERASE ADDR LOAD	**TEMPU**	**TEMPL**	**TEMP 1**	**TEMP 2**	**SAVE X** SAVE X REGISTER
18	**CNTR** WORKING COUNTER	**WRCNTR** # OF BYTES TO WRITE	**SAVE**					

Figure 2. RMS RAM MAP

CONTROLLER COMMANDS

The RMS controller contains an extensive command structure which falls into three primary categories. The first is the operational commands comprising read and write. The second category comprises the initialize, start and verify commands and thirdly a set of test commands is provided to access internal registers within the controller. The command code is always the first byte in the sequence; subsequent bytes contain parameters such as a block address, # of blocks, etc. The R6520 ports must be set up as input or output and the execute command sent.

Table 4. Command Codes

7	6	5	4	3	2	1	0	COMMAND	
0	0	a	a	a	# of Blocks To be Read	Beginning Block Address	X	Read from Bubble Memory	Operational
0	1	a	a	a	# of Blocks To be Written	Beginning Block Address	X	Write to Bubble Memory	
1	1	0	1	0	1	1	0	Read from Last Block Address	
1	1	0	1	0	1	1	1	Write to Last Block Address	
1	1	1	1	1	1	1	1	Execute	
1	1	0	1	0	1	0	0	Cancel—Terminate Multiple Operation	
1	1	0	1	0	0	0	0	Erase/Address Load—Clear and load Block Address	Initialize
1	1	0	1	0	0	0	1	Restart—Read Next Block Address (Same as Hardware Reset)	
1	1	0	1	0	0	1	0	Address Verify—Check RMS Address	
1	1	0	1	0	1	0	1	Load Registers & Execute—Transfer Command Parameters	Test
1	1	0	1	0	0	1	1	Read Registers—Read RMS Status and Address Registers	
1	0	1	2nd Block Address	X	# of Blocks	Block Address	X	Load RMS Registers	
1	1	0	0	b	b	X	X	Transfer RMS Registers—Move 4 Address Registers	
1	1	0	1	1	0	0	0	Read RMS RAM—Test—RAM to System	
1	1	0	1	1	0	1	0	Load RMS RAM—Test—System to RAM	
1	1	0	1	1	1	0	0	Shift—Test—Rotate Bubbles	

Note: X = Don't Care
a, b are parameters

READ AND WRITE COMMANDS

The Read (from Bubble Memory) and the Write (to Bubble Memory) commands are identical in set-up and operation except for the direction of data flow; therefore, these commands will be discussed together, and the notation R/W used to indicate either or both commands.

R/W are multiple byte commands (up to 5 bytes) that allow the user to specify the starting block address, and the number of blocks to be accessed. These parameters may be supplied explicitly as part of the R/W command sequence or the contents of the RMS internal registers may be used. The code for the Read command is "00XXXXXX" and for the Write command is "01XXXXXX" where "XXXXXX" is used to define the R/W parameter options. Some of these options indicate that additional data will be transmitted to the controller via handshake while others specify operations on the RMS internal registers. See Figures 3 and 4 for sequence and timing.

Bit 0 not used

Bit 1

To specify the starting block address as part of the R/W command, bit 1 is set to a logic 1. If the block address is specified, then two bytes of data are added to the R/W command sequence as shown in Figure 3. However, if bit 1 is a logic 0, then the block address is determined by bits 3, 4, and 5.

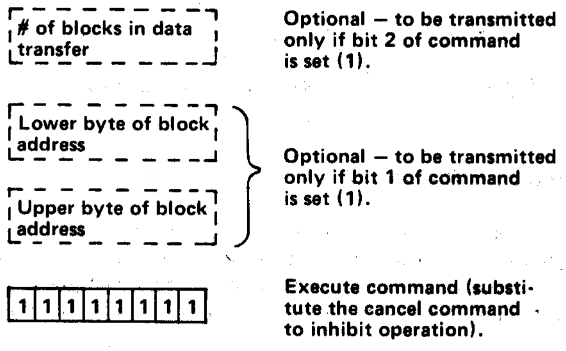
Bit 2

To specify the number of blocks to be accessed as part of the R/W command, bit 2 is set to logic 1. If the number of blocks is specified, then one byte of data is added to the R/W command sequence as shown in Figure 3. However, if bit 2 is a logic 0, the current contents of the # of Blocks Register is used.

Bits 3, 4, and 5

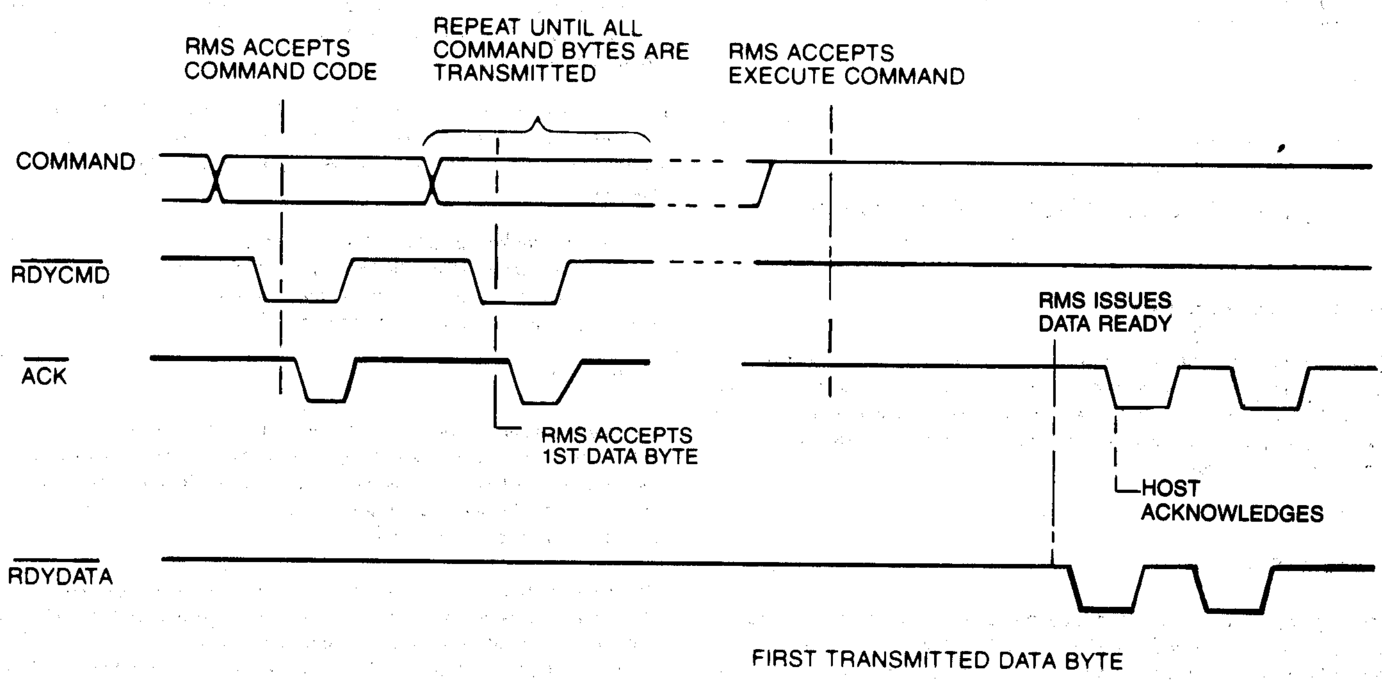
These bits are used to specify data transfers between the Block Address Register and the Secondary Block Address Register. The specific operations performed are shown in Figure 5. These operations are performed prior to the operations controlled by bits 1 and 2.

7	6	5 4 3	2	1	0
0	0: Read 1: Write	Block Address Register & Buffer Address Register Commands	Transmit data length in blocks	Transmit block address (2 bytes)	0



# of blocks in data transfer
Optional — to be transmitted only if bit 2 of command is set (1).
Lower byte of block address
Upper byte of block address
Optional — to be transmitted only if bit 1 of command is set (1).
1 1 1 1 1 1 1 1
Execute command (substitute the cancel command to inhibit operation).

Figure 3. Data Sequence for the Read and Write Commands



RMS ACCEPTS COMMAND CODE
REPEAT UNTIL ALL COMMAND BYTES ARE TRANSMITTED
RMS ACCEPTS EXECUTE COMMAND
COMMAND
RDYCMD
ACK
RMS ACCEPTS 1ST DATA BYTE
RMS ISSUES DATA READY
HOST ACKNOWLEDGES
RDYDATA
FIRST TRANSMITTED DATA BYTE

Figure 4. Timing Sequence for Read, Write, Load Registers, Read/Write Last Block, and Read/Write Controller Register RAM Commands

Bit 5	Bit 4	Bit 3	1st Blk. Addr.	2nd Blk. Addr.
0	0	0	No Change	No Change
0	0	1	No Change	No Change
0	1	0	2nd Blk. Addr.	No Change
0	1	1	2nd Blk. Addr.	No Change
1	0	0	2nd Blk. Addr.	Blk. Addr.
1	0	1	No Change	No Change
1	1	0	No Change	Blk. Addr.
1	1	1	No Change	Blk. Addr.

Figure 5. Address Register Operations

Read from Last Block (D6)

This command allows the user to read from the last accessed block of bubble memory. The last block is determined by decrementing the Block Address Register. Figure 6 shows the data sequence and Figure 4 the timing diagram except that no data bytes are transmitted between the command and execute codes.

1 1 0 1 0 1 1 0	Read from Last Block code

Set up R6520 PA port as Input

NOTE

Write to Last Block command uses same sequence except for command code which is "11010111".

1 1 1 1 1 1 1 1	Execute command. (Substitute the cancel code to inhibit operation.)

Figure 6. Read from Last Block Data Sequence

Write to Last Block (D7)

This command allows the user to write to the last accessed block of bubble memory. Otherwise, operation is the same as for the Read from Last Block command. Except for the command code Figure 6 shows the data sequence and Figure 4 the timing diagram except that no data bytes are transmitted between the command and execute codes.

Execute (FF)

The single-byte Execute command starts the command processing after the parameters, if any, have been passed for a Read, Read from Last Block, Write, Write to Last Block, Restart, Address Verify, Erase/Address Load, Read Registers, or Load Registers & Execute.

Before the Execute command is sent, the R6520 PB port must be set up as inputs and the PA port set up as outputs for the Write, Write to Last Block, and Load Registers & Execute commands and as inputs for the Read, Read from Last Block, Read Registers, Restart, Address Verify, and Erase/Address Load commands.

Cancel (D4)

This single-byte command is used to terminate an interrupted multiple block operation and to cancel a Read, Write, Restart, Address Verify, or Erase/Address Load after the parameters have been passed, but before the Execute command has been given. The Cancel command is also used to clear error conditions prior to entering a new operational command.

Transfer Registers (CX)

This single byte command allows the user to manipulate the four address registers in the controller below. The figures below show the resulting contents of the registers as a function of bits in the command code and the contents of the registers prior to execution of the Transfer Register's command.

Bit 3	Bit 2	2nd Block Address Reg.	Block Address Reg.
0	0	No Change	No Change
0	1	No Change	2nd Block Address
1	0	Block Address	No Change
1	1	Block Address	2nd Block Address

Erase/Address Load (D0)

The Erase/Address Load command is used to clear the bubble memory, load the address field and store one of three different values in each block: 1) all zeroes, 2) all FF's and 3) incrementing values in each block (0's in Block 0, 1's in Block 1, and so on). The block addresses are loaded each block.

This command is generally used when first bringing a system up. To merely turn the system on, the Restart command or a hardware reset is used. An Erase/Address Load operation requires about 6 secs. for 256K device and 21 secs. for megabit devices.

Erase/Address Load is a two byte command. The second byte specifies the value to be input to each block; a value of 00 will cause all zeroes to be input, a value of 80 will cause incrementing values to be input, and a value of 40 will cause FF's to be input.

If an error is detected, operation will terminate and the error condition will be indicated on the eight data lines (R6520 PA port). The Cancel command resets the error flags and terminates the Erase/Address Load execution. Any other command will produce an invalid command error.

Restart (D1)

The restart command causes the controller to read 2 or 3 physical blocks in the bubble memory to establish positioning. The number of physical blocks read is determined by the location of the bubble controller before the restart is issued. After reading the blocks, address positioning is established and the information stored in the block address register.

If the first block read is the LS Byte of the address, that block is ignored and the following two blocks are read to establish positioning. In addition to establishing positioning, a mini address verify is performed by checking alternating blocks of MS Byte and LS Byte. Block address error is flagged to the host if that condition does not exist.

Address Verify (D2)

This command is used to check the RMS block addresses that are stored with each block of data. Since the block address is stored as an extension of the data stream, Address Verify provides a check on the controller integrity. In addition, an interruption of a Write command between erasing the old data and loading the new data (perhaps due to loss of power) can be detected. The execution of this command consists of a series of internally generated RMS Read operations in which the block address fields are examined and compared. Approximately 6 seconds per 256K-bit devices and 21 seconds for megabit devices are required for a complete Address Verify; if an error is found, execution is suspended at that point. Since a series of complete Read operations are performed, two error types may be detected: Block address error or checksum error.

Load Registers and Execute (D5)

This multiple byte command allows the user to load all seven Controller registers. After the controller receives the registers it executes the command that was passed to it in the command register.

`1 1 0 1 0 1 0 1` Load Registers and Execute Command

The registers will be loaded in this order:

Command Register
Status Register
Block Size Register
Block Counter Register
Number of Blocks Register
Block Address (U) Register
Block Address (L) Register
Secondary Block Address (U) Register
Secondary Block Address (L) Register

Figure 7. Data Sequence for the Load Registers & Execute Command (Note: The Cancel Code can be Substituted for any of the Execute Codes to Terminate Operation.)

Read Registers (D3)

This command allows the user to read the controller status and address registers by handshaking this data to the host. One byte is needed in the command sequence: the Read Registers command code. Before receiving the first byte from controller, the PA port must be set up as inputs. The data sequence is shown in Figure 8.

`1 1 0 1 0 0 1 1` Read Register Commands Code

(Set up R6520 PA port as inputs). There is a 27 microsecond delay in the controller before the first $\overline{\text{RDYDATA}}$ is sent. Between each suceeding $\overline{\text{RDYDATA}}$, there is a 60 microsecond time span. This interval gives the host time to set up for accumulating the registers to be sent.

Figure 8. Data Sequence for Read Registers Command

Load Registers (A-B, X)

This multiple byte command (up to 7 bytes) allows the user to load three of the controller registers via a handshake sequence. This timing diagram is similar to that for the Read and Write commands shown in Figure 4. The data sequence is shown in Figure 9.

7	6	5	4	3	2	1	0
1	0	1	Transmit Secondary Block Address	0	Transmit # of Blocks	Transmit Block Address	0

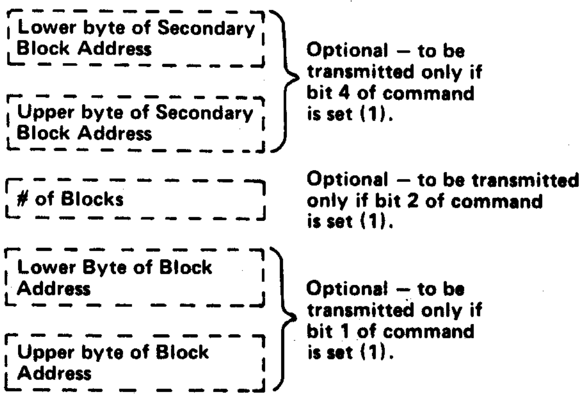

Lower byte of Secondary Block Address
Upper byte of Secondary Block Address
Optional – to be transmitted only if bit 4 of command is set (1).
# of Blocks
Optional – to be transmitted only if bit 2 of command is set (1).
Lower Byte of Block Address
Upper byte of Block Address
Optional – to be transmitted only if bit 1 of command is set (1).

Figure 9. Data Sequence for Load Registers Command

Read Controller Registers RAM (Test Command)

This command allows the user to read the controller register RAM. Figure 3 shows the RAM map. Figure 10 shows the command sequence and Figure 4 shows the timing diagram except that no data bytes are transmitted between the command and execute codes.

`1 1 0 1 1 0 0 0` Read Controller register RAM

(Set up R6520 PA port as outputs)

`1 1 1 1 1 1 1 1` Execute command to signal controller that the R6520 is set up as outputs. (Substitute the Cancel code to inhibit operation.)

There is a 27 microsecond delay in the controller before the first $\overline{\text{RDYDATA}}$ is sent. Between each suceeding $\overline{\text{RDYDATA}}$, there is a 60 microsecond time span. This interval gives the host time to set up for accumulating the registers to be sent.

Figure 10. Data Sequence for Read Controller Register RAM Command

Load Controller Register RAM (Test Command)

This command allows the user to load the first fifteen RAM locations from CHKSUM to BLKA2L as shown in Figure 2.

`1 1 0 1 1 0 1 0` Load controller register RAM

(Set up R6520 PA port as outputs), then send each RAM location by handshaking them to the controller with the $\overline{\text{RDYCMD}}$ line.

Figure 11. Data Sequence for Load Controller Register RAM Command

Shift (Test Command) (DC)

The shift command allows the user to shift the data in bubble memory from 0 to 4095 positions. The current block address is recovered by performing an automatic Restart.

ERROR CONDITIONS

The RMS can detect three types of errors. These errors are recorded in the status register and returned to the system through the data lines after each command is completed ($\overline{\text{RDYCMD}}$ goes high). The error types are:

- Invalid Command Code—Read, Write, Erase/Address Load, Restart, Address Verify, Load Registers and Execute, Read Last Block, Write Last Block, Read Registers, or Load Registers command sequence ending with any code other than "11111111" (Execute command) or "11010100" (Cancel command). This code is also set if a block address is greater than the maximum block address.

- Block Address Error—After each Read operation, the block address from bubble memory is compared to that stored in the RMS (address of the block requested). If the two do not agree, this error flag will be set.

- Checksum Error—After each Read or address verify operation, the checksum result is compared to the stored value. This error flag is set if they do not agree.

- If an error condition exists, the Cancel command must be sent to clear the error.

AVAILABLE FIRMWARE

A 4K-byte optional PROM contains the software necessary to implement a file system and directory similar to the disk-based system resident in System 65 or the cassette-based system resident in the AIM 65. This software package includes the subroutines necessary to communicate commands and control data to the controller. A 4K PROM used to perform utility routines is also available, for both 6500- and 6800-based systems.

These PROMS assume that memory slice "8XXX" is chosen and are identified as

RPR106	System 65 File Program
RPR107	AIM 65 File Program
RPR108	System 65 Utility Program
RPR109	AIM 65 Utility Program
RPR110	EXORciser Utility Program

Tables 5, 6 and 7 show the memory maps for the System 65, AIM 65 and EXORciser systems.

Table 5. System 65 Memory Map

Address	Contents	
0000-00FF	Page 0	
0100-01FF	Stack	On 16K RAM Board in standard System 65
0200-3FFF	RAM	
4000-7FFF	User Memory or I/O	
8000-8FFB	4K PROM	
8FFC	Port A data direction register	
8FFD	Control Register A	
8FFE	Port B data direction register	R6520 on RMS
8FFF	Control Register B	
9000-BFFF	User Memory or I/O	
C000-C3FF	System I/O	
C4000-C7FF	System RAM	See System 65 User's Manual
C800-FFFF	System ROM	for details
00F0-00F7	Utility Memory	
0000-0001	File Management	
C630-C7E9		

Table 6. AIM 65 Memory Map

Address	Contents	
0000-00FF	Page 0	
0100-01FF	Stack	On Board in standard AIM 65
0200-0FFF	RAM	
1000-7FFF	User Memory or I/O	
8000-8FFB	4K PROM	
8FFC	Port A data direction register	
8FFD	Control Register A	
8FFE	Port B data direction register	R6520 on RMS
8FFF	Control Register B	
9000-9FFF	User Memory or I/O	
A000-AFFF	System RAM & I/O	See AIM 65 User's Guide
B000-FFFF	System ROM	for details
0000-002F	Utility Memory	
00DD-00DE	File Management	
0E80-0FFF		

Table 7. EXORciser Memory Map Requirements

Address	Contents	
0000–0025	Variable assignments	
8000–8FFB	4K PROM	
8FFC	Port A data direction register	
8FFD	Control Register A	R6520
8FFE	Port B data direction register	on RMS
8FFE	Control Register B	

Note: Allocation of other memory space is dependent on the equipment and software packages being used.

FILE MANAGEMENT SYSTEM

A set of subroutines (bubble memory I/O drivers) written for System 65 and AIM and resident in the on-board PROM organizes the effective blocks of 128 bytes each (see Table 8) into files with names which are stored in a directory along with the associated block addresses. The directory is stored in a designated location in the bubble memory.

The bubble memory I/O drivers can be used by both the system programs (editor, loader, etc.) and by users programs to read byte, write byte, and close file. The open file routine will search the directory for the named file and determine its starting block address. The first effective block will then be read in for later processing. The second function is read byte. This routine will return next byte from last effect block read. It will also determine when all bytes have been accessed so that the next effective block of data can be read. The write byte routine performs a similar function, but in reverse. Every time a byte is transferred to the subroutine it will be stored in a buffer. As soon as the buffer is full the routine will automatically transfer the effective block of data to the bubble memory. The fourth routine is a close routine. This routine will write the last effective block of data (if any) to the bubble memory and insert the new name in the directory. The data will be kept in two buffers in RAM, one for input and one for output.

From these four basic functions more complex file management routines have been derived such as initializing directory, compressing data, deleting files, renaming files, listing files, etc. In addition, the AIM 65 editor and loader can use these routines for source program and object code manipulations.

The Bubble Memory File Program will operate properly with any of the configurations shown in Table 8. No user input is needed to account for the different block sizes possible. The RMS board must be set up to respond to address $8000.

Table 8. Files

Devices:	RBM256	RBM256	RBM411	RBM411
Number of devices	1	2	1	2
Block Size	32	64	64	128
Number of Blocks	1025	1025	2052	2052
Number of Read/ Writes needed for 128 byte effective block	4	2	2	1
Number of effective* Blocks	256	512	1026	2052
Number of Blocks unused by file management	1	1	0	0

* = The first ten effective blocks are reserved for the directory.

FILE DIRECTORY DEFINITION

The directory is a list of the current files in bubble memory. Each entry requires 16 bytes and specifies the file name, current status, length of file, starting address, and ending address. The format of a directory entry is shown in Figure 12; the U and L indicate the upper and lower bytes of the parameter. The status code indicates the current file status: A "2" indicates an active file and "3" a deleted file. In addition, the end of the directory is marked by a "0" status code. The file name may be any combination of six or fewer alphanumeric characters. If fewer than six characters are used, the file name is padded with ASCII "blanks". The length and address parameters are self explanatory. Note that the last three bytes are not used by the present file program.

Ten effective blocks are allocated for the directory. Each effective block can contain 128/16 = 8 entries, a total of 8 × 10 = 80 entries are possible. However, the last entry must be a marker with the status code equal to "0". Thus 79 user entries are possible.

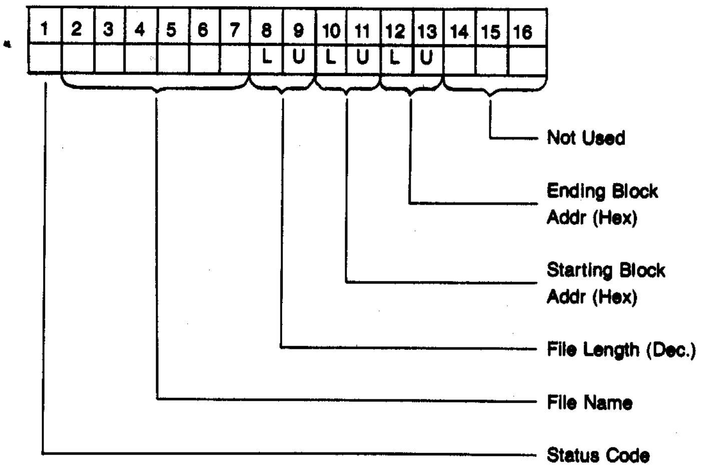


Figure 12. File Directory Entry Definition

UTILITY PROGRAMS

A set of utility programs is available to aid the user in performing input and output operations between the host and the RMS. The routines greatly simplify the user design cycle by providing software control of the R6520 Peripheral Interface Adaptor and the RMS control register.

GENERAL PARAMETERS

DC Power Requirements

	+5V	+12V	−12V
Operating	1.1A	0.4A*	0.1A
Non-Operating	0.9A	0.1A	0.1A

*0.5A for RMS122, 0.6A for RMS142

Temperature Range

Operating (ambient)	0°C to 50°C
Non-Operating	−40°C to +125°C with data retention
Size	9.75" × 6.0" × 0.65" (247.65mm × 152.4mm × 16.5mm)
Relative Humidity	Up to 90% without condensation
Vibration & Shock	That encountered in handling and serving electronic hardware.
Mating Connector	86 pin SAE Type SAM-43D/2-2

Electrical Characteristics

Symbol	Characteristic	Min	Max	Units
V_{OL}	Output Low Voltage (I_{OC} = 16 mA)		0.4	V
I_{OL}	Output Low Current			
	DIO0-7	8		
	Other Signals	16		mA
V_{OH}	Output High Voltage (V_{CC} = 4.75V)	2.4		V
I_{OH}	Output High Current (V_{CC} = 4.75V)	400		μA
V_{IL}	Input Low Voltage		0.8	V
I_{IL}	Input Low Current (V_{CC} = 5.25V)			
	DIO0-7	0.4		
	Other Signals	2.0		mA
V_{IH}	Input High Voltage	2.0		V
I_{IH}	Input High Current (V_{CC} = 5.25V)	50		μA

CONNECTOR P1 PIN ASSIGNMENTS

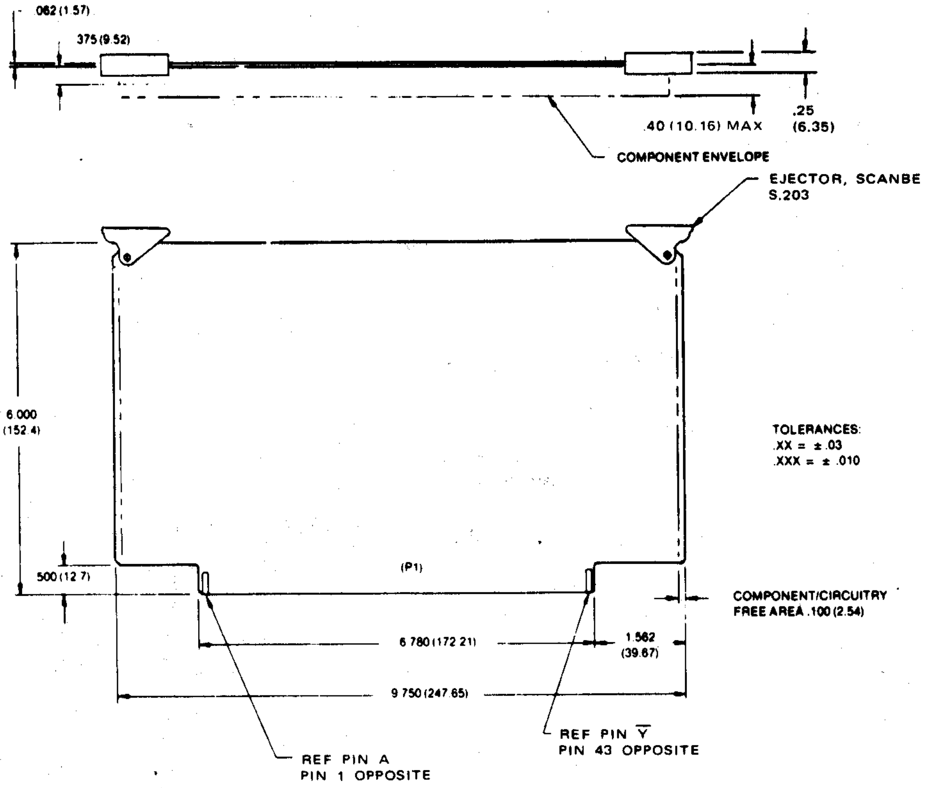
Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	+5V	23	Not Used	A	+5V	$\overline{A}$	Not Used
2	+5V	24	Not Used	B	+5V	$\overline{B}$	Not Used
3	+5V	25	Not Used	C	+5V	$\overline{C}$	Not Used
4	Not Used	26	Not Used	D	$\overline{IRQ}$	D	Not Used
5	B Reset	27	Spare	E	Not Used	$\overline{E}$	Spare
6	BRW	28	Spare	F	VMA	$\overline{F}$	Spare
7	Not Used	29	$\overline{D1}$	H	Not Used	$\overline{H}$	$\overline{D3}$
8	Not Used	30	$\overline{D5}$	J	BØ2	$\overline{J}$	$\overline{D7}$
9	Not Used	31	$\overline{D0}$	K	Not Used	$\overline{K}$	$\overline{D2}$
10	VUA	32	$\overline{D4}$	L	Spare	$\overline{L}$	$\overline{D6}$
11	−12V	33	BA15	M	−12V	$\overline{M}$	BA14
12	Not Used	34	BA12	N	Not Used	$\overline{N}$	BA13
13	Not Used	35	BA11	P	Not Used	$\overline{P}$	BA10
14	Not Used	36	BA8	R	Not Used	$\overline{R}$	BA9
15	Not Used	37	BA7	S	Not Used	$\overline{S}$	BA6
16	+12V	38	BA4	T	+12V	$\overline{T}$	BA5
17	Not Used	39	BA3	U	Not Used	$\overline{U}$	BA2
18	Not Used	40	BA0	V	Not Used	$\overline{V}$	BA1
19	VXA	41	Ground	W	Spare	$\overline{W}$	Ground
20	Not Used	42	Ground	X	Not Used	$\overline{X}$	Ground
21	Not Used	43	Ground	Y	Not Used	$\overline{Y}$	Ground
22	Not Used			Z	Not Used		

OUTLINE DIMENSIONS

The outline dimensions of the RMS circuit card are shown below. This outline is compatible with the Rockwell System 65, and the Motorola EXORciser development systems.

The RMS is intended to mount on 0.75 inch centers.

Card extractors are provided for card removal.



ELECTRONIC DEVICES DIVISION REGIONAL ROCKWELL SALES OFFICES

UNITED STATES

Electronic Devices Division
Rockwell International
3310 Miraloma Avenue
P.O. Box 3669
Anaheim, California 92803
(714) 632-3729
TWX: 910 591-1698

Electronic Devices Division
Rockwell International
P.O. Box 10462
Dallas, Texas 75207
(214) 996-5794/5780
Telex: 73-307 73-2490

Electronic Devices Division
Rockwell International
1601 Civic Center Drive, Suite 203
Santa Clara, California 95050
(408) 984-6070
Telex: 171135 Mission snta

Electronic Devices Division
Rockwell International
10700 West Higgins Rd., Suite 102
Rosemont, Illinois 60018
(312) 297-8862
TWX: 910 233-0179 (RI MED ROSM)

Electronic Devices Division
Rockwell International
5001B Greentree
Executive Campus, Rt. 73
Marlton, N.J. 08053 (609) 596-0090
TWX: 710 940-1377

EUROPE

Electronic Devices Division
Rockwell International GmbH
Fraunhoferstrasse 11
D-8033 Munchen-Martinsried
Germany
(089) 859-9575 Telex: 0521/2650

Electronic Devices Division
Rockwell-Collins
Heathrow House, Bath Rd.
Crawford, Hunslow,
Middlesex, England
(01) 759-9911 Telex: 851-25463

FAR EAST

Electronic Devices Division
Rockwell International Overseas Corp.
Itohpia Hirakawa-cho Bldg.
7-6, 2-chome, Hirakawa-cho
Chiyoda-ku, Tokyo 102, Japan
(03) 265-8806 Telex: J22198

YOUR LOCAL REPRESENTATIVE

5-80

Rockwell International

...where science gets down to business