7.11 RWC-MATH : SCIENTIFIC MATH MODULE

7.11.1 FUNCTIONAL DESCRIPTION

The RWC-MATH Real-World interface module provides the DCE user with complete scientific floating-point math functions for manipulating numbers with 8-digit mantissa and 2-digit exponent.

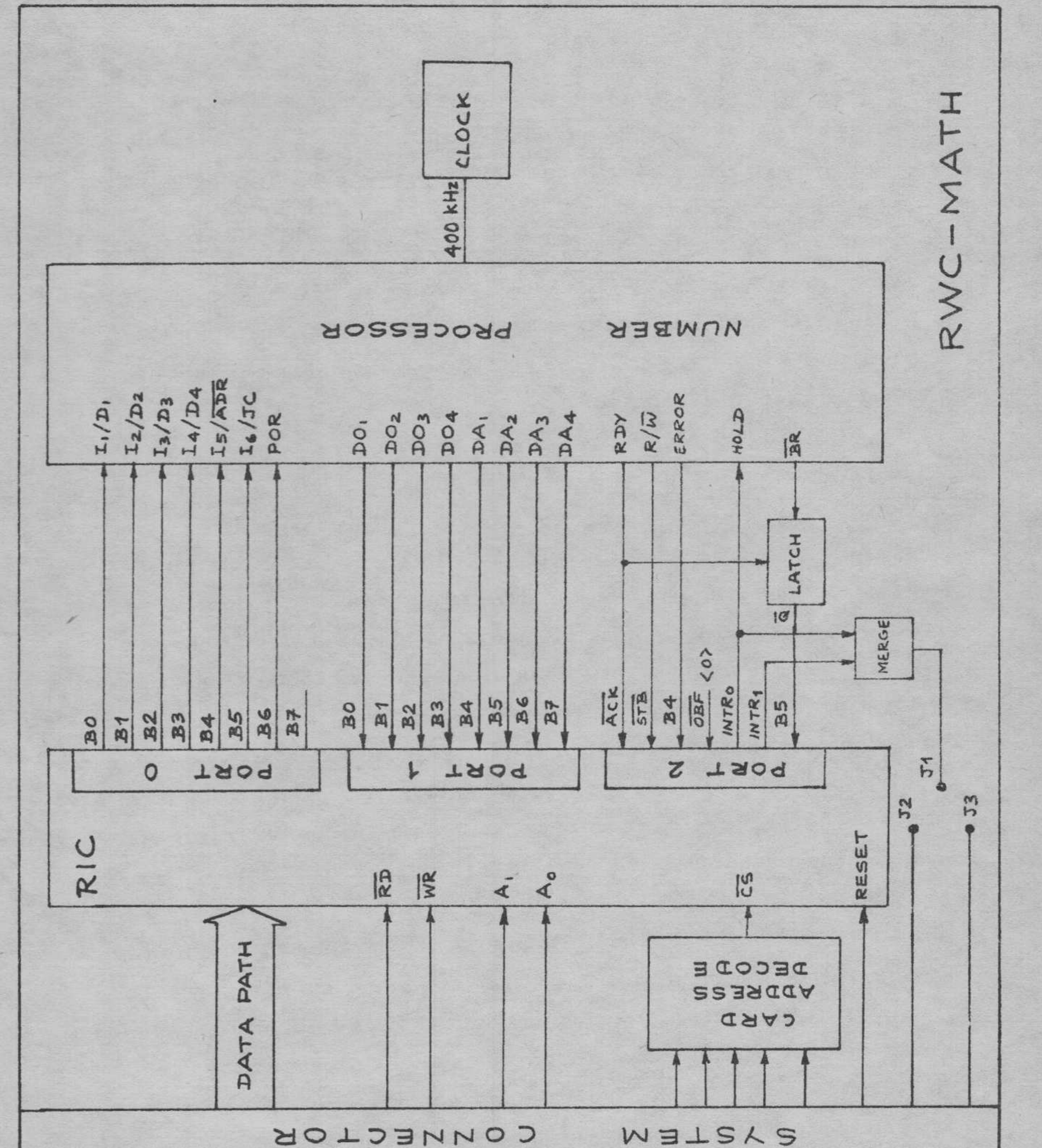
The following functions are provided: +, -, x, ÷, 1/x, $\sqrt{x}$, x^2, 10^x, e^x, y^x, π , ln x, log x, SIN (x), COS (x), TAN (x), $SIN^{-1}(x)$, $COS^{-1}(x)$, $TAN^{-1}(x)$ and radian-degree, degree-radian conversions. The RWC-MATH module functions independently of the DCE processor. Completion of the specified math operations could be detected by software polling the status of the module, or by jumper selectable interrupt generation.

Each module has an identification address defined by a hexadecimal switch, and up to fifteen modules can be directly connected to the DCE-BUS.

7.11.2 FEATURES

- scientific calculations with upto 8-digit mantissa and 2-digit exponent.
- trigonometrical, logarithmic and exponential functions.
- 4 register Reverse Polish Notation push down stack.
- extra memory register.
- error flag generation.
- autonomous operation.
- interrupt generation on completion of operation by jumper selection.
- standard hardware and software interface to the DCE-BUS.
- selectable card address.
- single 100 x 160 mm eurocard format.

7.11.3 FUNCTIONAL BLOCK DIAGRAM



7.11.4 SYSTEM DESIGN PARAMETERS

7.11.4.1 Hardware Configuration

The functional block diagram in Section 7.11.3 illustrates the hardware configuration of the RWC-MATH module. It uses a 57109 MOS/LSI Number-Oriented Microprocessor ("Number Cruncher") for performing all the mathematical calculations.

Data is passed to the RWC-MATH module on a digit by digit basis. The function commands can be interspersed with the data, in the same manner as when pressing keys to enter number and function sequences into a calculator.

The number processor on the module contains 5 internal registers (X, Y, Z, T and M). Each register has 8 mantissa BCD digits with sign, and 2 exponent BCD digits with sign. A complete set of instructions are provided for performing various arithmetic and transfer operations on these registers. Internal calculations are always in the 8-digit scientific notation specified above. Input/Output operations to the number processor may also be done in decimal notation, consisting of upto 8 mantissa BCD digits, a mantissa sign digit and a decimal point position indicator. A special toggle command permits mode changing from decimal to scientific notation, or vice-versa, when reading back the results from the number processor.

Ports 0 and 1 of the RWC-MATH module RIC are configured in handshake output and input modes. The command and data inputs to the number processor are provided by RIC Port 0 in handshake mode. The address and data outputs from the number processor are read into RIC Port 1, also in handshake mode. Bits 4 and 5 of RIC Port 2 are configured as input, for reading the ERROR and $\overline{\text{BR}}$ signals from the number processor.

The two interrupt request signals associated with RIC Ports 0 and 1 are merged into a single interrupt request. This may be connected to the interrupt request lines IN7 or EXINTR on the DCE-BUS via a jumper, for interrupt driven operation of the RWC-MATH module. For polled operation, the two interrupt request signals from RIC Port 2 can be software scanned periodically.

7. 11. 4. 2 Programming Specifications

The RWC-MATH module is addressed via the standard DCE-BUS interface. Programming specifications for driving the DCE-BUS are given in Section 4. 1.

RIC Initialization

The RIC on the RWC-MATH module should be initialized by writing the control word 0AEH to the RIC Command Register. This configures RIC Port 0 as handshaking output and RIC Port 1 as handshaking input. Unused Bits 4 and 5 of Port 2 are configured as input.

RIC Device Addresses

The RIC on the RWC-MATH module has 3 data ports and a command register. Different modes of communication between the RIC Ports and the DCE-BUS data path are established depending on the Device Address received by the RWC-MATH module from the DCE-BUS. Table 7. 11. 1 shows the Device Addresses needed for different communication modes.

Format of Data

The command and data inputs to the number processor are sent from the DCE-BUS via RIC Port 0. The address and data outputs from the number processor are read into the DCE-BUS via RIC Port 1. The free bits of RIC Port 2 are used for reading the error flag and the branch sense input from the number processor.

Signal allocation for RIC Ports 0, 1 and 2 are as follows:

Port 0 :	b0 - b3 =	instruction bits, mantissa digit count or digit data (I1/D1 to I4/D4). b3 is the most significant bit.
	b4 =	instruction bit, or data ready indicator (I5/$\overline{\text{ADR}}$).
	b5 =	most significant instruction bit, or jump condition indicator (I6/JC).
	b6 =	power on reset (POR)
	b7 =	not used
Port 1 :	b0 - b3 =	BCD digit output (DO1 to DO4) b3 is the most significant bit.
	b4 - b7 =	digit address bits (DA1 to DA4). b7 is the most significant bit.
Port 2 :	b0 =	Port 1 service interrupt request
	b3 =	Port 0 service interrupt request
	b4 =	error flag (ERROR)
	b5 =	latched branch sense input ($\overline{\text{BR}}$). Normally not used.
	b1, b2, b6, b7 =	handshake control signals for Ports 0 and 1.

DEVICE ADDRESS (HEX)	$\overline{RD}$	$\overline{WR}$	OPERATION
Y0	0	1	Invalid Operation
Y1	0	1	RIC Port 1 → DCE Data Bus
Y2	0	1	RIC Port 2 → DCE Data Bus
Y3	0	1	Illegal Condition
Y0	1	0	DCE Data Bus → RIC Port 0
Y1	1	0	Invalid Operation
Y2	1	0	Invalid Operation
Y3	1	0	DCE Data Bus → RIC Command Register
ZX	X	X	RIC Data Bus in 3-state

Notes:

1. Y is the card address select switch setting in hex (1 to F).
2. Z is any number other than Y.
3. X means don't care.
4. Bits 2 and 3 in Device Addresses are don't care states.
5. RDRWC and WRRWC software routines provide the $\overline{RD}$ and $\overline{WR}$ signals accordingly.

Table 7. 11. 1 : Device Address Table for RWC-MATH

Interpretation of Data

All data and instructions are input to the number processor via RIC Port 0. Numbers and instructions can be interspersed in any suitable sequence to produce the desired mathemetical result. Numbers are input as BCD digits, with Bits 4 to 7 equal to zero. Instructions are input as 6 bit words, with Bits 6 and 7 equal to zero. Table 7. 11. 2 shows the relevant instruction codes for the number processor.

The number processor is reset by the power on reset signal POR. This should be set high during initialization, and then maintained low.

All result output from the number processor are read back via RIC Port 1. Bits 4 to 7 of the result give a 4-bit address associated with each output digit. They may be used if necessary for checking that none of the output digits arriving synchronously are lost. These 4 bits are set to zero after each output sequence.

Result data is always read back from the X register of the number processor. Each digit is read as a BCD number via Bits 0 to 3 of RIC Port 1. The data format can be decimal or scientific notation, as shown in Tables 7. 11. 3 and 7. 11. 4 respectively.

RIC Port 2 provides the necessary control signals for maintaining correct protocol during the input and output operations with the number processor. After the reset sequence, or after the execution of a command, readiness of the number processor to accept the next input is detected via the Port 0 service request signal at Port 2 Bit 3. This bit may be made to generate an interrupt, or it may be polled.

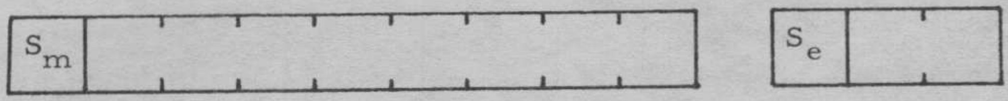
After an output request command has been sent to the number processor, it will send the contents of the X register to Port 1, one digit at a time. The availability of this data can be detected via the Port 1 service request signal at Port 2 Bit 0. This bit can be made to generate an interrupt, or it can be polled.

The number processor sets the error flag upon detection of an arithmetic or output error. The flag is made available at Port 2 Bit 4, for checking the validity of the output from the number processor.

It is good practice to read the control word from RIC Port 2 periodically and store it at start of the result buffer, or some other suitable location. This enables the main program to determine the execution status of an initiated command sequence, specially during interrupt driven operation of the number processor.

Special Considerations

The number processor on the RWC-MATH module has 4 work registers (X, Y, Z, T) forming a stack, and one memory register (M). Each of these registers has 8 mantissa digits with sign, and 2 exponent digits with sign:



A comprehensive set of commands are provided for performing mathematical functions and maipulations on the contents of these 5 registers. Input/output commands operate on the X register.

Command Codes

Table 7. 11. 2 gives the complete set of commands for the number processor. Command strings can be assembled by interspersing these commands with BCD data digits, for performing any desired calculation. Command strings can be any number of characters in length, and must be terminated with TERM. Some examples of command strings are given below :

a) 2, DP, 5, 8, 9, EE, 2, CS, SQRT, TERM

= square root of 2.589×10^{-2}

b) 6, DP, 3, 5, 4, 2, CS, EE, 3, CS, EN, 3, 1, 8, YTX, TERM

= -6.3542×10^{-3} X 318

c) 2, EX, EN, 2, SIN, YPX, TERM

= $e^{2} + \sin 2°$

In all the above examples, the answer at the end of the computation will be in the X register. It can then be read via the OUTT command.

When building command strings, it is necessary to consider the effect of each command on the register stack, to ensure correct computational sequence. In the above examples, the command EN is used where necessary to end the digit entry and push the stack, so that the entered argument is pushed into register Y, and register X is available for the next entry.

Table 7.11.2 explains the results of each command, including the effects on the stack. Some commands are two bytes in length. These should be generated by using the 'DW' assembler command. Since the DW command generates two bytes with the low order byte first, the values for double byte commands given in Table 7.11.2 are inverted. For example the value for ISIN is given as 2420H, so that DW ISIN can generate the correct two byte code 2024H.

Some of the commands are explained below :

RAD	Convert X from degrees to radians.
DEG	Convert X from radians to degrees.
DP	Decimal point position. Indicates that the following digits will be mantissa fraction.
CS	Change sign of exponent or mantissa. Changes the sign of the mantissa if present before EE; changes the sign of the exponent if present after EE.

EN	Terminates digit entry and pushes the stack. The entered argument will be in X and Y registers.
TOGM	Changes mode from decimal notation to scientific notation, or vice-versa, depending on the current mode. The mode is set to decimal notation by the master clear (MCLR). The mode change affects only the OUTT instruction. Internal calculations are always in 8-digit scientific notation.
SMDC	Mantissa digit count (MDC) is set to the contents of the following byte (1 to 8).
MCLR	Master clear. Clears all internal registers and memory to zero, initializes I/O control signals from the number processor, sets mantissa digit count (MDC) = 8, and sets output mode = decimal notation.
ECLR	Error flag reset. Normally not used.
POPS	Pops the X, Y, Z, T register stack. Contents of the X register are lost.
ROLL	Similar to POPS, except that none of the register contents are lost.
PI	Sets $X = \pi$. Stack is not pushed.
LSH	X mantissa is left shifted while leaving the decimal point in the same position. Former most significant digit is saved in a separate link digit. This link digit is normally zero except after a left shift. Zero is shifted into the least significant digit.
RSH	X mantissa is right shifted while leaving the decimal point in the same position. The link digit, which is normally zero except after a LSH, is shifted into the most significant digit. The least significant digit is lost.
NOOP	Null command (NOP), used for terminating digit entry. Does not push the stack.
TERM	Table terminator, used by system software to detect the end of the command string.

The trigonometrical functions SIN, COS and TAN assume that the contents of the X register is in degrees.

Table 7.11.2 : Equates for RWC-MATH Command Codes

```
;         NOTE THAT THE DOUBLE BYTE COMMANDS SHOULD
;         BE GENERATED USING THE 'DW' ASSEMBLER
;         COMMAND.
;
;
YPX       EQU       39H       ;  X<-Y+X, Y<-Z, Z<-T, T<-0
YMX       EQU       3AH       ;  X<-Y-X, Y<-Z, Z<-T, T<-0
YTX       EQU       3BH       ;  X<-Y*X, Y<-Z, Z<-T, T<-0
YDX       EQU       3CH       ;  X<-Y/X, Y<-Z, Z<-T, T<-0
YTOX      EQU       38H       ;  X<-Y**X, Y<-Z, Z<-T, T<-0
;
MPX       EQU       3920H     ;  M<-M+X
MMX       EQU       3A20H     ;  M<-M-X
MTX       EQU       3B20H     ;  M<-M*X
MDX       EQU       3C20H     ;  M<-M/X
;
ODX       EQU       37H       ;  X<-1.0/X
SQRT      EQU       34H       ;  X<-X**0.5
SQ        EQU       33H       ;  X<-X**2
TENX      EQU       32H       ;  X<-10.0**X
EX        EQU       31H       ;  X<-E**X
LN        EQU       35H       ;  X<-LN  X
LOG       EQU       36H       ;  X<-LOG X
SIN       EQU       24H       ;  X<- SIN X
COS       EQU       25H       ;  X<- COS X
TAN       EQU       26H       ;  X<-TAN X
ISIN      EQU       2420H     ;  X<-ARCSIN X
ICOS      EQU       2520H     ;  X<-ARCCOS X
ITAN      EQU       2620H     ;  X<-ARCTAN X
RAD       EQU       2DH       ;  X<-RAD X
DEG       EQU       2CH       ;  X<-DEG X
;
;
DP        EQU       0AH       ;  DECIMAL POINT POSITION
EE        EQU       0BH       ;  FOLLOWING DIGITS ARE EXPONENT
CS        EQU       0CH       ;  CHANGE MANTISSA OR EXPONENT SIGN
EN        EQU       21H       ;  T<-Z, Z<-Y, Y<-X ; END DIGIT ENTRY.
TOGM      EQU       22H       ;  TOGGLE MODE (SCIENTIFIC <-> FP)
SMDC      EQU       18H       ;  SET MANTISSA DIGIT COUNT
                              ;  FOLLOWING BYTE MUST CONTAIN
                              ;     REQUIRED MANTISSA DIGIT COUNT.
MCLR      EQU       2FH       ;  MASTER CLEAR:  X, Y, Z, T, M<-0;
                              ;                    ERR FLAG<-0; MDC<-8;
                              ;                    MODE<-FLOATING POINT.
ECLR      EQU       2BH       ;  ERR FLAG<-0
POPS      EQU       2EH       ;  X<-Y, Y<-Z, Z<-T, T<-0
ROLL      EQU       23H       ;  X<-Y, Y<-Z, Z<-T, T<-X
XEY       EQU       30H       ;  X<->Y
XEM       EQU       1BH       ;  X<->M
MS        EQU       1CH       ;  M<-X
MR        EQU       1DH       ;  X<-M
PI        EQU       0DH       ;                     X<-3.1415927
LSH       EQU       1EH       ;  LEFT SHIFT MANTISSA OF X
RSH       EQU       1FH       ;  RIGHT SHIFT MANTISSA OF X
NOOP      EQU       3FH       ;  NULL COMMAND (NOP), TERMINATES
                              ;       DIGIT ENTRY.
OUTT      EQU       0016H     ;  MULTIDIGIT OUTPUT FROM X
TERM      EQU       0FFH      ;  TABLE TERMINATOR
```

Number Entry

When a digit (0 - 9), decimal point (DP), or π (PI) is entered into the number processor via RIC Port 0, the stack is first pushed and the X register cleared :

$X \leftarrow 0,\ Y \leftarrow X,\ Z \leftarrow Y,\ T \leftarrow Z$

This process is referred to as "initiation of number entry". Following this, that digit and the following digits are entered into the X mantissa. Subsequent entry of digits, DP, EE or CS commands do not cause initiation of number entry. Digits following the eighth mantissa digit are ignored. This number entry mode is terminated by any command except 0-9, DP, EE, CS, or PI.

Termination of number entry results in the following:

- the entered number is normalized by adjusting the exponent and decimal point position, so that the decimal point is to the right of the first mantissa digit.
- the next digit, DP, or PI entered will cause initiation of number entry, as already described.

Arguments can be entered as decimal, floating point, or a combination of both.

Result Output

There are two possible modes of operation for the OUTT instruction, which reads back the results of the execution of a command string from the X register :

- Decimal notation, which transfers mantissa digits, a mantissa sign digit, and a decimal point position digit.

- Scientific notation, which transfers mantissa digits, 2 exponent digits, and a digit containing mantissa and exponent sign bits. The decimal point is always to the right of the first mantissa digit.

Initially the number processor output is in the decimal notation (after MCLR command). The TOGM command toggles from any current mode to the opposite mode. The number of mantissa digits output is equal to the mantissa digit count (MDC). The MDC is initially 8 and can be set to any value from 1 to 8 using the SMDC command. Output digits are in BCD, and are read via RIC Port 1 Bits 0 to 3.

After execution of the OUTT command, the number processor periodically strobes each data digit into RIC Port 1. This data is output synchronously, and must be read from RIC Port 1, as soon as its service request signal (RIC Port 2 Bit 0) becomes active. Each data digit remains at RIC Port 1 for approximately 125μs after Port 1 service request signal becomes active. This service request signal can be made to generate an interrupt via jumper selection, or it can be software scanned. In either case, it is essential to read RIC Port 1 within the above time limit. The general purpose Real-World read subroutine RDRWC is too slow for reading this data. A simplified read routine is illustrated in the programming example in Section 7. 11. 5. DCE CPU interrupts should be disabled during the read sequence.

Table 7. 11. 3 shows the format of output data from the number processor, in decimal notation, read via RIC Port 1. Bits 0 to 3 give the mantissa sign indication word (0 = positive, 8 = negative), decimal point indication word (11 to 12 - MDC) and the individual mantissa digits (each 0 to 9). DP POS takes values from B (hex) to 11-MDC (= 3 normally). If for example, DP POS = 9, the decimal point is located to the right of the third most significant mantissa digit.

Table 7. 11. 3 : Output Data Format for Decimal Mode

Address Bits b7 - b4	DP Position (hex)	BCD Data Digits b3 b2 b1 b0
2		S_m 0 0 0
3		DP POS
4	B	Most significant mantissa digit
5	A	.
6	9	.
.	.	.
.	.	.
MDC + 3	12-MDC	Least significant mantissa digit

Table 7. 11. 4 : Output Data Format for Scientific Mode

Address Bits b7 - b4	BCD Data Digits b3 b2 b1 b0
0	Most significant exponent digit
1	Least significant exponent digit
2	S_m 0 0 S_e
3	not used
4	Most significant mantissa digit (decimal point follows this digit)
.	.
.	.
MDC + 3	Least significant mantissa digit

Notes:

MDC	=	Mantissa digit count, set by SMDC command; = 8 initially.
S_m	=	Sign of mantissa : 0 = positive, 1 = negative.
S_e	=	Sign of exponent : 0 = positive, 1 = negative (S_e = 0 in decimal mode).
DP POS	=	Decimal point position indicator for Decimal mode. Takes values from 11 down to 12 - MDC, which indicates a digit as given by the DP column in Table 7. 11. 3. The decimal point is located to the right of this digit.

The folowing sequence is an example of an output data string read in decimal mode (MDC = 8). The values indicate the contents of RIC Port 1 Bits 0 - 3, for each output character received from the number processor (MDC = 8) :

```
8 B 0 2 5 7 8 7 4 3
( = - 0.2578743)
```

Table 7.11.4 shows the format of output data from the number processor, in scientific notation, read via RIC Port 1. Bits 0 to 3 give the two exponent digits, mantissa and exponent sign indication word, and the individual mantissa digits. The decimal point is always located to the right of the most significant mantissa digit. The mantissa and exponent sign indication word can have the following values :

0 (hex)	: mantissa + ;	exponent +
1	: mantissa + ;	exponent -
8	: mantissa - ;	exponent +
9	: mantissa - ;	exponent -

The following sequence is an example of an output data string read in scientific mode (MDC = 8). It is the output from the same internal result as in the last example.

0 1 9 - 2 5 7 8 7 4 3 4

$(=-2.5787434 \times 10^{-1})$

<u>Error Conditions</u>

Several commands to the number processor have conditions which will cause an error. When such an error occurs, the number processor sets an Error Flag, which can be read via RIC Port 2 Bit 4.

The error conditions are as follows :

1) LN X, LOG X, when $X \leqslant 0$
2) any result $< 10^{-99}$ or $\geqslant 10^{100}$
3) TAN 90°, 270°, 450°, etc.
4) SIN X, COS X, TAN X, when $|X| \geqslant 9000°$
5) ISIN X, ICOS X, when $|X| > 1$ or $|X| \leqslant 10^{-50}$
6) SQRT X, when $X < 0$
7) YDX, MDX, ODX, when $X = 0$
8) In decimal mode OUTT instructions, if the number of mantissa digits to the left of decimal point is > mantissa digit count.

The number processor sets the error flag only during the command which gave rise to the error condition. Therefore it is necessary to scan the error flag from RIC Port 2 Bit 4, while sending commands to the number processor, to ensure the detection of intermediate error conditions. One way to realize such a scheme is to store the status word read from RIC Port 2, at the beginning of the result buffer, and to set Bit 4 in it if the error flag becomes active at any time during the execution of the command string. Once the complete result has been stored in the result buffer by the appropriate subroutine, the main program can test Bit 4 of this status word to check the validity of the contents of the result buffer, before processing the result. If the error bit in this status word is set, the main program must clear it before initiating input of the next command string. An example of such a scheme is found within subroutine MATO in the test program given in Section 7.11.5.

Command Execution Times

Table 7.11.5 shows the execution times of each command. These times are shown in microcycles. One microcycle has a value equal to 10 μs. The execution of a single command by the number processor involves thousands of such microcycles.

Command Mnemonic	Execution Time in Microcycles	
	average	worst-case
0 - 9		238
DP		152
EE		151
CS		166
PI		1312
SMDC		163
XEM		812
MS		839
MR		1385
LSH		168
RSH		173
EN		552
TOGM		157
ROLL		905
ECLR		163
POPS		448
MCLR		734
XEY		652
NOOP		122
OUTT		583

Command Mnemonic	Execution Time in Microcycles	
	average	worst-case
SIN	56200	95900
COS	56200	95900
TAN	35000	97600
ISIN	54000	93900
ICOS	54000	93900
ITAN	30200	92900
LN	24800	92000
LOG	30700	92600
EX	30800	93900
TENX	27400	96500
YPX	2200	6600
YMX	2200	6600
MPX	1700	5000
MMX	1700	5000
YTX	3200	22700
MTX	2700	21400
YDX	7800	22300
MDX	7300	21100
ODX	4500	22800
YTOX	55400	95500
SQRT	7000	30200
SQ	3000	21900
RAD	9600	41700
DEG	9600	41700

Table 7. 11. 5 : Number Processor Command Execution Times

Note: 1 microcycle has a value of 10 μsec.

RWC-MATH RIC / DCE-BUS Protocol

Initialization

First initialize the RWC-MATH module RIC by writing control word 0AEH to its Command Register. This will configure the three RIC ports as required.

Initialize the number processor by setting RIC Port 0 Bit 6 to logic one, and then to logic zero. This will reset the number processor.

Enable the two service request interrupt signals associated with RIC Ports 0 and 1, by setting RIC Port 2 Bits 2 and 6 to logic one.

Send out a NOOP command to the number processor via RIC Port 0, and then perform a dummy read operation to RIC Port 1, to clear the internal logic of the number processor. Clear the status byte at the start of the result buffer, if present. Set up a dummy command string containing several NOOP commands and a MCLR command. Send this command string to the number processor, and then read back the dummy output data from it as explained below. The number processor is now ready to perform calculations.

Subroutine 'MATHI' within the program given in Section 7.11.5, is an example of a RWC-MATH initialization routine.

Performing Calculations

Set up a command string with BCD data digits and command codes as desired. Wait until the number processor is ready to accept an input character. This can be detected either by polling the service request signal for RIC Port 0 (Port 2 Bit 3), or by causing it to generate a CPU interrupt by jumper selection. Send the entire command string to the number processor, a character at a time in this manner, until the table terminator FF is detected.

The results of any calculation can be read back from the X register of the number processor, by outputting an OUTT command to it. The number processor will then send the results synchronously to RIC Port 1, one character at a time. When reading the results from the number processor, wait until each character is loaded into RIC Port 1. This can be detected either by polling the service request signal for RIC Port 1 (Port 2 Bit 0), or by causing it to generate a CPU interrupt by jumper selection. When the number processor has output the entire result, it will be ready to accept an input character, and the service request signal for RIC Port 0 will become active.

Subroutine 'MATH' within the program given in Section 7. 11. 5, is an example of an interface routine illustrating input and output to the number processor, for polled operation.

For interrupt driven operation, the two service request signals for RIC Ports 0 and 1 are merged into a single interrupt request jumper connectable to one of the two interrupt request lines on the DCE-BUS. The interrupt service routine must test Bits 0 and 3 of RIC Port 2 to determine the interrupt source.

7. 11. 4. 3 User Options

Interrupt Generation Jumpers

The two service request interrupt signals for RIC Ports 0 and 1 are merged together and brought to jumper pad J1. It may be linked to pad J2 or J3, for connection to the DCE-BUS interrupt request line IN7 or EXINTR respectively.

The RWC-MATH module is usually delivered without any jumper connection.

7.11.4.4 Module Connector Definitions

System Connector

See Section 6.1.1 for the pin definitions.

Device Connector

There is no device connector on the RWC-MATH module.

7.11.4.5 Operational Requirements

Power Requirements

The RWC-MATH module requires two power supplies from the DCE-BUS. The values given below are for the quiescent state. Active state values are typically 20% higher.

+ 5V	:	120mA
- 5V	:	40mA

Environmental Requirements

Operating temperature	:	0°C to 55°C
Storage temperature	:	-25°C to +85°C
Relative humidity	:	upto 95% noncondensing

Bus Loading

The RWC-MATH module presents 1 unit load to the DCE-BUS (see Section 4.4).

7.11.5 TEST PROCEDURE

This section defines a simple test configuration and a test program for performing a basic functional test on the RWC-MATH module. Users are advised to carry out such a test procedure when necessary to establish the correct functioning of a module. The test program also provide: a good example of RWC-MATH module driver software.

Test Configuration

The following test program requires a RWC-MATH module, a DCE microcomputer, UPT Utility software package (version 2.0), a rack, power supply and a TTY or equivalent. The RWC-MATH module addres: select switch should be set to '4'. The program is entered from the Utility, and returns to the Utility at the end.

The program contains a test command string to perform the following calculation :

$$12.34 + 1234 \times 10^{-2}$$

On return to the Utility, the answer in decimal notation will be stored in the result buffer with start address RESLT + 1. The end of the answer field is indicated by a byte containing FF. The location RESLT is used to store the status byte read from RIC Port 2, containing the error flag etc. On return to the Utility, the contents of the RESLT buffer can be examined by the command :

```
D1043  104E   (display RESLT to RESLT + 11)
```

The following values will then be displayed on the console device :

```
CC 00 0A 02 04 06 08 00 00 00 00 FF
```

This represents the correct answer +24.68, in decimal notation.

The command string can be easily changed via the Utility, for other desired test computations.

```
                 ;
                 ;
                 ;       EQUATES FOR RWC-MATH COMMAND CODES.
                 ;
                 ;       NOTE THAT THE DOUBLE BYTE COMMANDS SHOULD
                 ;       BE GENERATED USING THE 'DW' ASSEMBLER
                 ;       COMMAND.
                 ;
                 ;
0039             YPX     EQU     39H     ; X<-Y+X, Y<-Z, Z<-T, T<-0
003A             YMX     EQU     3AH     ; X<-Y-X, Y<-Z, Z<-T, T<-0
003B             YTX     EQU     3BH     ; X<-Y*X, Y<-Z, Z<-T, T<-0
003C             YDX     EQU     3CH     ; X<-Y/X, Y<-Z, Z<-T, T<-0
0038             YTOX    EQU     38H     ; X<-Y**X, Y<-Z, Z<-T, T<-0
                 ;
3920             MPX     EQU     3920H   ; M<-M+X
3A20             MMX     EQU     3A20H   ; M<-M-X
3B20             MTX     EQU     3B20H   ; M<-M*X
3C20             MDX     EQU     3C20H   ; M<-M/X
                 ;
0037             ODX     EQU     37H     ; X<-1.0/X
0034             SQRT    EQU     34H     ; X<-X**0.5
0033             SQ      EQU     33H     ; X<-X**2
0032             TENX    EQU     32H     ; X<-10.0**X
0031             EX      EQU     31H     ; X<-E**X
0035             LN      EQU     35H     ; X<-LN  X
0036             LOG     EQU     36H     ; X<-LOG X
0024             SIN     EQU     24H     ; X<- SIN X
0025             COS     EQU     25H     ; X<- COS X
0026             TAN     EQU     26H     ; X<-TAN X
2420             ISIN    EQU     2420H   ; X<-ARCSIN X
2520             ICOS    EQU     2520H   ; X<-ARCCOS X
2620             ITAN    EQU     2620H   ; X<-ARCTAN X
002D             RAD     EQU     2DH     ; X<-RAD X
002C             DEG     EQU     2CH     ; X<-DEG X
                 ;
                 ;
000A             DP      EQU     0AH     ; DECIMAL POINT POSITION
000B             EE      EQU     0BH     ; FOLLOWING DIGITS ARE EXPONENT
000C             CS      EQU     0CH     ; CHANGE MANTISSA OR EXPONENT SIGN
0021             EN      EQU     21H     ; T<-Z, Z<-Y, Y<-X ; END DIGIT ENTRY.
0022             TOGM    EQU     22H     ; TOGGLE MODE (SCIENTIFIC <-> FP)
0018             SMDC    EQU     18H     ; SET MANTISSA DIGIT COUNT
                                         ; FOLLOWING BYTE MUST CONTAIN
                                         ;    REQUIRED MANTISSA DIGIT COUNT.
002F             MCLR    EQU     2FH     ; MASTER CLEAR: X, Y, Z, T, M<-0;
                                         ;               ERR FLAG<-0; MDC<-8;
                                         ;               MODE<-FLOATING POINT.
002B             ECLR    EQU     2BH     ; ERR FLAG<-0
```

```
002E            POPS     EQU    2EH       ; X<-Y, Y<-Z, Z<-T, T<-0
0023            ROLL     EQU    23H       ; X<-Y, Y<-Z, Z<-T, T<-X
0030            XEY      EQU    30H       ; X<->Y
001B            XEM      EQU    1BH       ; X<->M
001C            MS       EQU    1CH       ; M<-X
001D            MR       EQU    1DH       ; X<-M
000D            PI       EQU    0DH       ;                  X<-3.1415927
001E            LSH      EQU    1EH       ; LEFT SHIFT MANTISSA OF X
001F            RSH      EQU    1FH       ; RIGHT SHIFT MANTISSA OF X
003F            NOOP     EQU    3FH       ; NULL COMMAND (NOP), TERMINATES
                                          ;      DIGIT ENTRY.
0016            OUTT     EQU    16H       ; MULTIDIGIT OUTPUT FROM X
00FF            TERM     EQU    0FFH      ; TABLE TERMINATOR
                ;
                ;
                ;        TEST PROGRAM FOR MATH MODULE ROUTINES
                ;                (POLLED OPERATION)
                ;        MODULE ADDRESS SWITCH SET TO '4'.
                ;
1000                     ORG    01000H
1000 316B10              LXI    SP,STACK
                ;
1003 3E40                MVI    A,040H    ; INIT MATH CARD
1005 CDE010              CALL   MATHI
                ;
1008 211610              LXI    H,CMDS    ; ADDRESS COMMAND STRING
100B 114410              LXI    D,RESLT+1        ; ADDRESS RESULT AREA
100E 3E40                MVI    A,040H    ; RWC ADDR
1010 CD6B10              CALL   MATH      ; PERFORM FUNCTION
                ;
1013 C30000              JMP    0         ; RETURN TO MONITOR
                ;
                                          ; EXAMPLE COMMAND STRING:
                ;
1016 01020A03   CMDS:    DB     1,2,DP,3,4,EN    ; = 12.34
101A 0421
101C 01020304            DB     1,2,3,4,EE,2,CS ; = 1234 * 10**-2 (= 12.34)
1020 0B020C
1023 39FF                DB     YPX,TERM
1025                     DS     30        ; LEAVE SOME SPACE
1043            RESLT:   DS     20        ; RESULT BUFFER
1057                     DS     20
106B            STACK    EQU    $
                ;
                ;
                ;
                ;
                ;
                ;
```

```
                        ;
                        ;
                        ;       EQUATES
                        ;
0008                    INTRO   EQU     08H       ; MASK FOR PORT 0 INTERRUPT
0001                    INTR1   EQU     01H       ; MASK FOR PORT 1 INTERRUPT
0001                    GICP1   EQU     1         ; USED IN GICC MACRO
1C02                    GICP2   EQU     1C02H
031E                    RDRWC   EQU     031EH
0349                    WRRWC   EQU     0349H
                        ;
                        ;
                        ;
                        ;       MATH - MATH MODULE INTERFACE ROUTINE
                        ;       (POLLED OPERATION)
                        ;
                        ;       INPUT PARAMETERS:
                        ;       ACC - RWC ADDRESS IN HIGH ORDER 4 BITS
                        ;       H,L - START ADDRESS OF COMMAND STRING
                        ;       D,E - START ADDRESS FOR RESULT
                        ;       NO REGISTERS DESTROYED
                        ;
106B F5                 MATH:   PUSH    PSW
106C C5                         PUSH    B
106D D5                         PUSH    D         ; SAVE ALL REGISTERS
106E E5                         PUSH    H
                        ;
106F 47                         MOV     B,A       ; SAVE RWC ADDRESS
                        ;
1070 CDB710                     CALL    MATO      ; OUTPUT COMMAND STRING
                        ;
1073 21B410                     LXI     H,MATOI ; SEND OUT 'OUTPUT' REQUEST
1076 CDB710                     CALL    MATO    ;      COMMAND
                        ;
                      +         GICC    2,0       ; INITIALIZE THE GIC FOR INPUT
1079 3E90
107B 32031C

                                                  ; FOLLOWING CODE READS THE
                                                  ; SYNCHRONOUS OUTPUT FROM
                                                  ; NUMBER PROCESSOR
                        ;
107E 21021C                     LXI     H,GICP2 ; REAL WORLD CONTROL
1081 36FF                       MVI     M,0FFH  ; DISABLE CONTROL LINES
1083 78                         MOV     A,B     ; GET RWC ADDRESS
1084 F602                       ORI     2       ; SELECT RIC PORT 2 ADDRESS
1086 2B                         DCX     H       ; REG 1 RWC ADDRESS
1087 77                         MOV     M,A     ; SET UP RIC PORT 2 ADDRESS
1088 01FFFB                     LXI     B,0FBFFH        ; CONTROL BYTES
```

```
                MAT1:
108B 23                 INX     H           ; ADDRESS CONTROL
108C 70                 MOV     M,B         ; READ ACTIVE
                +       LDGI    0           ; READ RIC PORT 2 DATA
108D 3A001C

1090 71                 MOV     M,C         ; DISABLE READ
1091 2B                 DCX     H           ; BACK TO ADDRESS PORT
1092 E609               ANI     INTR0+INTR1       ; POLL INTERRUPTS
1094 CA8B10             JZ      MAT1        ; LOOP UNTIL CHAR READY
1097 E608               ANI     INTR0       ; TEST WHICH INTERRUPT
1099 C2AC10             JNZ     MAT2        ; JUMP IF END OF OUTPUT
                ;
109C 35                 DCR     M           ; ADDRESS PORT 1 RWC
109D 23                 INX     H           ; CONTROL PORT 2
109E 70                 MOV     M,B         ; ENABLE READ
                +       LDGI    0           ; READ RIC PORT 1
109F 3A001C

10A2 71                 MOV     M,C         ; DISABLE READ
10A3 2B                 DCX     H           ; ADDRESS RWC ADDRESS
10A4 34                 INR     M           ; RESELECT RIC PORT 2
10A5 E60F               ANI     0FH         ; CLEAR ADDRESS BITS FROM DATA
10A7 12                 STAX    D           ; STORE IN RESULT BUFFER
10A8 13                 INX     D           ; ADDRESS NEXT CHAR
10A9 C38B10             JMP     MAT1
                ;
10AC 3EFF       MAT2:   MVI     A,TERM      ; SET TERMINATION FLAG
10AE 12                 STAX    D
                ;
10AF E1                 POP     H
10B0 D1                 POP     D           ; RESTORE REGISTERS
10B1 C1                 POP     B
10B2 F1                 POP     PSW
                ;
10B3 C9                 RET                 ; RETURN TO CALLER
                ;
10B4 1600FF     MATOI:  DB      OUTT,0,TERM
                                            ; COMMAND STRING TO GET OUTPUT DATA
                                            ; FROM X REGISTER OF NUMBER PROCESSOR.

                ;
                ;
                ;       MATO - OUTPUT STRING TO MATH MODULE
                ;
                ;       INPUT PARAMETERS:
                ;       B    - RWC ADDRESS (H.O. 4 BITS)
                ;       H,L - ADDRESS OF STRING TO BE OUTPUT ( TERMINATED
                ;                BY 'TERM' CHARACTER)
```

```
                 ;
10B7 7E          MATO:   MOV     A,M     ; SEE IF ALL OF STRING HAS
10B8 FEFF                CPI     TERM    ;    BEEN SENT
10BA C8                  RZ              ; EXIT IF NO MORE
                 ;
10BB 78                  MOV     A,B     ; SELECT RIC PORT 2
10BC F602                ORI     2
               +         STGI    GICP1
10BE 32011C

10C1 CD1E03              CALL    RDRWC   ; READ STATUS
                                         ; ERROR FLAG MUST BE CHECKED
                                         ;   AFTER EACH OPERATION IN
                                         ;   COMMAND STRING.
10C4 4F                  MOV     C,A
10C5 3A4310              LDA     RESLT   ; RESULT STATUS FLAG BYTE
10C8 E610                ANI     10H     ; CLEAR ALL EXCEPT ERR BIT
10CA B1                  ORA     C       ; ADD TO NEW STATUS
10CB 324310              STA     RESLT   ; RESTORE
10CE 79                  MOV     A,C     ; RELOAD NEW STATUS
10CF E608                ANI     INTRO   ; LOOP UNTIL NUMBER PROCESSOR
10D1 CAB710              JZ      MATO    ;   READY FOR NEXT CHARACTER.
                 ;
10D4 78                  MOV     A,B
               +         STGI    GICP1   ; STORE PORT 0 ADDRESS
10D5 32011C

10D8 7E                  MOV     A,M     ; LOAD NEXT COMMAND OR DATA
10D9 23                  INX     H       ; ADDRESS NEXT CHAR.
10DA CD4903              CALL    WRRWC   ; OUTPUT COMMAND
10DD C3B710              JMP     MATO    ; LOOP FOR NEXT
                 ;
                 ;
                 ;
                 ;       MATHI - MATH MODULE INITIALIZATION ROUTINE
                 ;
                 ;       INPUT PARAMETERS
                 ;       ACC - RWC ADDRESS IN UPPER 4 BITS
                 ;             O IN LOWER 4 BITS
                 ;
                 ;       NO REGISTERS DESTROYED
                 ;
10E0 E5          MATHI:  PUSH    H
10E1 D5                  PUSH    D
10E2 C5                  PUSH    B
10E3 F5                  PUSH    PSW     ; SAVE ALL REGISTERS
                 ;
10E4 47                  MOV     B,A     ; PUT RWC ADDR IN B
                 ;
```

```
10E5 F603               ORI      3
                +       STGI     GICP1    ; SELECT RIC COMMAND REGISTER
10E7 32011C

10EA 3EAE               MVI      A,0AEH   ; CONFIGURE PORT 0 AS H.S.O/P,
10EC CD4903             CALL     WRRWC    ; PORT 1 AS H.S.I/P, PORT 2
                                          ; BITS 4,5 AS I/P.
                 ;
10EF 78                 MOV      A,B      ; ADDRESS PORT 0
                +       STGI     GICP1
10F0 32011C

10F3 3E40               MVI      A,040H   ; SET THE POWER ON RESET LINE OF
10F5 CD4903             CALL     WRRWC    ;     MATH MODULE HIGH
                                          ; REMOVED LATER DURING INPUT.
                 ;
10F8 78                 MOV      A,B
10F9 F603               ORI      3
                +       STGI     GICP1    ; ADDRESS COMMAND REG
10FB 32011C

10FE 3E05               MVI      A,05H    ; SET INTR0 ACTIVE
1100 CD4903             CALL     WRRWC
1103 3E0D               MVI      A,0DH    ; SET INTR1 ACTIVE
1105 CD4903             CALL     WRRWC
                                          ; (PORT 2 BITS 2 AND 6 CONTROL
                                          ;   INTR0 AND INTR1 RESPECTIVELY)
                 ;
1108 78                 MOV      A,B
                +       STGI     GICP1    ; ADDRESS PORT 0
1109 32011C

110C 3E3F               MVI      A,NOOP   ; SEND OUT NOP TO NUMBER PROCESSOR
110E CD4903             CALL     WRRWC
                 ;
1111 78                 MOV      A,B
1112 3C                 INR      A        ; ADDRESS PORT 1
                +       STGI     GICP1
1113 32011C

1116 CD1E03             CALL     RDRWC    ; DUMMY READ PORT 1
                 ;
1119 AF                 XRA      A
111A 324310             STA      RESLT    ; ZERO RESULT STATUS BYTE
                 ;
111D 212D11             LXI      H,CLEAR  ; MASTER CLEAR SEQUENCE
1120 114410             LXI      D,RESLT+1
1123 3E40               MVI      A,040H
1125 CD6B10             CALL     MATH
```

```
                  ;
1128 F1                   POP      PSW
1129 C1                   POP      B
112A D1                   POP      D          ; RESTORE REGISTERS
112B E1                   POP      H
                  ;
112C C9                   RET
                  ;
112D 3F3F3F       CLEAR:  DB       NOOP,NOOP,NOOP
1130 2F3FFF               DB       MCLR,NOOP,OFFH  ; MASTER CLEAR
                  ;
0000                      END
```

7.11.6 ORDERING INFORMATION

RWC-MATH : Standard Version

7.12 RWC-CO4 : ISOLATED ANALOG CURRENT OUTPUT

7.12.1 FUNCTIONAL DESCRIPTION

The RWC-CO4 Real-World interface module provides four identical isolated channels for outputting precise analog currents under DCE program control. Each current output channel has a range of 4 to 20mA, definable with a digital resolution of 16 bits. The 4 current output channels are independent, and each given current value is held stable until changed by DCE software.

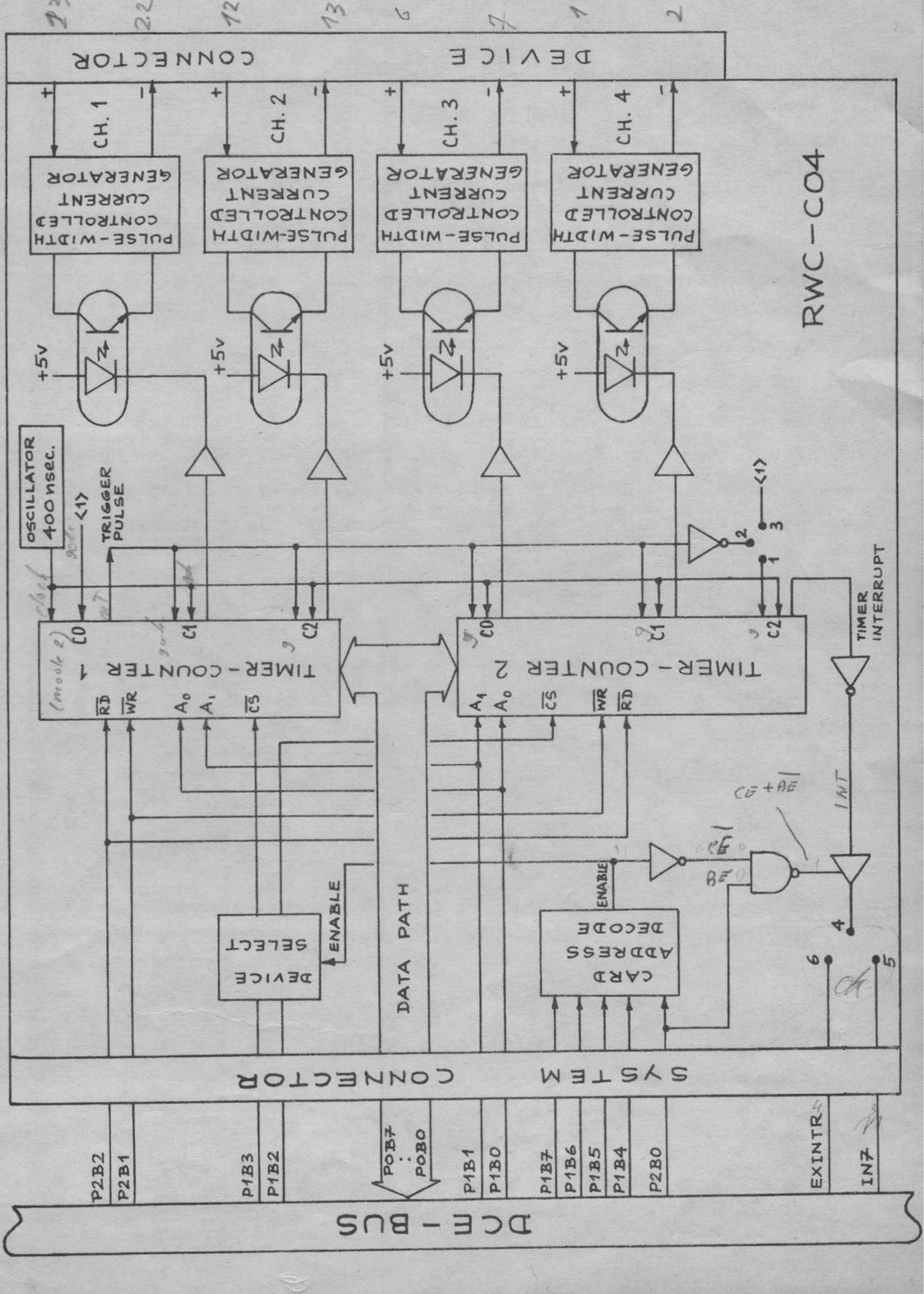
This module has the unique feature of total isolation between each analog channel and the DCE system. To ensure complete isolation, the drive voltage for each current generating channels is derived from the connecting device's base supply. The module is ideal for precision control environments where industrial noise and ground loops make such control difficult.

Each module has an identification address defined by a hexadecimal switch and up to fifteen cards may be directly connected to the DCE-BUS.

7.12.2 FEATURES

- 4 identical simultaneous independent current output channels.
- 4 to 20mA programmable current outputs with 16-bit digital resolution.
- total opto-isolation between each analog output channel and DCE system.
- each channel current output held stable until changed to a new value.
- drive voltage for each output channel derived from the connecting device's base supply.
- standard hardware and software interface to the DCE-BUS.
- selectable card address.
- single 100 x 160mm eurocard format.

7.12.3 : FUNCTIONAL BLOCK DIAGRAM



7. 12. 4 SYSTEM DESIGN PARAMETERS

7. 12. 4. 1 Hardware Configuration

The functional block diagram in Section 7. 12. 3 illustrates the hardware configuration. The module does not use a single device RIC for interfacing to the DCE-BUS. The RWC-CO4 module works on the principle of modulating the duty cycle of a square wave, averaging the voltage at the other side of an opto-isolator, and linearly converting to the equivalent analog current in the range 4 to 20mA.

The module has two programmable timer-counter devices (8253), each with three independent 16-bit counters C0, C1, C2. The period for all the counters is derived from counter C0 of device 1. This overall period as well as the pulse widths of the generated square waves (via counters C1, C2 and C0, C1 of device 1 and 2 respectively), can be independently software programmed.

The RWC-CO4 module contains the hardware logic to realize the multiple vectored interrupt scheme on the DCE-BUS, described in Section 4. 3. 2.

7. 12. 4. 2 Programming Specifications

zie ook µcomputer interfacing

The RWC-CO4 module is addressed via the standard DCE-BUS interface. Programming specifications for driving the DCE-BUS are given in Section 4. 1.

The module does not have the usual single device RIC interface to the DCE-BUS. In addition to the 4-bit card address and the 2-bit register address, it also requires a 2-bit device address for the two timer-counters. The 8-bit card/device address received from GIC Port 1 via the DCE-BUS, is interpreted as follows:

b7 - b4	=	card select address (1 to F)
b3 - b2	=	device select address (1 to 2)
b1 - b0	=	internal register select address within selected device (1 to 4)

For module activation, the card select address must correspond to the setting of the hexadecimal address switch on the module. The two device select address bits individually enable timer-counter devices 1 and 2, when the module is correctly addressed. The register select address bits select 1 out of the 4 internal registers within each timer/counter device.

<u>Device/Counter Addresses</u>

The two timer-counter devices on the module each has a Control Word Register and three counters. The functional definition of each timer-counter device is software programmable. A control word must be sent to these devices to initialize each counter with the desired Mode and quantity information.

The table 7. 12. 1 indicates the addressing necessary for the relevant operations.

The table 7. 12. 2 shows the Mode Word values for defining the operation modes of Counters C0, C1, C2 on each timer-counter device. These must be used for configuring the corresponding Counters during module initialization.

During module initialization, the period for all the channels is set to a constant value by writing a 16-bit count value to Counter C0 of device 1 (least significant 8-bits must be written first). The duration of each pulse (duty cycle) is then programmed to provide the required analog output by writing a 16-bit counter value to the Counter associated with that output channel. The table 7. 12. 3 specifies the Counter associated with each analog output channel.

ADDRESS FOR TIMER-COUNTER 1 (HEX)	ADDRESS FOR TIMER-COUNTER 2 (HEX)	$\overline{RD}$	$\overline{WR}$	OPERATION
OUT Y0 period	OUT Y4 CH3	1	0	Load counter C0
OUT Y1 CH1	OUT Y5 CH4	1	0	Load counter C1
OUT Y2 CH2	OUT Y6 INTOEN	1	0	Load counter C2
OUT Y3 MODE reg1	OUT Y7 MODE Reg2	1	0	Write mode word
ZX	ZX	X	X	Data Bus in 3-state

Notes:

1. Y is the card address select switch setting in hex (1 to F).
2. Z is any number other than Y.
3. X means don't care.
4. Bits 2 and 3 are used directly for timer-counter device enable.
5. RDRWC and WRRWC software routines provide the $\overline{RD}$ and $\overline{WR}$ signals accordingly.

Table 7. 12. 1 : Device/Counter Address Table for RWC-CO4

COUNTER	TIMER-COUNTER 1 MODE WORD (HEX)	TIMER-COUNTER 2 MODE WORD (HEX)
C0	mode 2 $\rightarrow$ 34 or 3C	32 mode 1
C1	72 mode 1	72 mode 1
C2	B2	B4 or BC

Table 7. 12. 2 : Mode Word Value Table

ANALOG OUTPUT CHANNEL NO.	TIMER-COUNTER 1 COUNTER	TIMER-COUNTER 2 COUNTER
1	C1	-
2	C2	-
3	-	C0
4	-	C1
Period Generator	C0	-
Interrupt Generator	-	C2

Table 7. 12. 3 : Analog Output Channel - Counter Relationship

Format of Data

The data for the various counters must be 2 bytes wide and in binary. The data should be written with the least significant byte first, followed by the most significant byte.

Special Considerations

The values in the counters corresponding to the 4 analog output channels must never be equal to or greater than the counter value in the period counter (C0 in device 1).

The systems software must program each counter of the timer-counter devices with the mode and quantity desired. Writing out the Mode control word can be in any sequence for the three counters. This operation should be followed by the loading of the actual count value into the selected counter register, with the least significant byte first.

The two-byte value to be loaded into the counter register does not have to follow the associated Mode control word. They can be programmed at any time following the Mode control word, as long as the two bytes are loaded in the correct order.

If the Mode word for a timer-counter device is to be changed, it must be done after two successive Read operations to clear the internal control logic of the device.

Notes on the Usage of RWC-CO4

The four output current channels are polarized, and their signal lines must carry the connecting device's base supplies with a suitable load resistance (see Section 7. 12. 4. 5). The optimum effective load resistance for a given supply voltage shown in Figure 7. 12. 1, ensures minimum power dissipation resulting in low drift and stable operation while producing the 4-20 mA range.

A resistance lower than this optimum value may be used for a given supply voltage to produce the same 4-20 mA current output range. But, this will cause higher heat dissipation on the current drivers, which could cause drift and lower stability.

A resistance higher than the above optimum value will lower the upper limit of the output current. The current range will then be from 4 mA to a value less than 20 mA.

The output current corresponding to a particular digital number loaded into a channel counter will be independant of the operating point on the load resistance - supply voltage curve.

The following setting is used for testing the RWC-CO4 module in the factory. Other settings may be used if desired.

- C0 of Timer-Counter 1 = 5600H (initialization)
- writing 0500H to any channel will produce 4 mA output
- writing 4500H to any channel will produce 20 mA output.

This setting gives a count range of 4000H for the 16 mA range, and 1 mA output for 0400H step. These settings are recommended to ensure linearity.

7.12.4.3 User Options

The RWC-CO4 provides the facility for software programmable time interval interrupt generation. The signal is derived from counter C2 of device 2, and may be connected to the IN7 or EXINTR interrupt requests on the DCE-BUS via a jumper network.

Alternatively, it is possible to jumper select an interrupt request in synchronisation with the trigger pulse for the four analog channels, provided by counter C0 of device 1.

See the Functional Block Diagram in Section 7.12.3 for details of jumper connections.

7.12.4.4 Module Connector Definitions

System Connector

See Section 6.1.4 for the pin definitions.

Device Connector (25-pin D-type female)

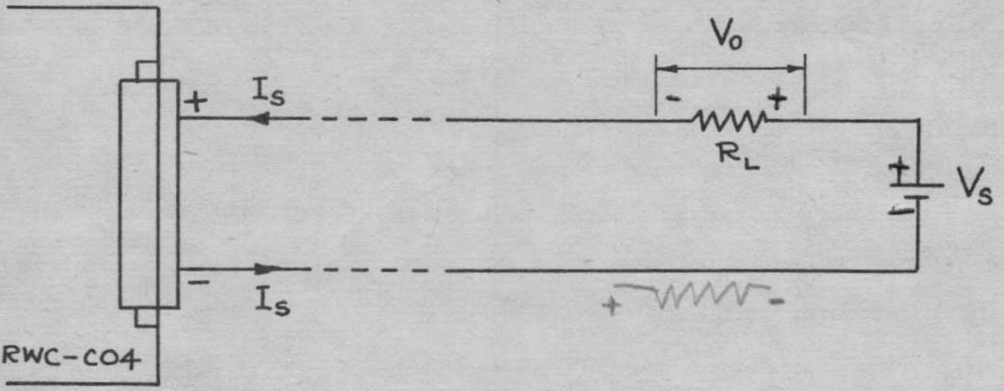
Pin Number	Signal
1	Channel 3 +
2	Channel 3 -
6	Channel 2 +
7	Channel 2 -
12	Channel 1 +
13	Channel 1 -
22	Channel 0 -
23	Channel 0 +

7.12.4.5 Operational Requirements

Signal Characteristics

The four output analog channels are polarized, and their signal lines must carry the connecting device's base supplies necessary for deriving the current generating channel drive voltages.

The diagram below illustrates one method of connecting the base voltages and signal lines to the module:



The above scheme provides a voltage output (V_0) with respect to the positive rail. By connecting the load resistance on the other side of V_s, it is possible to get a voltage output with respect to the negative rail.

Power Requirements

The RWC-CO4 uses a single +5V supply available on the DCE-BUS, for the digital configuration. Typical consumption is 240mA.

The base voltages necessary for deriving the current generating channel drive voltages must be provided by the connecting devices, as shown before. The base voltage should not exceed 50V. The graph below defines the optimum load resistance (R_L) in KΩ for a particular supply voltage. Note that the load resistance is the sum total of the resistances on the lines (load resistance + d. c. resistance of transmission line).

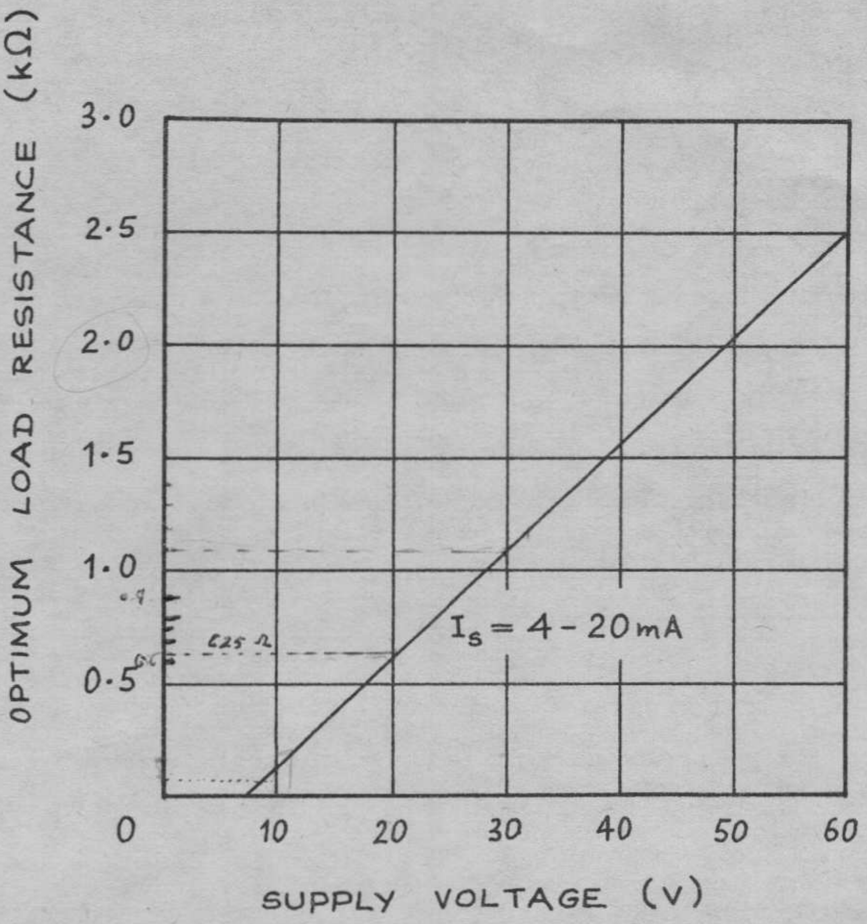


Figure 7. 12. 1 : Optimum Load Resistance (R_L) vs Supply Voltage (V_s)

Environmental Requirements

Operating temperature	:	0°C to 55°C
Storage temperature	:	-25°C to +85°C
Relative humidity	:	95% noncondensing

Bus Loading

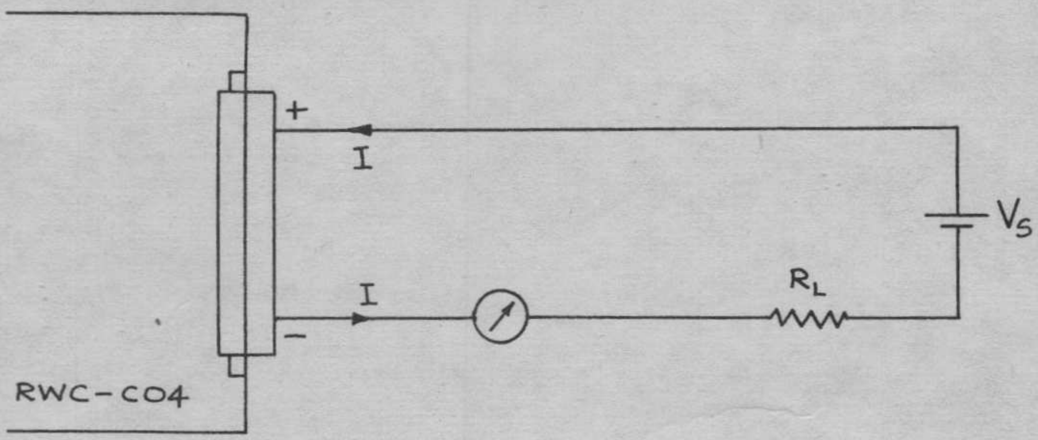
The RWC-CO4 module is equivalent to 2 unit-loads on the DCE-BUS (see Section 4. 4).

7. 12. 5 TEST PROCEDURE

This section defines a simple test configuration and a test program for performing a basic functional test on the RWC-CO4 module. Users are advised to carry out such a test procedure when necessary to establish the correct functioning of a module. The test program also provides a good example of RWC-CO4 module driver software.

Test Configuration

The following test program relates to the test configuration below. The quiescent current and the full range currents are both read in the dynamic state. The program enables the user to put any suitable value in the counter associated with a particular channel, and to get the expected value of output current.



```
                ; RWC-C04 MODULE TEST PROGRAM
                ; TYPE CHANNEL NO BETWEEN 0-5
                ; FOLLOWED BY CHANNEL COUNT
                ; CHANNEL 0 IS PERIOD TRIGGER PULSE GENERATOR
                ; FOR THE OTHER CHANNELS
                ; SET CARD ADDRESS SELECT SWITCH TO '2'.
                                        ;
                ORG     1020H   ;PROGRAM START LOCATION
        START:  LXI     SP,17C0H        ;INITIALISE STACK POINTER
                CALL    TCRLF   ;DO A CARRIAGE RET LINE FEED
                CALL    CIE     ;GET A CHARACTER &ECHO
                ANI     07H     ;MASK .>5 CH NO
                STA     KEY     ;STORE CH. NO. IN KEY
                CALL    TSP     ;OUTPUT SPACE TO
                                ;CONSOLE
                MVI     C,01    ;INITIALISE ARGUMENT REG
                CALL    ADARG   ;READ 2 BYTE DATA FROM
                                ;CONSOLE
                POP     H       ;PUT 2 BYTE DATA INTO H&L
                SHLD    DATAL   ;MOVE CONTENTS OF H&L
                                ;INTO 2 CONT LOCATION
                                ;STARTING AT DATAL


        SORT:   LXI     H,KEY   ;LOAD NAME KEY IN ACC
                MOV     A,M     ;PUT CH. NO DATA INTO ACC
                ANA     A       ;GENERATE ZERO FLAG
                JZ      TRIG    ;JUMP TO SPECIAL CONTROL
                                ;WORD SUBTINE
                MOV     A,M     ;PUT CH. NO. INTO ACC
                CPI     03H     ;CHECK IF DATA IN KEY
                                ;>2
                JNC     MAN     ;JUMP TO MANIPULATE
                                ;>2 COUNTER ADD
                MOV     A,M     ;MOVE CH. NO. INTO
                                ;ACC
                RRC
                RRC
                ADI     32H     ;FORM COMTROL WORD
                                ;FOR MODE 1
                LXI     B,COND  ;LOAD NAME COND INTO
                                ;B&C REG
                STAX    B       ;STORE ACC IN COND
                JMP     CONT
        MAN:    MOV     A,M     ;PUT CH. NO IN ACC
                SBI     03H     ;SUBTRACT 3 FROM CH. NO
                RRC
                RRC
                ADI     32H     ;ADD 32 TO FORM CONTROL WORD
                LXI     B,COND  ;LOAD NAME COND IN B&C
                STAX    B       ;LOAD ACC IN COND
```

```
CONT:   MOV     A,M       ;PUT CH. NO.  IN ACC
        CPI     03H       ;CHECK IF DATA IN
                          ;>2
        JNC     CHIP2     ;IF NO CARRY CHIP2 USED
        MVI     A,23H     ;LOAD CHIP 1 CONTROL
                          ;REG ADD
        LXI     B,COADD          ;LOAD B&C WITH COADD
        STAX    B         ;STORE ACC IN COADD
        MOV     A,M       ;MOVE CH. NO.  IN ACC
        ADI     20H       ;FORM CH. ADD
        LXI     B,CHNO    ;LOAD NAME CHNO IN
                          ;REG B&C
        STAX    B         ;STORE ACC INTO CHNO
        JMP     INIT
CHIP2:  MVI     A,27H     ;LOAD CHIP2 CONTROL REG ADD
        LXI     B,COADD          ;LOAD B&C WITH COADD
        STAX    B         ;STORE ACC IN COADD
        MOV     A,M       ;MOVE CH. NO. IN ACC
        SBI     03H       ;SUBTRACT 3
        ADI     24H       ;FORM CH ADD FOR CHIP2
        LXI     B,CHNO    ;LOAD NAME CHNO IN B&C
        STAX    B         ;STORE ACC IN CHNO
        JMP     INIT

TRIG:   MVI     A,34H     ;LOAD ACC WITH CHO
                          ;CONTROL DATA
        LXI     D,COND    ;LOAD NAMECOND IN D&E
        STAX    D         ;STORE ACC IN COND
        MVI     A,23H     ;LOAD CHO CONTROL ADD
        LXI     B,COADD          ;PUT NAME COADD IN B&C
        STAX    B         ;STORE ACC IN COADD
        MVI     A,20H     ;LOAD CHIP1 CHO ADD
        LXI     B,CHNO    ;LOAD NAME CHNO IN B&C
        STAX    B         ;STORE ACC IN CHNO
        JMP     INIT
```

```
INIT:   LXI     D,COADD         ;LOAD MAME COADD IN D&E
        LDAX    D       ;LOAD CONTENTS OF COADD
                        ;IN ACC
        STA     GICB    ;LOAD ADD ON THE BUS
        LXI     D,COND  ;LOAD NAME COND IN D&E
        LDAX    D       ;LOAD CONTROL WORD INTO ACC
        CALL    WRRWC   ;WRITE CONTROL DATA 1N 8253
        LXI     D,CHNO  ;LOAD NAME CHNO IN D&E
        LDAX    D       ;LOAD CHNO ADD IN ACC
        STA     GICB    ;STORE IT ON BUS
        CALL    RDRWC   ;CLEAR LOWER BYTE REG
        CALL    RDRWC   ;CLEAR HIGHER BYTE REG
        LXI     D,COADD         ;LOAD NAME COADD IN D&E
        LDAX    D       ;LOAD ACC WITH CONTROL REG
                        ;ADD
        STA     GICB    ;LOAD ADD ON THE BUS
        LXI     D,COND  ;LOAD NAME COND IN D&E
        LDAX    D       ;LOAD CONTROL WORD IN ACC
        CALL    WRRWC   ;WR1TE CONTROLWORD INTO 8253
        LXI     D,CHNO  ;LOAD NAME CHNO IN D&E
        LDAX    D       ;LOAD CH NO IN ACC
        STA     GICB    ;STORE CHNO ON BUS
        LXI     D,DATAL         ;LOAD NAME DATAL IN D&E
        LDAX    D       ;LOAD LOWER DATA BYTE INTO ACC
        CALL    WRRWC   ;LOAD DATA IN TO 8253
        LXI     D,DATAM         ;LOAD NAME DATAM IN D&E
        LDAX    D       ;LOAD HIGHER DATA BYTE INTO ACC
        CALL    WRRWC   ;LOAD DATA IN 8253
        JMP     START

        RDRWC   EQU     031EH
        WRRWC   EQU     0349H
        CIE     EQU     0561H
        TCRLF   EQU     061FH
        TSP     EQU     053AH
        ADARG   EQU     01AEH
        TADDR   EQU     05FDH
        GICB    EQU     1C01H
        COADD   EQU     1000H
        COND    EQU     1001H
        CHNO    EQU     1002H
        DATAL   EQU     1003H
        DATAM   EQU     1004H
        KEY     EQU     1005H

        END
```

7.12.6 ORDERING INFORMATION

RWC-CO4 : Standard Version