8231A

Arithmetic Processing Unit

The Intel 8231A Arithmetic Processing Unit (APU) is a monolithic HMOS LSI device that provides high performance fixed and floating point arithmetic and floating point trigonometric operations. It may be used to enhance the mathematical capability of a wide variety of processor-oriented systems. Chebyshev polynomials are used in the implementation of the APU algorithms.

All transfers, including operand, result, status and command information, take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack and commands are issued to perform operations on the data and the stack. Results are then available to be retrieved from the stack.

Rochester Electronics Manufactured Components

Rochester branded components are manufactured using either die/wafers purchased from the original suppliers or Rochester wafers recreated from the original IP. All recreations are done with the approval of the OCM.

Parts are tested using original factory test programs or Rochester developed test solutions to guarantee product meets or exceeds the OCM data sheet.

Quality Overview

- ISO-9001
- AS9120 certification
- Qualified Manufacturers List (QML) MIL-PRF-38535
    - Class Q Military
    - Class V Space Level
- Qualified Suppliers List of Distributors (QSLD)
    - Rochester is a critical supplier to DLA and meets all industry and DLA standards.

Rochester Electronics, LLC is committed to supplying products that satisfy customer expectations for quality and are equal to those originally supplied by industry manufacturers.

The original manufacturer's datasheet accompanying this document reflects the performance and specifications of the Rochester manufactured version of this device. Rochester Electronics guarantees the performance of its semiconductor products to the original OEM specifications. 'Typical' values are for reference purposes only. Certain minimum or maximum ratings may be based on product characterization, design, simulation, or sample testing.

8231A
ARITHMETIC PROCESSING UNIT

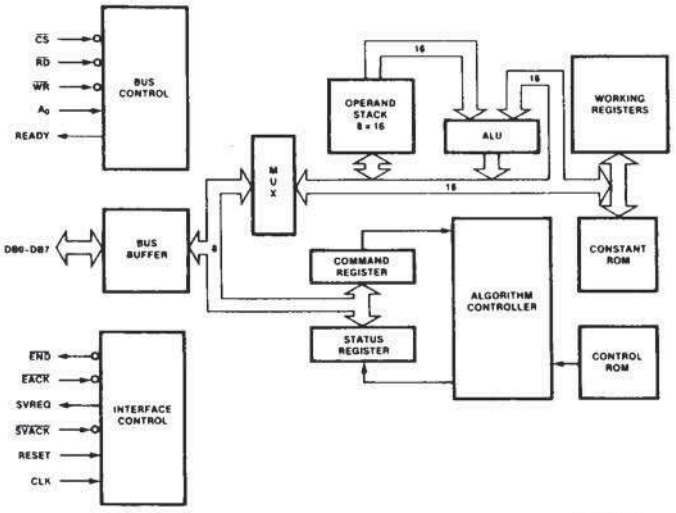
- **Fixed Point Single and Double Precision (16/32 Bit)**
- **Floating Point Single Precision (32 Bit)**
- **Binary Data Formats**
- **Add, Subtract, Multiply and Divide**
- **Trignometric and Inverse Trigonometric Functions**
- **Square Roots, Logarithms, Exponentiation**
- **Float to Fixed and Fixed to Float Conversions**
- **Stack Oriented Operand Storage**
- **Compatible with all Intel and most other Microprocessor Families**
- **Direct Memory Access or Programmed I/O Data Transfers**
- **End of Execution Signal**
- **General Purpose 8-Bit Data Bus Interface**
- **Standard 24 Pin Package**
- **+12V and +5V Power Supplies**
- **Advanced N-Channel Silicon Gate HMOS Technology**

The Intel® 8231A Arithmetic Processing Unit (APU) is a monolithic HMOS LSI device that provides high performance fixed and floating point arithmetic and floating point trigonometric operations. It may be used to enhance the mathematical capability of a wide variety of processor-oriented systems. Chebyshev polynomials are used in the implementation of the APU algorithms.

3

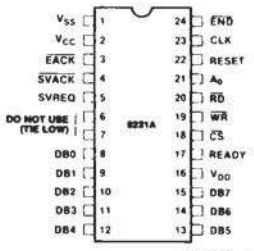
All transfers, including operand, result, status and command information, take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack and commands are issued to perform operations on the data and the stack. Results are then available to be retrieved from the stack.

Transfers to and from the APU may be handled by the associated processor using conventional programmed I/O, or may be handled by a direct memory access controller for improved performance. Upon completion of each command, the APU issues an end of execution signal that may be used as an interrupt by the CPU to help coordinate program execution.



231305-1

Figure 1. Block Diagram



231305-2

Figure 2. Pin Configuration

September 1987
Order Number: 231305-002

Table 1. Pin Description

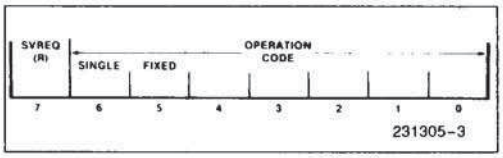
Symbol	Pin No.	Type	Name and Function
V_{CC}	2		**POWER:** +5V power supply.
V_{DD}	16		**POWER:** +12V power supply.
V_{SS}	1		**GROUND.**
CLK	23	I	**CLOCK:** An external, TTL compatible, timing source is applied to the CLK pin.
RESET	22	I	**RESET:** The active high reset signal provides initialization for the chip. RESET also terminates any operation in progress. RESET clears the status register and places the 8231A into the idle state. Stack contents and command registers are not affected (5 clock cycles).
$\overline{CS}$	18	I	**CHIP SELECT:** $\overline{CS}$ is an active low input signal which selects the 8231A and enables communication with the data bus.
A_0	21	I	**ADDRESS:** In conjunction with the $\overline{RD}$ and WR signals, the A_0 control line establishes the type of communication that is to be performed with the 8231A as shown below:

A_0	$\overline{RD}$	$\overline{WR}$	Function
0	1	0	Enter data byte into stack
0	0	1	Read data byte from stack
1	1	0	Enter command
1	0	1	Read status

$\overline{RD}$	20	I	**READ:** This active low input indicates that data or status is to be read from the 8231A if CS is low.
$\overline{WR}$	19	I	**WRITE:** This active low input indicates that data or a command is to be written into the 8231A if CS is low.
$\overline{EACK}$	3	I	**END OF EXECUTION:** This active low input clears the end of execution output signal ($\overline{END}$. If $\overline{EACK}$ is tied low, the $\overline{END}$ output will be a pulse that is one clock period wide.
$\overline{SVACK}$	4	I	**SERVICE REQUEST:** This active low input clears the service request output (SVREQ).
$\overline{END}$	24	O	**END:** This active low, open-drain output indicates that execution of the previously entered command is complete. It can be used as an interrupt request and is cleared by $\overline{EACK}$, RESET or any read or write access to the 8231.
SVREQ	5	O	**SERVICE REQUEST:** This active high output signal indicates that command execution is complete and that post execution service was requested in the previous command byte. It is cleared by $\overline{SVACK}$, the next command output to the device, or by RESET.
READY	17	O	**READY:** This active high output indicates that the 8231A is able to accept communication with the data bus. When an attempt is made to read data, write data or to enter a new command while the 8231A is executing a command, READY goes low until execution of the current command is complete (See READY Operation, p. 6).
DB0–DB7	8-15	I/O	**DATA BUS:** These eight bidirectional lines provide for transfer of commands, status and data between the 8231A and the CPU. The 8231A can drive the data bus only when CS and RD are low.

COMMAND STRUCTURE

Each command entered into the 8231A consists of a single 8-bit byte having the format illustrated below:



Bits 0–4 select the operation to be performed as shown in the table. Bits 5–6 select the data format appropriate to the selected operation. If bit 5 is a 1, a fixed point data format is specified. If bit 5 is 0, floating point format is specified. Bit 6 selects the precision of the data to be operated upon by fixed point commands only (if bit 5 = 0, bit 6 must be 0). If bit 6 is a 1, single-precision (16-bit) operands are assumed. If bit 6 is a 0, double-precision (32-bit) operands are indicated. Results are undefined for all illegal combinations of bits in the command byte. Bit 7 indicates whether a service request is to be issued after the command is executed. If bit 7 is a 1, the service request output (SVREQ) will go high at the conclusion of the command and will remain high until reset by a low level on the service acknowledge pin ($\overline{\text{SVACK}}$) or until completion of execution of the succeeding command where service request (bit 7) is 0. Each command issued to the 8231A requests post execution service based upon the state of bit 7 in the command byte. When bit 7 is a 0, SVREQ remains low.

Table 2. 32-Bit Floating Point Instructions

Instruction	Description	Hex(1) Code	Stack Contents(2) After Execution A B C D	Status Flags(4) Affected
ACOS	Inverse Cosine of A	0 6	R U U U	S, Z, E
ASIN	Inverse Sine of A	0 5	R U U U	S, Z, E
ATAN	Inverse Tangent of A	0 7	R B U U	S, Z
CHSF	Sign Change of A	1 5	R B C D	S, Z
COS	Cosine of A (radians)	0 3	R B U U	S, Z
EXP	e^A Function	0 A	R B U U	S, Z, E
FADD	Add A and B	1 0	R C D U	S, Z, E
FDIV	Divide B by A	1 3	R C D U	S, Z, E
FLTD	32-Bit Integer to Floating Point Conversion	1 C	R B C U	S, Z
FLTS	16-Bit Integer to Floating Point Conversion	1 D	R B C U	S, Z
FMUL	Multiply A and B	1 2	R C D U	S, Z, E
FSUB	Subtract A from B	1 1	R C D U	S, Z, E
LOG	Common Logarithm (base 10) of A	0 8	R B U U	S, Z, E
LN	Natural Logarithm of A	0 9	R B U U	S, Z, E
POPF	Stack Pop	1 8	B C D A	S, Z
PTOF	Stack Push	1 7	A A B C	S, Z
PUPI	Push π onto Stack	1 A	R A B C	S, Z
PWR	B^A Power Function	0 B	R C U U	S, Z, E
SIN	Sine of A (radians)	0 2	R B U U	S, Z
SQRT	Square Root of A	0 1	R B C U	S, Z, E
TAN	Tangent of A (radians)	0 4	R B U U	S, Z, E
XCHF	Exchange A and B	1 9	B A C D	S, Z

Table 3. 32-Bit Integer Instructions

Instruction	Description	Hex(1) Code	Stack Contents(2) After Execution A B C D	Status Flags(4) Affected
CHSD	Sign Change of A	3 4	R B C D	S, Z, O
DADD	Add A and B	2 C	R C D A	S, Z, C, E
DDIV	Divide B by A	2 F	R C D U	S, Z, E
DMUL	Multiply A and B (R = lower 32-bits)	2 E	R C D U	S, Z, O
DMUU	Multiply A and B (R = upper 32-bits)	3 6	R C D U	S, Z, O
DSUB	Subtract A from B	2 D	R C D A	S, Z, C, O
FIXD	Floating Point to Integer Conversion	1 E	R B C U	S, Z, O
POPD	Stack Pop	3 8	B C D A	S, Z
PTOD	Stack Push	3 7	A A B C	S, Z
XCHD	Exchange A and B	3 9	B A C D	S, Z

Table 4. 16-Bit Integer Instructions

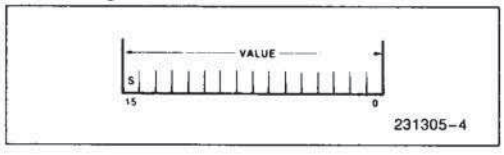
Instruction	Description	Hex(1) Code	Stack Contents(3) After Execution A_U A_L B_U B_L C_U C_L D_U D_L	Status Flags(4) Affected
CHSS	Change Sign of A_U	7 4	R A_L B_U B_L C_U C_L D_U D_L	S, Z, O
FIXS	Floating Point to Integer Conversion	1 F	R B_U B_L C_U C_L U U U	S, Z, O
POPS	Stack Pop	7 8	A_L B_U B_L C_U C_L D_U D_L A_U	S, Z
PTOS	Stack Push	7 7	A_U A_U A_L B_U B_L C_U C_L D_U	S, Z
SADD	Add A_U and A_L	6 C	R B_U B_L C_U C_L D_U D_L A_U	S, Z, C, E
SDIV	Divide A_L by A_U	6 F	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SMUL	Multiply A_L by A_U (R = lower 16-bits)	6 E	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SMUU	Multiply A_L by A_U (R = upper 16-bits)	7 6	R B_U B_L C_U C_L D_U D_L U	S, Z, E
SSUB	Subtract A_U from A_L	6 D	R B_U B_L C_U C_L D_U D_L A_U	S, Z, C, E
XCHS	Exchange A_U and A_L	7 9	A_L A_L B_U B_L C_U C_L D_U D_L	S, Z
NOP	No Operation	0 0	A_U A_L B_U B_L C_U C_L D_U D_L	

NOTES:
1. In the hex code column, SVREQ is a 0.
2. The stack initially is composed of four 32-bit numbers (A, B, C, D). A is equivalent to Top Of Stack (TOS) and B is Next On Stack (NOS). Upon completion of a command the stack is composed of: the result (R); undefined (U); or the initial contents (A, B, C, or D).
3. The stack initially is composed of eight 16-bit numbers (A_U, A_L, B_U, B_L, C_U, C_L, D_U, D_L). A_U is the TOS and A_L is NOS. Upon completion of a command the stack is composed of: the result (R); undefined (U); or the initial contents (A_U, A_L, B_U, B_L . . .).
4. Nomenclature: Sign (S); Zero (Z); Overflow (O); Carry (C); Error Code Field (E).

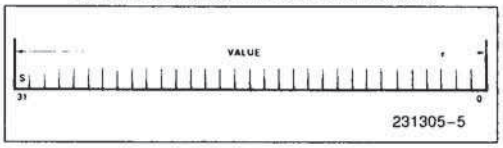
DATA FORMATS

The 8231A arithmetic processing unit handles operands in both fixed point and floating point formats. Fixed point operands may be represented in either single (16-bit operands) or double precision (32-bit operands), and are always represented as binary, two's complement values.

Single Precision Fixed Point Format



Double Precision Fixed Point Format



The sign (positive or negative) of the operand is located in the most significant bit (MSB). Positive values are represented by a sign bit of zero (S = 0). Negative values are represented by the two's complement of the corresponding positive value with a sign bit equal to 1 (S = 1). The range of values that may be accommodated by each of these formats is −32,768 to +32,767 for single precision and −2,147,483,648 to +2,147,483,647 for double precision.

Floating point binary values are represented in a format that permits arithmetic to be performed in a fashion analogous to operations with decimal values expressed in scientific notation.

$$(5.83 \times 10^{2})\,(8.16 \times 10^{1}) = (4.75728 \times 10^{4})$$

In the decimal system, data may be expressed as values between 0 and 10 times 10 raised to a power that effectively shifts the implied decimal point right or left the number of places necessary to express the result in conventional form (e.g., 47,572.8). The value-portion of the data is called the mantissa. The exponent may be either negative or positive.

The concept of floating point notation has both a gain and a loss associated with it. The gain is the ability to represent the significant digits of data with values spanning a large dynamic range limited only by the capacity of the exponent field. For example, in decimal notation in the exponent field is two digits wide, and the mantissa is five digits, a range of values (positive or negative) from 1.0000×10^{-99} to $9.9999 \times 10^{+99}$ can be accommodated. The loss is that only the significant digits of the value can be represented. Thus there is no distinction in this representation between the values 123451 and 123452, for example since each would be expressed as: 1.2345×10^{5}. The sixth digit has been discarded. In most applications where the dynamic range of values to be represented in large, the loss of significance, and hence accuracy of results, is a minor consideration. For greater precision a fixed point format could be chosen, although with a loss of potential dynamic range.

The 8231A is a binary arithmetic processor and requires that floating point data be represented by a fractional mantissa value between 0.5 and 1 multiplied by 2 raised to an appropriate power. This is expressed as follows:

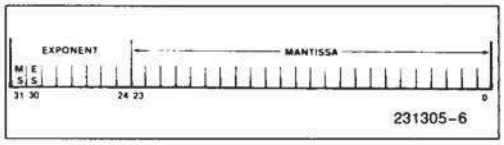
$$\text{value} = \text{mantissa} \times 2^{\text{exponent}}$$

For example, the value 100.5 expressed in this form is $0.1100\ 1001 \times 2^{7}$. The decimal equivalent of this value may be computed by summing the components (powers of two) of the mantissa and then multiplying by the exponent as shown below:

$$\begin{aligned}\text{value} &= (2^{-1} + 2^{-2} + 2^{-5} + 2^{-8}) \times 2^{7}\\ &= 0.5 + 0.25 + 0.03125 + 0.00290625) \times 128\\ &= 0.78515625 \times 128\\ &= 100.5\end{aligned}$$

FLOATING POINT FORMAT

The format for floating point values in the 8231A is given below. The mantissa is expressed as a 24-bit (fractional) value; the exponent is expressed as a two's complement 7-bit value having a range of −64 to +63. The most significant bit is the sign of the mantissa (0 = positive, 1 = negative), for a total of 32 bits. The binary point is assumed to be the left of the most significant mantissa bit (bit 23). All floating point data values must be normalized. Bit 23 must be equal to 1, except for the value zero, which is represented by all zeros.

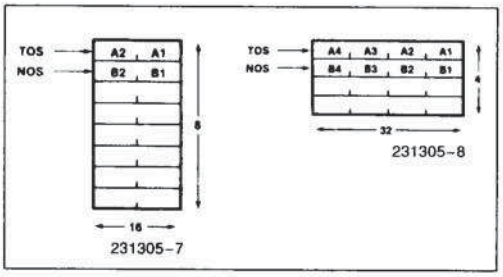


The range of values that can be represented in this format is $\pm(2.7 \times 10^{-20}$ to $9.2 \times 10^{18})$ and zero.

FUNCTIONAL DESCRIPTION

STACK CONTROL

The user interface to the 8231A includes access to an 8 level 16-bit wide data stack. Since single precision fixed point operands are 16-bits in length, eight such values may be maintained in the stack. When using double precision fixed point or floating point formats four values may be stored. The stack in these two configurations can be visualized as shown below:



Data are written onto the stack, eight bits at a time, in the order shown (A1, A2, A3, . . .). Data are removed from the stack in reverse byte order (A4, A3, A2 . . .). Data should be entered onto the stack in multiples of the number of bytes appropriate to the chosen data format.

DATA ENTRY

Data entry is accomplished by bringing the chip select ($\overline{CS}$), the command/data line (A_0), and $\overline{WR}$ low, as shown in the timing diagram. The entry of each new data word "pushes down" the previously entered data and places the new byte on the top of stack (TOS). Data on the bottom of the stack prior to a stack entry are lost.

DATA REMOVAL

Data are removed from the stack in the 8231A by bringing chip select ($\overline{CS}$), command/data (A_0), and $\overline{RD}$ low as shown in the timing diagram. The removal of each data world redefines TOS so that the next successive byte to be removed becomes TOS. Data removed from the stack rotates to the bottom of the stack.

COMMAND ENTRY

After the appropriate number of bytes of data have been entered onto the stack, a command may be issued to perform an operation on that data. Commands which require two operands for execution (e.g., add) operate on the TOS and NOS values. Single operand commands operate only on the TOS.

Commands are issued to the 8231A by bringing the chip select ($\overline{CS}$) line low, command data (A_0) line high, and $\overline{WR}$ line low as indicated by the timing diagram. After a command is issued, the CPU can continue execution of its program concurrently with the 8231A command execution.

COMMAND COMPLETION

The 8231A signals the completion of each command execution by lowering the End Execution line ($\overline{END}$). Simultaneously, the busy bit in the status register is cleared and the Service Request bit of the command register is checked. If it is a "1" the service request output level (SVREQ) is raised. $\overline{END}$ is cleared on receipt of an active low End Acknowledge ($\overline{EACK}$) pulse. Similarly, the service request line is cleared by recognition of an active low Service Acknowledge ($\overline{SVACK}$) pulse.

READY OPERATION

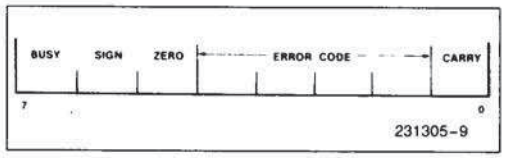
An active high ready (READY) is provided. This line is high in its quiescent state and is pulled low by the 8231A under the following conditions:

1. A previouly initiated operation is in progress (device busy) and Command Entry has been attempted. In this case, the READY line will be pulled low and remain low until completion of the current command execution. It will then go high, permitting entry of the new command.
2. A previously initiated operation is in progress and stack access has been attempted. In this case, the READY line will be pulled low, will remain in that state until execution is complete, and will then be raised to permit completion of the stack access.
3. The 8231A is not busy, and data removal has been requested. READY will be pulled low for the length of time necessary to transfer the byte from the top of stack to the interface latch, and will then go high, indicating availability of the data.
4. The 8231A is not busy, and a data entry has been requested. READY will be pulled low for the length of time required to ascertain if the preceding data byte, if any, has been written to the stack. If so READY will immediately go high. If not, READY will remain low until the interface latch is free and will then go high.
5. When a status read has been requested, READY will be pulled low for the length of time necessary to transfer the status to the interface latch, and will then be raised to permit completion of the status read. Status may be read whether or not the 8231A is busy.

When READY goes low, the APU expects the bus control signals present at the time to remain stable until READY goes high.

DEVICE STATUS

Device status is provided by means of an internal status register whose format is shown below:



Busy: Indicates that 8231A is currently executing a command (1 = Busy)

Sign: Indicates that the value on the top of stack is negative (1 = Negative)

Zero: Indicates that the value on the top of stack is zero (1 = Value is zero)

Error Code: This field contains an indication of the validity of the result of the last operation. The error codes are:

0000—No error

1000—Divide by zero

0100—Square root or log of negative number

1100—Argument of inverse sine, cosine, or e^x too large

XX10—Underflow

XX01—Overflow

Carry: Previous operation resulted in carry or borrow from most significant bit. (1 = Carry/Borrow, 0 = No Carry/No Borrow).

If the BUSY bit in the status register is a one, the other status bits are not defined; if zero, indicating not busy the operation is complete and the other status bits are defined as given above.

READ STATUS

The 8231A status register can be read by the CPU at any time (whether an operation is in progress or not) by bringing the chip select ($\overline{CS}$) low, the command/data line (A_0) high, and lowering $\overline{RD}$. The status register is then gated onto the data bus and may be input by the CPU.

EXECUTION TIMES

Timing for execution of the 8231A command set is contained below. All times are given in terms of clock cycles. Where substantial variation of execution times is possible, the minimum and maximum values are quoted; otherwise, typical values are given. Variations are data dependent.

Total execution times may require allowances for operand transfer into the APU, command execution, and result retrieval from the APU. Except for command execution, these times will be heavily influenced by the nature of the data, the control interface used, the speed of memory, the CPU used, the priority allotted to DMA and interrupt operations, the size and number of operands to be transferred, and the use of chained calculations, etc.

3

DERIVED FUNCTION DISCUSSION

Computer approximations of transcendental functions are often based on some form of polynomial equation, such as:

$$F(X) = A_0 + A_1X + A_2X^2 + A_3X^3 + A_4X^4 \ldots \quad (1\text{-}1)$$

Table 5. Command Execution Times

Command Mnemonic	Clock Cycles	Command Mnemonic	Clock Cycles	Command Mnemonic	Clock Cycles	Command Mnemonic	Clock Cycles
SADD	17	FADD	54-368	LN	4298-6956	POPF	12
SSUB	30	FSUB	70-370	EXP	3794-4878	XCHS	18
SMUL	84-94	FMUL	146-168	PWR	8290-12032	XCHD	26
SMUU	80-98						
SDIV	84-94	FDIV	154-184	NOP	4	XCHF	26
DADD	21	SORT	800	CHSS	23	PUPI	16
DSUB	38	SIN	4464	CHSD	27		
DMUL	194-210	COS	4118	CHSF	18		
DMUU	182-218						
DDIV	208	TAN	5754	PTOS	16		
FIXS	92-216	ASIN	7668	PTOD	20		
FIXD	100-346	ACOS	7734	PTOF	20		
FLTS	98-186	ATAN	6006	POPS	10		
FLTD	98-378	LOG	4474-7132	POPD	12		

The primary shortcoming of an approximation is this form is that it typically exhibits very large errors when the magnitude of |X| is large, although the errors are small when |X| is small. With polynomials in this form, the error distribution is markedly uneven over any arbitrary interval.

A set of approximating functions exists that not only minimizes the maximum error but also provides an even distribution of errors within the selected data representation interval. These are known as Chebyshev Polynomials and are based upon cosine functions. These functions are defined as follows:

$$T_n(X) = \text{Cos } n\theta; \text{ where } n = 0, 1, 2 \ldots \quad (1\text{-}2)$$

$$\theta = \text{Cos}^{-1} X$$

The various terms of the Chebyshev series can be computed as shown below:

$$T_0(X) = \text{Cos } (0 \times \theta) = \text{Cos } (0) = 1 \quad (1\text{-}4)$$

$$T_1(X) = \text{Cos } (\text{Cos}^{-1} X) = X \quad (1\text{-}5)$$

$$T_2(X) = \text{Cos } 2\theta = 2\text{Cos}^2\theta - 1 = 2\text{Cos}^2 (\text{Cos}^{-1} X) - 1 = 2X^2 - 1 \quad (1\text{-}6)$$

In general, the next term in the Chebyshev series can be recursively derived from the previous term as follows:

$$T_n(X) = 2X [T_{n-1}(X)] - T_{n-2}(X); n \geq 2 \quad (1\text{-}7)$$

Common logarithms are computed by multiplication of the natural logarithm by the conversion factor 0.43429448 and the error function is therefore the same as that for natural logarithm. The power function is realized by combination of natural log and exponential functions according to the equation:

$$X^Y = e^{Y \text{Ln} x}.$$

The error for the power function is a combination of that for the logarithm and exponential functions.

Each of the derived functions is an approximation of the true function. Thus the result of a derived function will have an error. The absolute error is the difference between the function's result and the true result. A more useful measure of the function's error is relative error (absolute error/true result). This gives a measurement of the significant digits of algorithm accuracy. For the derived functions except LN, LOG, and PWR the relative error is typically 4×10^{-7}. For PWR the relative error is the summation of the EXP and LN errors, 7×10^{-7}. For LN and LOG, the absolute error is 2×10^{-7}.

APPLICATION INFORMATION

The diagram in Figure 4 shows the interface connections for the APU with operand transfers handled by an 8237 DMA controller, and CPU coordination handled by an Interrupt Controller. The APU interrupts the CPU to indicate that a command has been completed. When the performance enhancements provided by the DMA and Interrupt operations are not required, the APU interface can be simplified as shown in Figure 3. The 8231A APU is designed with a general purpose 8-bit data bus and interface control so that it can be conveniently used with any general 8-bit processor.

In many systems it will be convenient to use the microcomputer system clock to drive the APU clock input. In the case of 8080A systems it would be the ϕ2TTL signal. Its cycle time will usually fall in the range of 250 ns to 1000 ns, depending on the system speed.

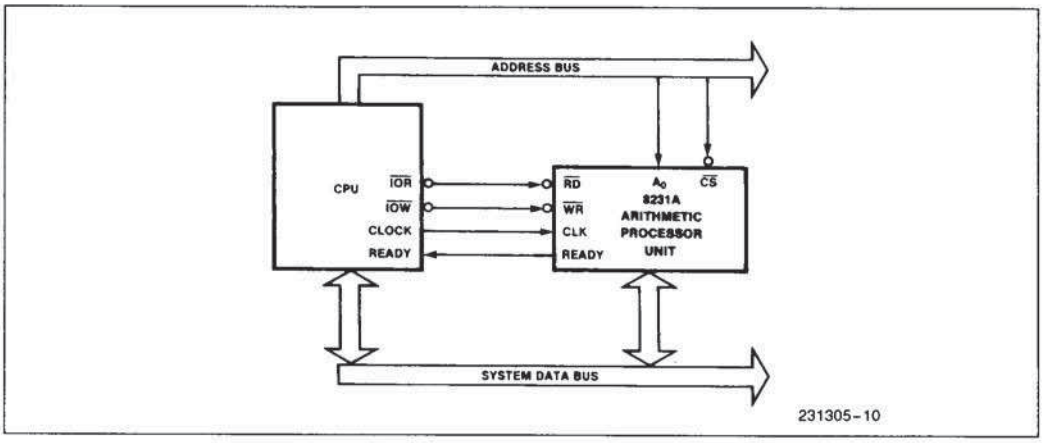


Figure 3. Minimum Configuration Example

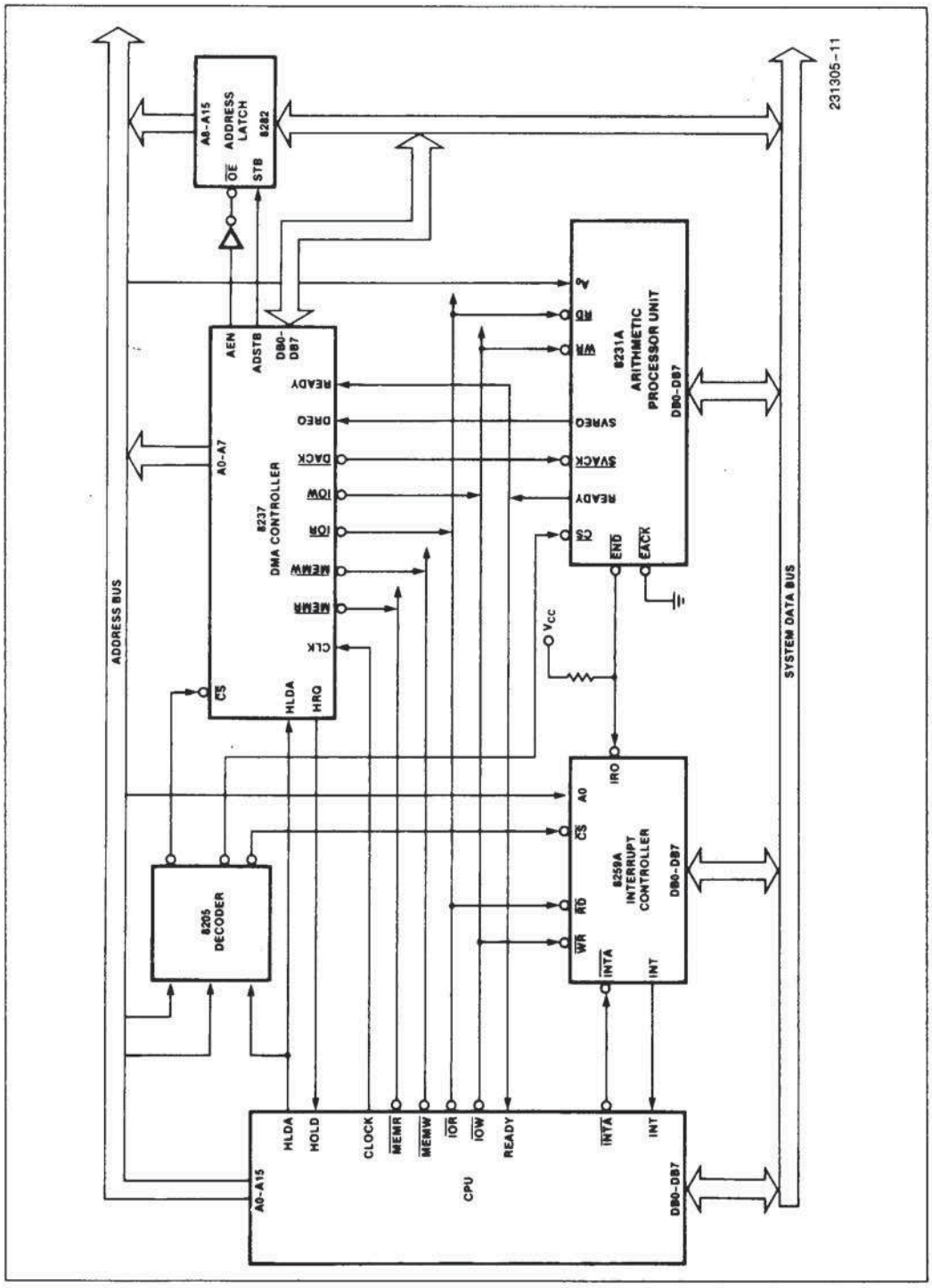


Figure 4. High Performance Configuration Example

ABSOLUTE MAXIMUM RATINGS*

Storage Temperature ..... −65°C to +150°C
Ambient Temperature Under Bias .....0°C to 70°C
V_{DD} with Respect to V_{SS} ..... −0.5V to +15.0V
V_{CC} with Respect to V_{SS} ..... −0.5V to +7.0V
All Signal Voltages with Respect to V_{SS} ..... −0.5V to +7.0V
Power Dissipation.....2.0W

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

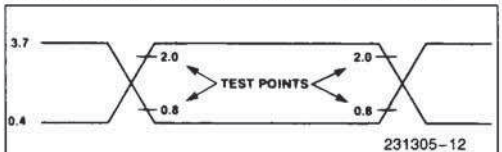
D.C. AND OPERATING CHARACTERISTICS

T_A = 0°C to 70°C, V_{SS} = 0V, V_{CC} = +5V ±10%, V_{DD} = +12V ±10%

Parameters	Description	Min	Typ	Max	Units	Test Conditions
V_{OH}	Output HIGH Voltage	3.7			V	I_{OH} = −200 μA
V_{OL}	Output LOW Voltage			0.4	V	I_{OL} = 3.2 mA
V_{IH}	Input HIGH Voltage	2.0		V_{CC}	V	
V_{IL}	Input LOW Voltage	−0.5		0.8	V	
I_{IL}	Input Load Current			±10	μA	$V_{SS} \le V_{IN} \le V_{CC}$
I_{OFL}	Data Bus Leakage			±10	μA	V_{SS} +0.45 ≤ $V_{OUT} \le V_{CC}$
I_{CC}	V_{CC} Supply Current		50	95	mA	
I_{DD}	V_{DD} Supply Current		50	95	mA	
C_O	Output Capacitance		8		pF	fc = 1.0 MHz, Inputs = 0V(1)
C_I	Input Capacitance		5		pF	
C_{IO}	I/O Capacitance		10		pF	

NOTE:
1. Sampled, not 100% tested.

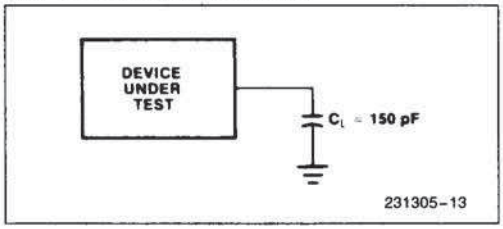
A.C. TESTING INPUT, OUTPUT WAVEFORM



231305-12

A.C. Testing: Inputs are driven at 3.7V for a logical "1" and 0.4V for a logic "0". Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0".

A.C. TESTING LOAD CIRCUIT



231305-13

A.C. CHARACTERISTICS T_A = 0°C to 70°C, V_{SS} = 0V, V_{CC} = +5V ±10%, V_{DD} = +12V ±10%

READ OPERATION

Symbol	Parameter		8231A-8		8231A		Units
			Min	Max	Min	Max	
t_{AR}	A_0, $\overline{CS}$ Setup to $\overline{RD}$		0		0		ns
t_{RA}	A_0, $\overline{CS}$ Hold from $\overline{RD}$		0		0		ns
t_{RY}	READY ↓ from $\overline{RD}$ ↓ Delay (Note 2)			150		100	ns
t_{YR}	Ready ↑ to $\overline{RD}$ ↑		0		0		ns
t_{RRR}	READY Pulse Width (Note 3)	Data	3.5 t_{CY} + 50		3.5 t_{CY} + 50		ns
		Status	1.5 t_{CY} + 50		1.5 t_{CY} + 50		ns
t_{RDE}	Data Bus Enable from $\overline{RD}$ ↓		50		50		ns
t_{DRY}	Data Valid to READY ↑		0		0		ns
t_{DF}	Data Float after $\overline{RD}$ ↑		50	200	50	100	ns

WRITE OPERATION

Symbol	Parameter		8231A-8		8231A		Units
			Min.	Max.	Min.	Max.	
t_{AW}	A_0, $\overline{CS}$ Setup to $\overline{WR}$		0		0		ns
t_{WA}	A_0, $\overline{CS}$ Hold after $\overline{WR}$		60		25		ns
t_{WY}	READY ↓ from $\overline{WR}$ ↓ Delay (Note 2)			150		100	ns
t_{YW}	READY ↑ to $\overline{WR}$ ↑		0		0		ns
t_{RRW}	READY Pulse Width (Note 4)			50		50	ns
t_{WI}	Write Inactive Time (Note 4)	Command	4 t_{CY}		4 t_{CY}		ns
		Data	5 t_{CY}		5 t_{CY}		ns
t_{DW}	Data Setup to $\overline{WR}$		150		100		ns
t_{WD}	Data Hold after $\overline{WR}$		20		20		ns

3

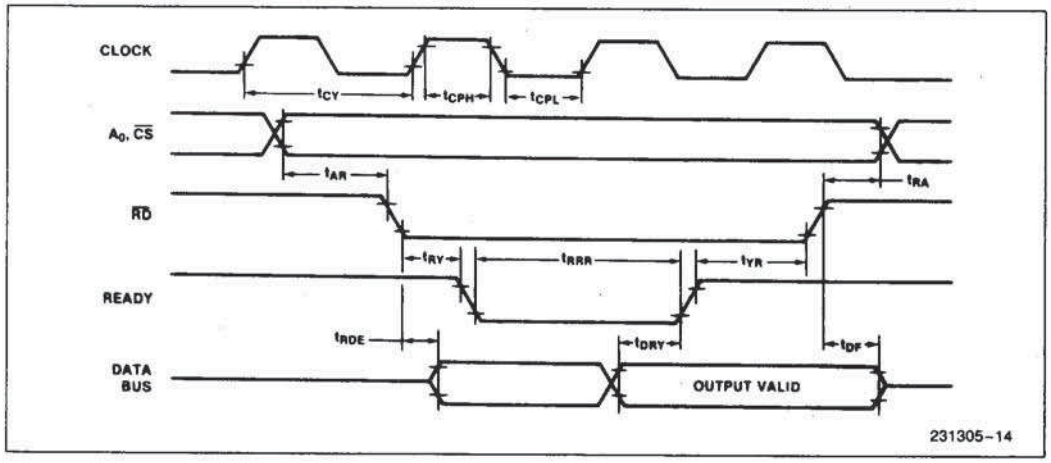
OTHER TIMINGS

Symbol	Parameter	8231A-8		8231A		Units
		Min	Max	Min	Max	
t_{CY}	Clock Period	480	5000	250	2500	ns
t_{CPH}	Clock Pulse High Width	200		100		ns
t_{CPL}	Clock Pulse Low Width	240		120		ns
t_{EE}	$\overline{END}$ Pulse Width (Note 5)	400		200		ns
t_{EAE}	$\overline{EACK}$ ↓ to $\overline{END}$ ↑ Delay		200		150	ns
t_{AA}	$\overline{EACK}$ Pulse Width	100		50		ns
t_{SA}	$\overline{SVACK}$ ↓ to SVREQ ↓ Delay		300		150	ns
t_{SS}	$\overline{SVACK}$ Pulse Width	100		50		ns

NOTES:
1. Typical values are for T_A = 25°C, nominal supply voltages processing parameters.
2. READY is pulled low for both command and data operations.
3. Minimum values shown assume no previously entered command is being executed for the data access. If a previously entered command is being executed, READY low pulse width is the time to complete execution plus the time shown. Status may be read at any time without exceeding the time shown.
4. READY low pulse width is less than 50 ns when writing into the data port or the control port as long as the duty cycle requirement (t_{WI}) is observed and no previous command is being executed. t_{WI} may be safely violated as long as the extended t_{RRW} that results is observed. If a previously entered command is being executed, READY low pulse width is the time to complete execution plus the time shown. These timings refer specifically to the 8231A.
5. $\overline{END}$ low pulse width is specified for $\overline{EACK}$ tied to VSS. Otherwise t_{EAE} applies.

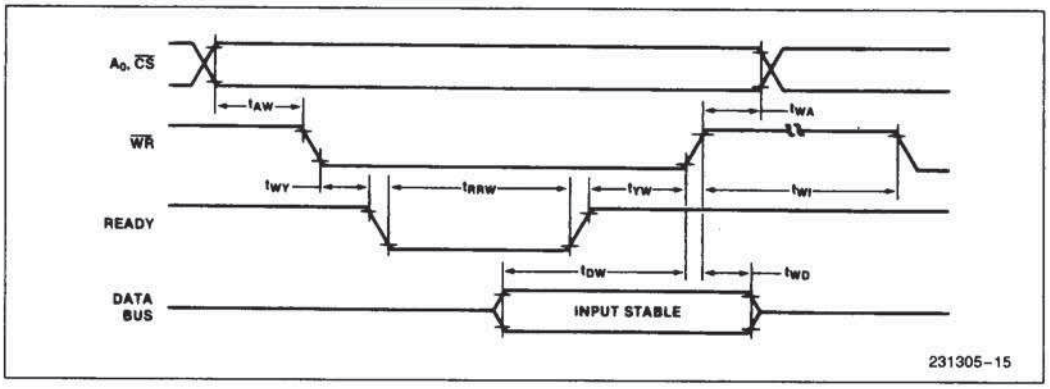
WAVEFORMS

READ OPERATION



3

WRITE OPERATION



INTERRUPT OPERATION