February 1976

Dear Sir:

Here is our latest newsletter updating you on our activities including new products, more detailed pricing and information on various items raised by our customers over the past few months. We are now delivering TIM chips (MCS6530-004) in volume and a special "ROM-less" MCS6530 (RAM, I/O and timer only, the MCS6530-005) in prototype systems where the mask-programmed ROM is not required.

You have recently received a brochure on our KIM-1 Microcomputer System and we have referenced that offering in both our Price List and Order Form in this mailing. We are indeed gratified by the response to this product by the marketplace.

MOS TECHNOLOGY, INC. continues to deliver the lowest cost and fastest 8 bit Microprocessor on the market. Synertek, our second source, will very shortly be delivering this product as well. We are also delivering 2 MHz microprocessors, identifiable on the price list by our "A" suffix.

Included in this newsletter are the following:

1. Price List - indicating low volume price and delivery of the microcomputer products.

2. TIM Program Description - providing a basic description of the features found in the pre-programmed MCS6530-004.

3. MDT System Description - providing a basic description of the Microcomputer Development Terminal. This sophisticated but easy to use system development tool will be available in the second quarter of 1976.

4. Handling to prevent static damage - some comments regarding recommended care in handling MOS TECHNOLOGY, INC. products.

5. Single Cycle/Single Instruction Schematic - several of our customers have brought to our attention that our Static Test Control Logic Schematic on page 125 of the Hardware Manual is incorrect. We regret the delays this may have caused any of our customers who attempted to use this circuitry. The enclosed schematic will allow you to perform Single Cycle and Single Instruction executions and should be used in place of that found in the Hardware Manual.
6. New Order Form

7. Discussion of MCS650X Circuit Modifications

8. Discussion of Crystal and RC Time Base Generation.

9. Listing of MOS TECHNOLOGY, INC. Sales Representatives.

10. UCS Timesharing Systems Brochure for the MCS650X product line.

Our next newsletter will introduce some of our new products which will be coming out in the remainder of 1976. In the mean time, we will continue to augment our customer support activities both in the factory and in the field to keep pace with our rapidly growing customer base.

Very truly yours,

MOS TECHNOLOGY, INC.

[Signature]

Charles I. Peddle
Marketing Director
Microcomputers

CIP/nac
Encl.

If your name or address is incorrect or if you are receiving duplicate mailings please return this portion with the correct information.

NAME_____________________________________

COMPANY__________________________________

ADDRESS__________________________________

__________________________________________

This is a duplicate mailing__________
<table>
<thead>
<tr>
<th>PART NUMBER</th>
<th>1 - 99</th>
<th>100 - 999</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCS6501 - 1 MHz</td>
<td>$20.00</td>
<td>$18.00</td>
<td></td>
</tr>
<tr>
<td>MCS6502 - 1 MHz</td>
<td>25.00</td>
<td>21.00</td>
<td></td>
</tr>
<tr>
<td>MPS6503 - 1 MHz</td>
<td>20.00</td>
<td>18.00</td>
<td>Off the shelf</td>
</tr>
<tr>
<td>MPS6504 - 1 MHz</td>
<td>20.00</td>
<td>18.00</td>
<td></td>
</tr>
<tr>
<td>MPS6505 - 1 MHz</td>
<td>20.00</td>
<td>18.00</td>
<td></td>
</tr>
<tr>
<td>MCS6502 A*</td>
<td>37.50</td>
<td>31.50</td>
<td></td>
</tr>
<tr>
<td>MPS6503 A</td>
<td>30.00</td>
<td>27.00</td>
<td>Off the shelf</td>
</tr>
<tr>
<td>MPS6504 A</td>
<td>30.00</td>
<td>27.00</td>
<td></td>
</tr>
<tr>
<td>MPS6505 A</td>
<td>30.00</td>
<td>27.00</td>
<td></td>
</tr>
<tr>
<td>MCS6530-004 (TIM)</td>
<td>30.00</td>
<td>26.00</td>
<td>Off the shelf</td>
</tr>
<tr>
<td>(Includes TIM Manual)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6530-005</td>
<td>18.00</td>
<td>16.00</td>
<td>Off the shelf</td>
</tr>
<tr>
<td>(No ROM Available)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6530 - Custom Program**</td>
<td>30.00</td>
<td>26.00</td>
<td>8 - 10 weeks after receipt of order</td>
</tr>
<tr>
<td>KIM-1 System</td>
<td>$245.00/system</td>
<td></td>
<td>Off the shelf</td>
</tr>
<tr>
<td>Postage and Handling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$4.50 - Domestic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.00 - International</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* "A" suffix imples 2 MHz product

** Mask tooling for custom programs is $1,000.00 with a minimum purchase quantity of 50 units. The $1,000.00 will be refunded if more than 1,000 units of the unique pattern is purchased within the first 12 months.

Questions concerning large volume price and delivery on all products should be directed to Mr. Julius C. Hertsch, Product Manager, MOS TECHNOLOGY, INC. (215-666-7950 x 220).
TIM is the Terminal Interface Monitor program for MOS TECHNOLOGY, INC.'s MCS650X microprocessors. It is supplied in read-only memory (ROM) as part of the MCS6530 multi-function chip. Because the TIM code is non volatile, it is available at system power-on and cannot be destroyed inadvertently by user programs. Furthermore, the user is free to selectively use only those TIM capabilities which he needs for a particular program. Both interrupt types, interrupt request (IRQ) and non-maskable interrupt (NMI) may be set to transfer control to TIM or directly to the user's program.

TIM communicates with the user via a serial full-duplex port (using ASCII codes) and automatically adjusts to the speed of the user's terminal. Any speed... even non-standard ones... can be accommodated. If the user's terminal has a long carriage return time, TIM can be set to perform the proper delay. Commands typed at the terminal can direct TIM to start a program, display or alter registers and memory locations, set breakpoints, and load or punch programs. If available in the system configuration, a high-speed paper tape reader may be used to load programs through a parallel port on the MCS6530 chip. Programs may be punched in either of two formats -- hexadecimal (assembler output) of BNPF (which is used for programming read-only memories). On loading or modifying memory, TIM performs automatic read-after-write verification to insure that addressed memory exists, is read-write type, and is responding correctly. Operator errors and certain hardware failures may thus be detected using TIM.

TIM also provides several subroutines which may be called by user programs. These include reading and writing characters on the terminal, typing a byte in hexadecimal, reading from high-speed paper tape, and typing a carriage-return, line-feed sequence with proper delay for the carriage of the terminal being used. Program tapes loaded by TIM may also specify a start address so that programs may be started with a minimum of operator action.

TIM is normally entered when a 'BRK' instruction is encountered during program execution. At that time CPU registers are output: * PC F A X Y S and control is given to the terminal.

Note:  PC = Program Counter
       F = Processor Status Register
       A = Accumulator
       X = Index Register, X
       Y = Index Register, Y
       S = Stack Pointer

In summary, TIM's features include the following capabilities:

- Self adapting to any terminal speed for 10-30 cps
- Display and Alter CPU registers
- Display and Alter Memory locations
- Read and Write/Punch hex formatted data
Write/Punch BNPF format data for PROM programmers

Unlimited breakpoint capability

Separate non-maskable interrupt entry and identification

External device interrupts directable to any user location or defaulted to TIM recognition

Capability to begin or resume execution at any location in memory

Completely protected, resident in Read-Only Memory

Capability to bypass TIM entirely to permit full user program control over system

High speed 8-bit parallel input option

User callable I/O subroutines

The commands used for directing TIM to perform these functions have been held to a minimum. This means that TIM is easily learned and readily remembered. TIM’s Command Set consists of:

.R Display registers (PC, F, A, X, Y, S)

.M ADDR Display memory (8 bytes beginning at ADDR)

.: DATA Alters perviously displayed item

.LH Load Hexadecimal tape

.WB ADDR1, ADDR2 Write BNPF tape (from ADDR1 to ADDR2)

.WH ADDR1, ADDR2 Write hex tape (from ADDR1 to ADDR2)

.G Go, continue execution from current PC address

.H Toggles high-speed-reader option (if its on, turns it off; if off, turns on)

TIM is offered in the form of a 1K x 8 program resident in the MCS6530-004 at a 1 to 99 price of $30.00

The TIM unit comes with:

1 - MCS6530-004 Multi-function Chip
1 - TIM Users Manual
1 - TIM System Schematic
* Note that the TIM as sold consists only of the MCS6530-004 component accompanied by supporting information to build this system.

TYPICAL MINIMUM CONFIGURATION FOR "TIM" SYSTEM
The MDT650 is a high level development tool for modeling new designs. New flexible modular techniques of system verification are provided the user prior to a "hard" finalized design commitment. Fully programmable user control is among the many outstanding features. Interactive design allows the user to assemble programs, debug software with on-line editing capability and, when the design is considered correct, program PROMs. The MDT can be interfaced through the integral keyboard/display or via a port for TTY or higher speed peripherals. The standard resident assembler, in conjunction with systemized option cards for interactive debugging, allow the user to engage this system as a total development tool for preproduction and final production systems.

GENERAL SYSTEM DESCRIPTION

The MDT650 series microcomputer development terminal evaluates and debugs the user's programs and system hardware. The unit can be configured to a wide range of design applications. Considerable development time and money may be saved through the unit's unique ability to allow user-system emulation. The MDT650 provides the user a completely separate processor and bus structure to configure his application without regard to memory or executive I/O functions. This eliminates executive overhead time during real time execution...particularly important for interrupt routines.

Two MCS650X series microprocessors are used to control all system functions. Interaction with the MDT650 is normally with the integral keyboard/display. A TTY or other terminal device can also be used. Expandable port configurations are TTL compatible.

The standard MDT650 system allows the user to assign up to 65K of memory as desired (with independent address and data bases). The ROM resident system monitor includes all necessary functions for program loading, debugging, and execution. A resident assembler may be used to assemble machine instructions. Interpretation of machine codes is language to original OP codes, labels and mnemonics. A resident editor provides source language editing capability.

HARDWARE FEATURES

Integral Keyboard with separate function keys for control.

Built-in 32 character display.

Terminal inputs for variety of terminals. Selectable BAUD rates of 110, 150, 300, 600, 1200, 2400, 4800 or 9600 are also provided.

Two address traps are provided to halt the user processor on: any address; instruction (operand fetch); read cycle or write cycle

Two scope syncs: address trap sync and instruction fetch (operand)cycle sync.

Single instruction mode with firmware enhancement.
HARDWARE FEATURES CONTINUED

Trace stack memory for storing the last 128 machine cycles. (Visually displayed using the disassembler firmware).

Seven board positions provided for user memory, bus light display, I/O or user options such as custom wire wrap boards.

Control firmware for the assembler, disassembler, and test editor that is independent of the user's 65K memory limit.

SOFTWARE SUPPORT

The MDT650 software consists basically of three straightforward and highly useful programs: the assembler, the disassembler and the text editor.

The MDT assembler is upwards compatible with the MCS650X Cross-Assembler. Features include:

- Can assemble from source tape or user RAM... eliminates having to feed the source tape through the reader two times.
- Six character labels and symbols.
- Free-form entry of source statements.
- Symbol table output.
- Error flags on listing.
- Assembled program is user memory ready for execution.
- Paper tape output option - compatible with loader

The MDT disassembler commands include:

Note: For maximum user control, most of the following commands have corresponding function keys provided on the integral keyboard. Additionally, the MDT has function keys to show register, program counter, accumulator, index and status registers, stack pointer, and either trap address; also provided are function keys to alter individual registers...display shows what register is being altered, and a selectable enable/disable switch for TTY printing.

- Display address. 8 bytes are displayed and cursor is positioned at start of first byte to allow it to be altered. If you want to alter, just start typing. The display is automatically changed after each two input characters (one byte of memory) so multiple alters are easily made. (Function keys provide the capability of displaying forward by one or eight characters.)
- Load interface file (symbols and code).
- Go to address. Execute one instruction and automatically disassemble.
- Run. Executes user program.
SOFTWARE SUPPORT CONTINUED

- Trap address and mode. Sets appropriate instruction.
- Forward one instruction from current location in stack.
- Backward one instruction.
- Show last/next cycle. Address, label at this address (if any), data, and decoded flags are displayed.

The MDT test editor program saves considerable money and frustration compared with trying to edit a paper tape by hand. The following capabilities are provided:

- Loads text buffer
- Insert or delete line
- Insert or delete character at current position in line
- Forward/backward one character
- Will step forward or backward to next blank (Very useful for jumping from label field to OP Code field, etc.)
- Goes to top or bottom of text
- Indexes up or down one line
- Search/find capability on first field
- Provides output text to printer/punch

STANDARD EQUIPMENT (BASE SYSTEM)

Dual Micro Processor Module
RAM Memory Module
Program trace and address trap board
  I/O board for Keyboard, Display and Peripherals
Resident Monitor ROM Module
Chassis with 14 Board Slots
Power Supplies
Finished Cabinet
Keyboard and Display
System Monitor
Assembler
Text Editor
User's Manual
MCS650 Assembly Language Programming Manual
MCS650 Assembly Language Reference Card
OPTIONS

PROM programmer available for 82S115, 2708 or 1702A.
Wire-wrap boards for custom designs.
Extender board module

Light display board to display address and data busses
4K RAM board
8K PROM board
2K RAM/4K PROM board
I/O board...allows interfacing system to various peripheral devices and terminals
Write protect available on user RAM
High speed ports for printer, card readers, etc. (and associated cables).
Floppy disc interface will be available at a later date
ICE (In-Circuit Emulator)
Source Listing

AVAILABILITY

The base system will be available in second quarter, 1976 with the various Hardware options becoming available in second and third quarters, 1976.

COST

$3,950.00 (Base System). Cost of options will be available at a later date.
PRECAUTIONARY HANDLING PROCEDURES
FOR "MOS" TYPE PRODUCTS

The MCS6500 Product Line has been designed with protective circuitry to guard against static charge damage on the inputs. However, normal precautions should be taken whenever possible to prevent exposure to environments of potential static charge. The following guidelines are recommended in handling the "MOS" type products and should be used whenever possible:

* Keep devices in the conductive shipping carrier until used.

* Perform work involving the "MOS" devices on a conductive surface where possible.

* In board assembly, place the "MOS" devices on the boards as late in the assembly cycle as possible. For low volume applications we recommend usage of a plug-in socket for the devices.

* Do not place the "MOS" device into position with power on. Always power up after the device is in place in the board assembly.
SINGLE CYCLE / SINGLE INSTRUCTION
CONTROL LOGIC
ORDER FORM

COMPONENTS

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>QUANTITY</th>
<th>PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCS6501 @ $20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6502 @ $25.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6503 @ $20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6504 @ $20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6505 @ $20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6502A @ $37.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6503A @ $30.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6504A @ $30.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPS6505A @ $30.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6530-004 @ $30.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Includes TIM Manual)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS6530-005 @ $18.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KIM-1 System @ $245.00 *</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Includes all documentation)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*Postage and Handling $4.50 - Domestic $20.00 - International</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DOCUMENTATION

<table>
<thead>
<tr>
<th>DOCUMENTATION</th>
<th>QUANTITY</th>
<th>PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Manual (Pub. No. 6500-10) @ $5.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming Manual (Pub. No. 6500-50) @ $5.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cross Assembler Manual @ $4.00 (Preliminary)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIM Manual (If purchased separately from the MCS6530-004) @ $4.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOTAL

NAME ___________________________ Purchase Order Encl. ________
COMPANY _________________________ Check Enclosed ____________
ADDRESS _________________________ ___________________________
MCS650X FAMILY CIRCUIT MODIFICATIONS

1. Since introduction of the MCS650X Family numerous customers have requested the addition of the ROR Instruction. This instruction is now being added to all MCS650X processors. The addressing modes will be Absolute (6 cycles, 3 bytes); Zero Page (5, 2); Accumulator (2, 1); Zero Page, X (6, 2); Absolute, X (7, 3). The implementation of ROR involves shifting all addressed locations one bit to the right with the carry bit shifted into Bit location 7 and Bit location 0 shifted into the carry position.

2. An additional modification is being made to the Branch, Ready circuitry. The present MCS650X processors do not execute the branch instructions correctly when the ready signal is used in the Single Cycle Mode if the low order effective address is FF without crossing page boundaries. For clarification the following program is executed, in both Single Cycle Mode and Single Instruction Mode.

<table>
<thead>
<tr>
<th>Memory</th>
<th>Contents</th>
<th>Single Cycle</th>
<th>Sample Program Execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>F5</td>
<td>18 CLC</td>
<td>ABH ABL DB</td>
<td>ABH ABL DB</td>
</tr>
<tr>
<td>F6</td>
<td>90 BCC</td>
<td>XX F5 18 1</td>
<td>XX F5 18 1</td>
</tr>
<tr>
<td>F7</td>
<td>offset</td>
<td>XX F6 90 0</td>
<td>XX F6 90 1</td>
</tr>
<tr>
<td>F8</td>
<td>LD A1MM</td>
<td>XX F6 90 1</td>
<td>XX FF (XXFF) 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XX F7 07 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XX F8 09 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>XX FF(XX+1,FF) 1</td>
<td></td>
</tr>
</tbody>
</table>

Note that in the Single Cycle Mode the ADH of the branch destination is incremented while in the Single Instruction Mode the branch is properly executed. The only time this occurs is when the ADL of the branch destination is FF and then only during Single Cycle with no page crossing; hence, the probability of this occurring in normal application is remote.

Availability - MCS650X microprocessors incorporating the above changes will be available in sample quantities in April with production deliveries beginning in May. Pricing for these versions of the microprocessor will be identical to the product currently being shipped.
CLOCK GENERATOR INFORMATION

Initial characterization of the MCS650X clock generator circuit has provided us with sufficient information to update the clock generator information found in our manuals and data sheets. The following discussion provides the user with information needed to obtain best performance for the time base generation scheme chosen.

Generally one would consider the following when designing systems with the MCS650X.

1. Which clock scheme is the best?

There is no one answer, however depending on the system there is one best way

A. TTL generated clock - drive \( \phi_0 \) (IN) with a TTL level clock, it doesn't require a high level clock; merely \( V_{OL} = 0.4V, V_{OH} = 2.4V \). Buffer \( \phi_2 \) (OUT) for use as system \( \phi_2 \) clock. This scheme allows maximum control of all clock variables (i.e. symmetry, frequency, frequency variation from system to system).

In the following discussion, reference will be made to \( \phi_0 \) (IN) and \( \phi_2 \) (OUT). The applicable pin numbers on the various MCS650X processors are found in the manuals or data sheets. The diodes (IN914's) are for the purposes of clamping the clock swings near ground and near \( V_{DD} \) and may not be required in all crystal applications.

B. Series Mode Crystal Controlled

\[
\begin{align*}
\phi_2 \text{ (OUT)} & \quad \uparrow \quad 7404 \quad \downarrow \quad \text{System } \phi_2 \\
\phi_0 \text{ (IN)} & \quad \uparrow \quad R_f \quad \downarrow \quad C_f = 10 \text{ pf} \\
& \quad \uparrow \quad \text{CF} \quad \downarrow \quad \text{RF} = 330K\Omega \\
& \quad \uparrow \quad \text{Vcc} \quad \downarrow \quad \text{crystal} \\
\end{align*}
\]

This scheme allows for crystal controlled operation which is least sensitive to crystal parameters and feedback circuit variables. Because the crystal is in the feedback path and not shunting as the parallel mode crystal controlled scheme, the serial mode is most reliable from a start-up standpoint.

C. Parallel Mode Crystal Controlled

\[
\begin{align*}
\phi_2 \text{ (OUT)} & \quad \uparrow \quad 7404 \quad \downarrow \quad \text{System } \phi_2 \\
\phi_0 \text{ (IN)} & \quad \uparrow \quad R_f \quad \downarrow \quad C_f = 10 \text{ pf} \\
& \quad \uparrow \quad \text{CF} \quad \downarrow \quad \text{RF} = 330K\Omega \\
& \quad \uparrow \quad \text{Vcc} \quad \downarrow \quad \text{crystal} \\
\end{align*}
\]
This scheme should be used when the symmetry is more desirable than the Serial Mode crystal controlled scheme symmetry. This scheme is most sensitive to feedback parameters as related to start-up. By varying the feedback resistor an appropriate combination can be found for the crystal chosen.

D. RC Controlled

\[ \varphi_2 \text{ (OUT)} \]

\[ \varphi_0 \text{ (IN)} \]

This scheme is recommended for those systems which do not require symmetry control, frequency variation control from system to system without manual adjustment, and systems where noise has been minimized.

Because of the ease of use of this scheme it is recommended for those systems which are in development, used as a microprocessor learning vehicle or in general systems in which noise on the clock circuit has been carefully handled (i.e. clean supply to microprocessor and clock buffers, isolate \( \varphi_2 \text{ (OUT)} \) and \( \varphi_2 \text{ (IN)} \) from stray system noise).

For frequency of operation around 1 MHz, a value of 10K to 50K should be used with \( C_F = 10 \text{ pf} \). To decrease noise sensitivity increase \( C_F \) and decrease \( R_F \). Also a shunting capacitor to ground from \( \varphi_2 \text{ (OUT)} \) the value of which should be the same range as \( C_F \), will help to decrease sensitivity to noise in the system.

2. What is the maximum clock loading?

One standard TTL Load and 30 pf

If the clock outputs (\( \varphi_1 \text{ (OUT)} \) and \( \varphi_2 \text{ (OUT)} \)) are loaded, there is the possibility of causing overlap, but this has no effect on the internal clocks on the microprocessor. The system designer should therefore be careful if non-overlapping system clocks are necessary such that the microprocessor can function properly with overlapped clocks but system problems can develop.

3. How can clock ringing be prevented?

A. Eliminate noise from \( \varphi_0 \text{ (IN)} \). The clock generator on the MCS650X will react to frequencies as high as 20 MHz, therefore it will also respond to noise.
B. Be sure the negative feedback ($R_f$) occurs after the positive feedback ($C_f$). This is most easily accomplished by tapping off the negative feedback after the positive feedback (note the inverter delay in the RC controlled schematic).

4. What are the typical clock widths and separations for TTL clock levels in? See graph #1.

5. What are typical frequencies for various RC combinations?

The RC values listed below provide the approximate operating frequencies listed. It is recommended that variable resistor pots be used if accuracy in the value of the operating frequency is required.

<table>
<thead>
<tr>
<th>Typical Frequency</th>
<th>$R_f$</th>
<th>$C_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>.5 MH$	ext{z}$</td>
<td>42K</td>
<td>10 pf</td>
</tr>
<tr>
<td>1.0 MH$	ext{z}$</td>
<td>17K</td>
<td>10 pf</td>
</tr>
<tr>
<td>2.0 MH$	ext{z}$*</td>
<td>6K</td>
<td>10 pf</td>
</tr>
</tbody>
</table>

*Applies to product guaranteed at 2 MH$	ext{z}$ operation.

NOTE: It should be understood that for maximum confidence in control of symmetry of the system clocks, it is recommended the TTL level $\phi_o$ (IN) be used. This can be generated from, for example a divide down from a high frequency crystal. In any case maximum pulse width control is formed with these inputs to $\phi_o$ (IN) in which the user has maximum control of the edges.
GRAPH #1
CLOCK PHASE RELATIONSHIPS

NOTE: This graph shows 2 MHz clocks. Clock delays do not change with frequency.

*Typical process related values. Edges marked "A" would correspond to part "A"
Edges marked "B" would correspond to part "B"
SALES OFFICES AND REPRESENTATIVES FOR MOS TECHNOLOGY, INC.

EASTERN REGION

REGIONAL DIRECTOR

William Whitehead
MOS TECHNOLOGY, INC.
410 Jericho Turnpike, Suite 312
Jericho, N.Y. 11753
516-822-4240

NEW ENGLAND STATES

The Orion Group
P. O. Box 248
Mont Vernon, N.H. 03057
617-890-3777

SOUTHERN N.J., EASTERN PA, DELA.

Rivco, Inc.
P. O. Box 338
King of Prussia, Pa. 19406
215-265-5211

MD., WASH, D.C., VA. W. VA.

Bernard White & Company
7 Church Lane
Baltimore, Maryland 21208
301-484-5400

NORTHERN N.J., NEW YORK CITY,
WESTCHESTER COUNTY, N.Y. FAIRFIELD COUNTY, CONN.

Falk-Baker Associates
382 Franklin Ave.
Nutley, N. J. 07110
201-661-2430

ALA, FLA, GA, TENN., N. C., S. C., MISS.

Currie, Peak & Frazier, Inc.
6880 Lake Ellenor Drive
Suite 237
Orlando, Florida 32809
305-855-0843

NEW YORK, EXCEPT NEW YORK CITY
AND WESTCHESTER COUNTY

Labtronics, Inc.
166 Pickard Bldg
Syracuse, N.Y. 13211
315-454-9314
WESTERN REGION

REGIONAL DIRECTOR

Jack Turk
MOS TECHNOLOGY, INC.
2172 Dupont Drive
Patio Bldg.
Irvine, CA. 92715
714-833-1600

APPLICATIONS MANAGER

Petr Sehnal
MOS TECHNOLOGY, INC.
22300 Foothill Blvd.
Hayward, CA. 94541
415-881-8080

ARIZONA, NEW MEXICO

Toward Engineering Assoc., Inc.
P. O. Box 15268
Phoenix, Arizona 85018
602-955-3193

COLORADO, UTAH, WYOMING
MONTANA (EAST OF GREAT FALLS)

R. G. Enterprises, Inc.
1898 So. Flatiron Ct.
Boulder, CO 80301
303-447-9211

WASHINGTON, OREGON, IDAHO,
MONTANA (WEST OF GREAT FALLS)

J. A. Tudor & Associates, Inc.
P. O. Box 21947
Seattle, Washington 98111
206-682-7444

NORTHERN CALIFORNIA, NORTHERN NEVADA

W. W. Posey Company, Inc.
895 Sherwood Ave.
Los Altos, CA. 94022
415-948-7771

SALES OFFICE AND FOREIGN REPRESENTATIVES

EUROPEAN SALES DIRECTOR

Mr. Gerold G. Wiese
MOS TECHNOLOGY, INC.
6382 Friedrichsdorf 2 (F. Frankfurt)
Otto-Hahn-Strasse 40
W. Germany
(06175) 1510, 1519

HONG KONG AND TAIWAN

Mr. Lyle Ronalds
SOLID STATE INTERNATIONAL
Daily House Suite 48
35-39 Haiphong Road
Kowloon, Hong Kong
China

JAPAN AND ITS TERRITORIES

Nihon Teksel Co., Ltd.
13-14 Sakuragaoka-Machi
Shibuya-Ku, Tokyo
Japan
(03) 461-5121

INDIA

Mr. Dinish C. Gupta
Superior Electronics
E13 Dattaguru Society
Deonarpada Road
Bombay 400 088 India
MOS TECHNOLOGY’S support software is now available on United Computing Systems’ time-sharing service. The package available provides online support to assist the microcomputer applications design engineer or programmer in program development for the MCS650X microcomputer family.

TO USE MOS TECHNOLOGY SUPPORT SOFTWARE:

1. Contact your local USC sales representative and request MOS TECHNOLOGY’S MCS650X Software System under user catalog number M490. Also request the UCS System Guide and the UNEDIT manuals.
3. Dial the appropriate telephone number supplied by your USC sales representative, sign on with your terminal, and begin entering your MCS650X microprocessor program.

THE SOFTWARE SUPPORT PACKAGE CONSISTS OF:

-MOS/*** - A text file containing the latest bulletins regarding MOS TECHNOLOGY Microprocessor Software.

-ASM/*** - An interactive program which builds the job control language required to submit your source code to ASM650X.

ASM650X - MCS650X Cross Assembler: the Cross Assembler is a program which translates a mnemonic or symbolic form of a computer program to machine language.

-SIM/*** - An interactive program which builds the job control language required to submit your simulator command file to SIM650X.

SIM650X - MCS650X Simulator. The simulator uses the command file to simulate execution of the machine language instructions created by the cross assembler in the MCS650X microprocessor.
-DMP/*** - ROM dump program. This program creates an output file of machine language instructions in a format suitable for MOS microcomputer loader programs.

The sample program shown in this brochure uses the UCS time-sharing system to give the user an overview of the procedure to be followed for using MOS TECHNOLOGY’S support software.

In brief the procedure to be followed is:

1. Create a source file using the time-sharing editor and save the file.
2. Submit the source file to the Cross Assembler by answering the questions asked by -ASM/***.
3. When the Cross Assembler run is completed list the output file to obtain a listing of the assembled program.
4. Create a file of simulator commands using the time-sharing editor and save the file.
5. Submit the simulator command file and the machine language file to the simulator by answering the questions asked by -SIM/***.
6. When the simulator run is completed list the output file to obtain the results of the program simulation.
7. Obtain a ROM dump object tape by answering the questions asked by -DMP/***.
1. CREATE A SOURCE FILE.

Enter proper response so that computer can determine your terminal's speed.
For 10 CPS enter 763
For 15 CPS enter 863
For 30 CPS enter 163

Enter your user number and password to log on to UCS system.
Indicates FORTRAN system is ready. (FORTRAN is automatically assigned.)
Enter -MOS/*** to obtain latest bulletins.

Indicates the end of the bulletin.
Create a new file with file name “SAMP4”.
Auto line number assignment.
Assembler directive to advance listing to top of page and title the page “MULTIPLE BYTE ADD”.
Semicolon indicates the start of a comment field.
=* assembler directive sets the program counter.
Sets NB equal to 8.
Reserves 8 bytes of memory for the label “PP”.
Start of program labeled “MAIN”
Note that there is only one space between a line number and a label. There are two or more spaces between a line number and an instruction. Comments may begin one space after the operand.

. END assembler directive defines the end of the source program.
Hitting the “ESC” key ends the auto line number assignment. The system replies “**DEL**”.
SAVE is the command to save the new file just created.

2. SUBMIT TO CROSS ASSEMBLER.

-ASM/*** invokes the cross assembler submitter software.

SOURCE file is the file containing the source code to be assembled.
OUTPUT file will contain the assembler listing.
INTERFACE file will contain the object code, line number and label information required by the simulator.
ERROR file will contain a listing of any errors that occur during the assembly.
DAY file is a history of steps taken by the UCS system in running your job.
CONTROL file is the file of JCL built by -ASM/*** to run your assembly.
Submits assembly job to the UCS system.
Indicates that the job has been submitted under the job name “RJEDZOM”.

-ASM/***
3. LIST OUTPUT FILE

Terminal input to list the output file “OUT4”.

Title created by PAGE assembler directive.

Program counter. (Hexadecimal)

Hexadecimal instruction, data, or value.

Program counter set to hexadecimal 64 by assembler directive "=100.

Error line will also appear in the ERROR file.

The version number is changed as improvements are made to the Cross Assembler.

Note: For more detailed information refer to the MCS6500 Microprocessor Programming and Cross Assembler manuals.

4. CREATE SIMULATOR COMMANDS

Create simulator command file called “ECSAMP1”.

Starting at location 1 set consecutive memory locations to the specified values.

Dump the contents of memory from decimal 1 to hexadecimal 18.

Trace every instruction executed.

Begin simulated execution at label “MAIN” and continue until instruction at label “NEXT” has been executed 3 times.

EXIT terminates simulator run.

-SIM/*** invokes the simulator submitter software.

COMMAND file is the file containing the simulator commands.

INTERFACE file is the interface file created by the cross assembler.

5. SUBMIT TO SIMULATOR

-SIM/***

MOS TECHNOLOGY 65KX SIMULATOR SUBMITTER

DO YOU WANT INSTRUCTIONS (YES OR NO) -- Y NO
ENTER USER/NO. PASSWORD, AND PID (IF NEEDED) -- ? M49001# EXAMPLE
DO YOU WANT TO CHANGE THE PRIORITY -- ? NO

ENTER COMMAND FILE NAME -- ? ECSAMP1

ENTER INTERFACE FILE NAME -- ? INT4

SAVE OUTPUT FILE (YES OR NO) -- ? YES
ENTER OUTPUT FILE NAME -- ? OUT4

SAVE DAYFILE FILE (YES OR NO) -- ? YES
ENTER DAYFILE FILE NAME -- ? DAY4

ENTER CONTROL FILE NAME -- ? CON4

TO RUN SIMULATOR TYPE -- OLD.ECON4

RJE (OR RRE)

STOP.

OLD.ECON4

READY - EXE1

RJE

11/19/75 89.23.50.

PROGRAM ECON4

RJE COMPLETE.ID = RJECHR
6. LIST SIMULATOR OUTPUT

Terminal commands required to list the Simulator output file.

;Dump ADDR=0010  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
;Dump ADDR=0018  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
;Dump ADDR=0020  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
;Dump ADDR=0028  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
;Dump ADDR=0030  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
;Dump ADDR=0038  01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

7. PUNCH OBJECT TAPE

-DMP/**

-MOS TECHNOLOGY -- ROM DUMP

INTERFACE file is the file created by the cross assembler.

OBJECT file is the file name the object code is to be saved in.

Terminal commands required to list and punch the object tape.

Note: The paper tape punch should be turned on after the carriage return is entered.

Sign-off the system by entering "BYE"
### MCS6500 MICROPROCESSOR LANGUAGE

#### INSTRUCTION SET

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>Add Memory to Accumulator with Carry</td>
</tr>
<tr>
<td>AND</td>
<td>&quot;AND&quot; Memory with Accumulator</td>
</tr>
<tr>
<td>ASL</td>
<td>Shift Left One Bit (Memory or Accumulator)</td>
</tr>
<tr>
<td>BCC</td>
<td>Branch on Carry Clear</td>
</tr>
<tr>
<td>BCS</td>
<td>Branch on Carry Set</td>
</tr>
<tr>
<td>BEQ</td>
<td>Branch on Equal Result</td>
</tr>
<tr>
<td>BIT</td>
<td>Test Bits in Memory with Accumulator</td>
</tr>
<tr>
<td>BMI</td>
<td>Branch on Negative Result</td>
</tr>
<tr>
<td>BNE</td>
<td>Branch on Result not Zero</td>
</tr>
<tr>
<td>BPL</td>
<td>Branch on Positive Result Plus</td>
</tr>
<tr>
<td>BRK</td>
<td>Force Break</td>
</tr>
<tr>
<td>BVS</td>
<td>Branch on Overflow Clear</td>
</tr>
<tr>
<td>BVC</td>
<td>Branch on Overflow Set</td>
</tr>
<tr>
<td>CLC</td>
<td>Clear Carry Flag</td>
</tr>
<tr>
<td>CLD</td>
<td>Clear Decimal Mode</td>
</tr>
<tr>
<td>CLI</td>
<td>Clear Interrupt Disable Bit</td>
</tr>
<tr>
<td>CLV</td>
<td>Clear Overflow Flag</td>
</tr>
<tr>
<td>CMP</td>
<td>Compare Memory and Accumulator</td>
</tr>
<tr>
<td>CPX</td>
<td>Compare Memory and Index X</td>
</tr>
<tr>
<td>CPY</td>
<td>Compare Memory and Index Y</td>
</tr>
<tr>
<td>DEC</td>
<td>Decrement Memory by One</td>
</tr>
<tr>
<td>DEY</td>
<td>Decrement Index Y by One</td>
</tr>
<tr>
<td>DEX</td>
<td>Decrement Index X by One</td>
</tr>
<tr>
<td>DEX</td>
<td>Decrement Index X by One</td>
</tr>
<tr>
<td>JMP</td>
<td>Jump to New Location</td>
</tr>
<tr>
<td>JSR</td>
<td>Jump to New Location Saving Return Address</td>
</tr>
<tr>
<td>LDA</td>
<td>Load Accumulator with Memory</td>
</tr>
<tr>
<td>LDX</td>
<td>Load Index X with Memory</td>
</tr>
<tr>
<td>LDY</td>
<td>Load Index Y with Memory</td>
</tr>
<tr>
<td>ORA</td>
<td>OR Memory with Accumulator</td>
</tr>
<tr>
<td>PHA</td>
<td>Push Accumulator on Stack</td>
</tr>
<tr>
<td>PHP</td>
<td>Push Processor Status on Stack</td>
</tr>
<tr>
<td>PLA</td>
<td>Pull Accumulator from Stack</td>
</tr>
<tr>
<td>PLP</td>
<td>Pull Processor Status from Stack</td>
</tr>
<tr>
<td>ROL</td>
<td>Rotate One Bit Left (Memory or Accumulator)</td>
</tr>
<tr>
<td>RTS</td>
<td>Return From Interrupt</td>
</tr>
<tr>
<td>RTS</td>
<td>Return From Subroutine</td>
</tr>
<tr>
<td>SBC</td>
<td>Subtract Memory from Accumulator with Borrow</td>
</tr>
<tr>
<td>SEC</td>
<td>Set Carry Flag</td>
</tr>
<tr>
<td>SED</td>
<td>Set Decimal Mode</td>
</tr>
<tr>
<td>sei</td>
<td>Set Interrupt Disable Status</td>
</tr>
<tr>
<td>STS</td>
<td>Store Accumulator in Memory</td>
</tr>
<tr>
<td>STA</td>
<td>Store Index X in Memory</td>
</tr>
<tr>
<td>TAX</td>
<td>Transfer Accumulator to Index X</td>
</tr>
<tr>
<td>TAY</td>
<td>Transfer Accumulator to Index Y</td>
</tr>
<tr>
<td>TSX</td>
<td>Transfer Stack Pointer to Index X</td>
</tr>
<tr>
<td>TXA</td>
<td>Transfer Index X to Accumulator</td>
</tr>
<tr>
<td>TXS</td>
<td>Transfer Index X to Stack Pointer</td>
</tr>
<tr>
<td>TYA</td>
<td>Transfer Index Y to Accumulator</td>
</tr>
</tbody>
</table>

#### EXECUTION TIMES (IN CLOCK CYCLES)

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Accumulator</th>
<th>Zero Page</th>
<th>Zero Page, Indirect</th>
<th>Absolute</th>
<th>Absolute, Indirect</th>
<th>Relative</th>
<th>Indexed</th>
<th>Absolute Indirect</th>
<th>Indexed indirect</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC, AND, ASL, BCC, BCS, BEQ, BIT, BMI, BNE, BPL, BRK, BVS, BVC, CLC, CLD, CLI, CLV, CMP, CPX, CPY, DEC, DEY, DEX, DEX, JMP, JSR, LDA, LDX, LDY, ORA, PHA, PHP, PLA, PLP, ROL, RTS, RTS, SBC, SEC, SED, sei, STA, STS, TAX, TAY, TSX, TXA, TYA</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

#### ASSEMBLER DIRECTIVES

- **.OPT** - If used must be the first executable statement in the program.
- **.OPTIONS ARE:** - (Options listed are the default value.)
- **COUNT** (COU or CNT) - List all instructions and their addresses.
- **NOGENERATE** (NOG) - Do not generate more than one line of code for ASCII strings.
- **XREF (XRE)** - Produce a cross-reference list in the symbol table.
- **ERRORS (ERR)** - Memory (MEM) - Produce an assembler object output file.
- **LIST (LIS)** - Produce a full assembly listing.
- **BYTE** - Produces a single BYTE in memory equal to each operand specified.
- **WORD** - Produces two BYTES in memory equal to each operand specified.
- **PAGE** - Defines the beginning of a new program counter sequence.
- **PAGE** - Advances the listing to the top of a new page.
- **END** - Defines the end of a source program.

#### LABELS:

- Labels begin in column 1 and are separated from the instruction by at least one space.
- Labels can be up to 6 alphanumeric characters long and must begin with an alpha character.
- A, X, Y, S, and P are reserved and cannot be used as labels.
- **LABEL** - Expression can be used to equate labels to instructions.
- **LABEL** *= * N can be used to reserve areas in memory.

#### CHARACTERS USED AS SPECIAL PREFIXES:

- Indicates an assembler directive.
- Specifies the immediate mode of addressing.
- Specifies a hexadecimal character.
- Specifies an octal number.
- Specifies a binary number.
- Specifies an ASCII literal character.
- Indicates indirect addressing.
- In column 1 indicates a comment.