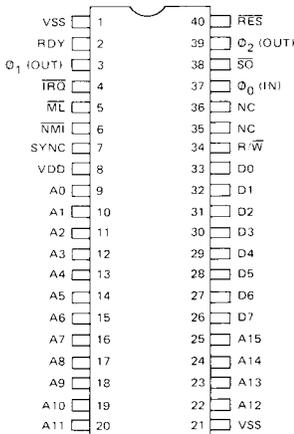


- Enhanced software performance including 27 additional OP codes encompassing ten new instructions and two additional addressing modes.
- 66 microprocessor instructions.
- 15 addressing modes.
- 178 operational codes.
- 1MHz, 2MHz operation.
- Operates at frequencies as low as 200 Hz for even lower power consumption (pseudo-static: stop during Φ_2 high).
- Compatible with NMOS 6500 series microprocessors.
- 64 K-byte addressable memory.

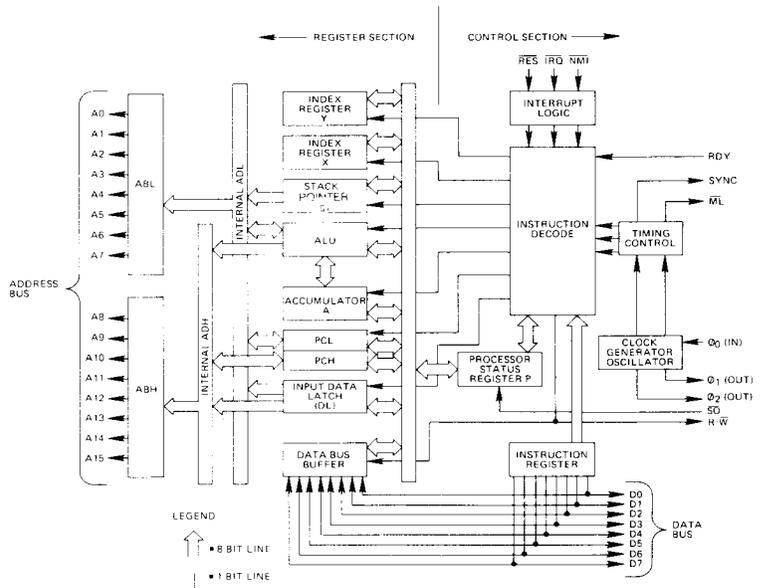
The NCR CMOS 6502 is an 8-bit microprocessor which is software compatible with the NMOS 6502. The NCR65C02 hardware interfaces with all 6500 peripherals. The enhancements include ten additional instructions, expanded operational codes and two new addressing modes. This microprocessor has all of the advantages of CMOS technology: low power consumption, increased noise immunity and higher reliability. The CMOS 6502 is a low power high performance microprocessor with applications in the consumer, business, automotive and communications market.

- Interrupt capability.
- Lower power consumption. 4mA @ 1MHz.
- +5 volt power supply.
- 8-bit bidirectional data bus.
- Bus Compatible with M6800.
- Non-maskable interrupt.
- 40 pin dual-in-line packaging.
- 8-bit parallel processing
- Decimal and binary arithmetic.
- Pipeline architecture.
- Programmable stack pointer.
- Variable length stack.
- Optional internal pullups for (RDY, IRQ, SO, NMI and RES)

PIN CONFIGURATION



FUNCTIONAL BLOCK DIAGRAM



* Specifications are subject to change without notice.

Copyright ©1982 by NCR Corporation, Dayton, Ohio, USA

MICROCOMPUTERS/
MICROPROCESSORS



■ **ABSOLUTE MAXIMUM RATINGS:** ($V_{DD} = 5.0\text{ V} \pm 5\%$, $V_{SS} = 0\text{ V}$, $T_A = 0^\circ\text{ to } +70^\circ\text{C}$)

RATING	SYMBOL	VALUE	UNIT
SUPPLY VOLTAGE	V_{DD}	-0.3 to +7.0	V
INPUT VOLTAGE	V_{IN}	-0.3 to +7.0	V
OPERATING TEMP.	T_A	0 to +70	°C
STORAGE TEMP.	T_{STG}	-55 to +150	°C

■ **PIN FUNCTION**

PIN	FUNCTION
A0 - A15	Address Bus
D0 - D7	Data Bus
\overline{IRQ}^*	Interrupt Request
RDY [*]	Ready
\overline{ML}	Memory Lock
\overline{NMI}^*	Non-Maskable Interrupt
SYNC	Synchronize
\overline{RES}^*	Reset
\overline{SO}^*	Set Overflow
NC	No Connection
R/ \overline{W}	Read/Write
VDD	Power Supply (+5V)
VSS	Internal Logic Ground
\emptyset_0	Clock Input
\emptyset_1, \emptyset_2	Clock Output

*This pin has an optional internal pullup for a No Connect condition.

■ **DC CHARACTERISTICS**

	SYMBOL	MIN.	TYP.	MAX	UNIT
Input High Voltage \emptyset_0 (IN)	V_{IH}	$V_{SS} + 2.4$	—	V_{DD}	V
Input High Voltage $\overline{RES}, \overline{NMI}, \text{RDY}, \overline{IRQ}, \text{Data}, \text{S.O.}$		$V_{SS} + 2.0$	—	—	V
Input Low Voltage \emptyset_0 (IN)	V_{IL}	$V_{SS} - 0.3$	—	$V_{SS} + 0.4$	V
$\overline{RES}, \overline{NMI}, \text{RDY}, \overline{IRQ}, \text{Data}, \text{S.O.}$		—	—	$V_{SS} + 0.8$	V
Input Leakage Current ($V_{IN} = 0$ to 5.25V, $V_{DD} = 5.25\text{V}$)	I_{IN}				
With pullups		-30	—	+30	μA
Without pullups		—	—	+1.0	μA
Three State (Off State) Input Current ($V_{IN} = 0.4$ to 2.4V, $V_{CC} = 5.25\text{V}$)					
Data Lines	I_{TSI}	—	—	10	μA
Output High Voltage ($I_{OH} = -100\ \mu\text{A}_{dc}$, $V_{DD} = 4.75\text{V}$ SYNC, Data, A0-A15, R/W)	V_{OH}	$V_{SS} + 2.4$	—	—	V
Out Low Voltage ($I_{OL} = 1.6\text{mA}_{dc}$, $V_{DD} = 4.75\text{V}$ SYNC, Data, A0-A15, R/W)	V_{OL}	—	—	$V_{SS} + 0.4$	V
Supply Current $f = 1\text{MHz}$	I_{DD}	—	—	4	mA
Supply Current $f = 2\text{MHz}$	I_{DD}	—	—	8	mA
Capacitance ($V_{IN} = 0$, $T_A = 25^\circ\text{C}$, $f = 1\text{MHz}$)	C				pF
Logic	C_{IN}	—	—	5	
Data		—	—	10	
A0-A15, R/W, SYNC	C_{out}	—	—	10	
\emptyset_0 (IN)	C_{\emptyset_0} (IN)	—	—	10	

■ AC CHARACTERISTICS $V_{DD} = 5.0V \pm 5\%$, $T_A = 0^\circ C$ to $70^\circ C$, Load = 1 TTL + 130 pF

Parameter	Symbol	1MHZ		2MHZ		3MHZ		Unit
		Min	Max	Min	Max	Min	Max	
Delay Time, θ_0 (IN) to θ_2 (OUT)	t_{DLY}	—	60	—	60	20	60	nS
Delay Time, θ_1 (OUT) to θ_2 (OUT)	t_{DLY1}	-20	20	-20	20	-20	20	nS
Cycle Time	t_{CYC}	1.0	5000*	0.50	5000*	0.33	5000*	μ S
Clock Pulse Width Low	t_{PL}	460	—	220	—	160	—	nS
Clock Pulse Width High	t_{PH}	460	—	220	—	160	—	nS
Fall Time, Rise Time	t_F, t_R	—	25	—	25	—	25	nS
Address Hold Time	t_{AH}	20	—	20	—	0	—	nS
Address Setup Time	t_{ADS}	—	225	—	140	—	110	nS
Access Time	t_{ACC}	650	—	310	—	170	—	nS
Read Data Hold Time	t_{DHR}	10	—	10	—	10	—	nS
Read Data Setup Time	t_{DSU}	100	—	60	—	60	—	nS
Write Data Delay Time	t_{MDS}	—	30	—	30	—	30	nS
Write Data Hold Time	t_{DHW}	20	—	20	—	15	—	nS
\overline{SO} Setup Time	t_{SO}	100	—	100	—	100	—	nS
Processor Control Setup Time**	t_{PCS}	200	—	150	—	150	—	nS
SYNC Setup Time	t_{SYNC}	—	225	—	140	—	100	nS
\overline{ML} Setup Time	t_{ML}	—	225	—	140	—	100	nS
Input Clock Rise/Fall Time	t_{F00}, t_{R00}	—	25	—	25	—	25	nS

*NCR65C02 can be held static with θ_2 high.

**This parameter must only be met to guarantee that the signal will be recognized at the current clock cycle.

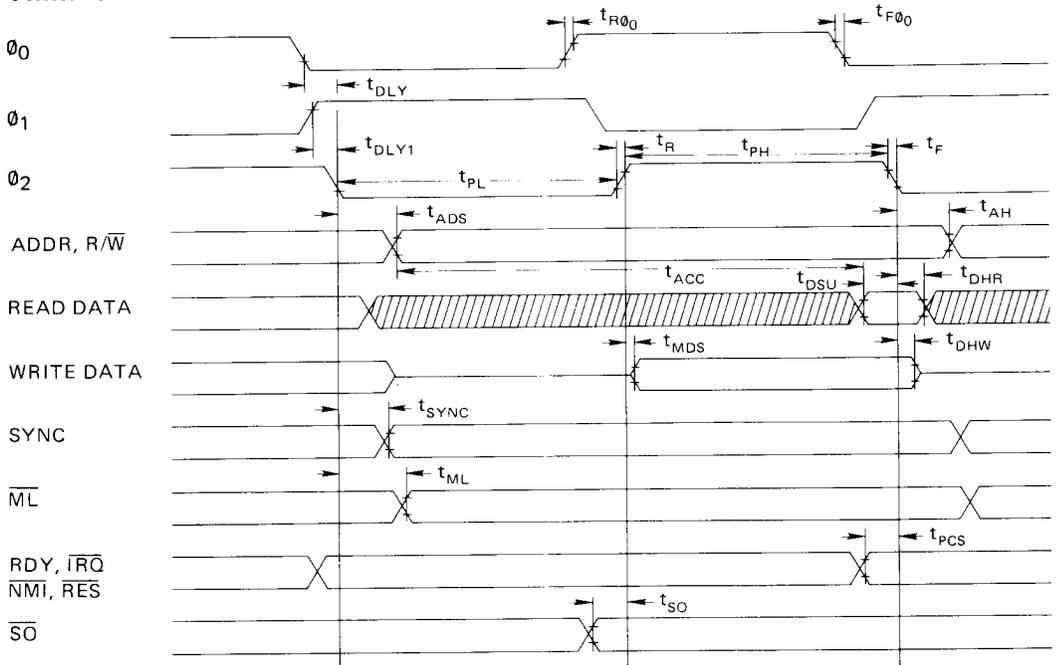
■ MICROPROCESSOR OPERATIONAL ENHANCEMENTS

Function	NMOS 6502 Microprocessor	NCR65C02 Microprocessor
Indexed addressing across page boundary.	Extra read of invalid address.	Extra read of last instruction byte.
Execution of invalid op codes.	Some terminate only by reset. Results are undefined.	All are NOPs (reserved for future use). Op Code Bytes Cycles X2 2 2 X3, X7, XB, XF 1 1 44 2 3 54, D4, F4 2 4 5C 3 8 DC, FC 3 4
Jump indirect, operand = XXFF.	Page address does not increment.	Page address increments and adds one additional cycle.
Read/modify/write instructions at effective address.	One read and two write cycles.	Two read and one write cycle.
Decimal flag.	Indeterminate after reset.	Initialized to binary mode (D=0) after reset and interrupts.
Flags after decimal operation.	Invalid N, V and Z flags.	Valid flag adds one additional cycle.
Interrupt after fetch of BRK instruction.	Interrupt vector is loaded, BRK vector is ignored.	BRK is executed, then interrupt is executed.

■ MICROPROCESSOR HARDWARE ENHANCEMENTS

Function	NMOS 6502	NCR65C02
Assertion of Ready RDY during write operations.	Ignored.	Stops processor during θ_2 .
Unused input-only pins (\overline{IRQ} , NMI, RDY, RES, \overline{SO}).	Must be connected to low impedance signal to avoid noise problems.	Connected internally by a high-resistance to V_{DD} (approximately 250 K ohm.)

■ TIMING DIAGRAM



Note: All timing is referenced from a high voltage of 2.0 volts and a low voltage of 0.8 volts.

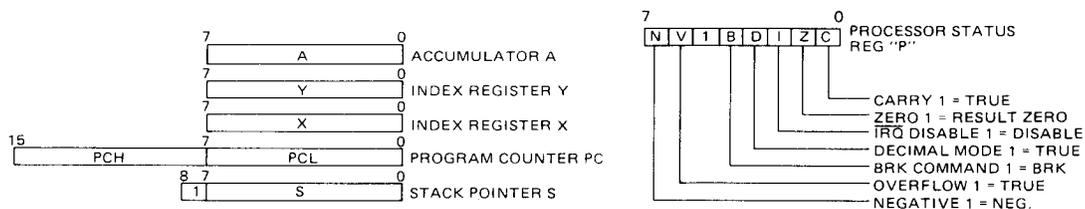
■ NEW INSTRUCTION MNEMONICS

HEX	MNEMONIC	DESCRIPTION
80	BRA	Branch relative always [Relative]
3A	DEA	Decrement accumulator [Accum]
1A	INA	Increment accumulator [Accum]
DA	PHX	Push X on stack [Implied]
5A	PHY	Push Y on stack [Implied]
FA	PLX	Pull X from stack [Implied]
7A	PLY	Pull Y from stack [Implied]
9C	STZ	Store zero [Absolute]
9E	STZ	Store zero [ABS, X]
64	STZ	Store zero [Zero page]
74	STZ	Store zero [ZPG,X]
1C	TRB	Test and reset memory bits with accumulator [Absolute]
14	TRB	Test and reset memory bits with accumulator [Zero page]
0C	TSB	Test and set memory bits with accumulator [Absolute]
04	TSB	Test and set memory bits with accumulator [Zero page]

■ ADDITIONAL INSTRUCTION ADDRESSING MODES

HEX	MNEMONIC	DESCRIPTION
72	ADC	Add memory to accumulator with carry [(ZPG)]
32	AND	"AND" memory with accumulator [(ZPG)]
3C	BIT	Test memory bits with accumulator [ABS, X]
34	BIT	Test memory bits with accumulator [ZPG, X]
D2	CMP	Compare memory and accumulator [(ZPG)]
52	EOR	"Exclusive Or" memory with accumulator [(ZPG)]
7C	JMP	Jump (New addressing mode) [ABS(IND,X)]
B2	LDA	Load accumulator with memory [(ZPG)]
12	ORA	"OR" memory with accumulator [(ZPG)]
F2	SBC	Subtract memory from accumulator with borrow [(ZPG)]
92	STA	Store accumulator in memory [(ZPG)]

■ MICROPROCESSOR PROGRAMMING MODEL



■ FUNCTIONAL DESCRIPTION

Timing Control

The timing control unit keeps track of the instruction cycle being monitored. The unit is set to zero each time an instruction fetch is executed and is advanced at the beginning of each phase one clock pulse for as many cycles as is required to complete the instruction. Each data transfer which takes place between the registers depends upon decoding the contents of both the instruction register and the timing control unit.

Program Counter

The 16-bit program counter provides the addresses which step the microprocessor through sequential instructions in a program.

Each time the microprocessor fetches an instruction from program memory, the lower byte of the program counter (PCL) is placed on the low-order bits of the address bus and the higher byte of the program counter (PCH) is placed on the high-order 8 bits. The counter is incremented each time an instruction or data is fetched from program memory.

Instruction Register and Decode

Instructions fetched from memory are gated onto the internal data bus. These instructions are latched into the instruction register, then decoded, along with timing and interrupt signals, to generate control signals for the various registers.

Arithmetic and Logic Unit (ALU)

All arithmetic and logic operations take place in the ALU including incrementing and decrementing internal registers (except the program counter). The ALU has no internal memory and is used only to perform logical and transient numerical operations.

Accumulator

The accumulator is a general purpose 8-bit register that stores the results of most arithmetic and logic operations, and in addition, the accumulator usually contains one of the two data words used in these operations.

Index Registers

There are two 8-bit index registers (X and Y), which may be used to count program steps or to provide an index value to be used in generating an effective address.

When executing an instruction which specifies indexed addressing, the CPU fetches the op code and the base address, and modifies the address by adding the index register to it prior to performing the desired operation. Pre- or post-indexing of indirect addresses is possible (see addressing modes).

Stack Pointer

The stack pointer is an 8-bit register used to control the addressing of the variable-length stack on page one. The stack pointer is automatically incremented and decremented under control of the microprocessor to perform stack manipulations under direction of either the program or interrupts (NMI and IRQ). The stack allows simple implementation of nested subroutines and multiple level interrupts. The stack pointer should be initialized before any interrupts or stack operations occur.

Processor Status Register

The 8-bit processor status register contains seven status flags. Some of the flags are controlled by the program, others may be controlled both by the program and the CPU. The 6500 instruction set contains a number of conditional branch instructions which are designed to allow testing of these flags (see microprocessor programming model).

■ ADDRESSING MODES

Fifteen addressing modes are available to the user of the NCR65C02 microprocessor. The addressing modes are described in the following paragraphs:

Implied Addressing [Implied]

In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

Accumulator Addressing [Accum]

This form of addressing is represented with a one byte instruction and implies an operation on the accumulator.

Immediate Addressing [Immediate]

With immediate addressing, the operand is contained in the second byte of the instruction; no further memory addressing is required.

Absolute Addressing [Absolute]

For absolute addressing, the second byte of the instruction specifies the eight low-order bits of the effective address, while the third byte specifies the eight high-order bits. Therefore, this addressing mode allows access to the total 64K bytes of addressable memory.

Zero Page Addressing [Zero Page]

Zero page addressing allows shorter code and execution times by only fetching the second byte of the instruction and assuming a zero high address byte. The careful use of zero page addressing can result in significant increase in code efficiency.

Absolute Indexed Addressing [ABS, X or ABS, Y]

Absolute indexed addressing is used in conjunction with X or Y index register and is referred to as "Absolute, X," and "Absolute, Y." The effective address is formed by adding the contents of X or Y to the address contained in the second and third bytes of the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields, resulting in reduced coding and execution time.

Zero Page Indexed Addressing [ZPG, X or ZPG, Y]

Zero page absolute addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y." The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additionally, due to the "Zero Page" addressing nature of this mode, no carry is added to the high-order eight bits of memory, and crossing of page boundaries does not occur.

Relative Addressing [Relative]

Relative addressing is used only with branch instructions;

it establishes a destination for the conditional branch. The second byte of the instruction becomes the operand which is an "Offset" added to the contents of the program counter when the counter is set at the next instruction. The range of the offset is -128 to +127 bytes from the next instruction.

Zero Page Indexed Indirect Addressing [(IND, X)]

With zero page indexed indirect addressing (usually referred to as indirect X) the second byte of the instruction is added to the contents of the X index register; the carry is discarded. The result of this addition points to a memory location on page zero whose contents is the low-order eight bits of the effective address. The next memory location in page zero contains the high-order eight bits of the effective address. Both memory locations specifying the high- and low-order bytes of the effective address must be in page zero.

***Absolute Indexed Indirect Addressing [ABS(IND, X)] (Jump Instruction Only)**

With absolute indexed indirect addressing the contents of the second and third instruction bytes are added to the X register. The result of this addition, points to a memory location containing the lower-order eight bits of the effective address. The next memory location contains the higher-order eight bits of the effective address.

Indirect Indexed Addressing [(IND), Y]

This form of addressing is usually referred to as Indirect, Y. The second byte of the instruction points to a memory location in page zero. The contents of this memory location are added to the contents of the Y index register, the result being the low-order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location, the result being the high-order eight bits of the effective address.

***Zero Page Indirect Addressing [(ZPG)]**

In the zero page indirect addressing mode, the second byte of the instruction points to a memory location on page zero containing the low-order byte of the effective address. The next location on page zero contains the high-order byte of the effective address.

Absolute Indirect Addressing [(ABS)] (Jump Instruction Only)

The second byte of the instruction contains the low-order eight bits of a memory location. The high-order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low-order byte of the effective address. The next memory location contains the high-order byte of the effective address which is loaded into the 16 bit program counter.

NOTE: * = New Address Modes

▪ SIGNAL DESCRIPTION

Address Bus (A0-A15)

A0-A15 forms a 16-bit address bus for memory and I/O exchanges on the data bus. The output of each address line is TTL compatible, capable of driving one standard TTL load and 130pF.

Clocks (θ_0 , θ_1 , and θ_2)

θ_0 is a TTL level input that is used to generate the internal clocks in the 6502. Two full level output clocks are generated by the 6502. The θ_2 clock output is in phase with θ_0 . The θ_1 output pin is 180° out of phase with θ_0 . (See timing diagram.)

Data Bus (D0-D7)

The data lines (D0-D7) constitute an 8-bit bidirectional data bus used for data exchanges to and from the device and peripherals. The outputs are three-state buffers capable of driving one TTL load and 130 pF.

Interrupt Request (\overline{IRQ})

This TTL compatible input requests that an interrupt sequence begin within the microprocessor. The \overline{IRQ} is sampled during θ_2 operation; if the interrupt flag in the processor status register is zero, the current instruction is completed and the interrupt sequence begins during θ_1 . The program counter and processor status register are stored in the stack. The microprocessor will then set the interrupt mask flag high so that no further \overline{IRQ} s may occur. At the end of this cycle, the program counter low will be loaded from address FFFE, and program counter high from location FFFF, transferring program control to the memory vector located at these addresses. The RDY signal must be in the high state for any interrupt to be recognized. A 3K ohm external resistor should be used for proper wire OR operation.

Memory Lock (\overline{ML})

In a multiprocessor system, the \overline{ML} output indicates the need to defer the re arbitration of the next bus cycle to ensure the integrity of read-modify-write instructions. \overline{ML} goes low during ASL, DEC, INC, LSR, ROL, ROR, TRB, TSB memory referencing instructions. This signal is low for the modify and write cycles.

Non-Maskable Interrupt (\overline{NMI})

A negative-going edge on this input requests that a non-maskable interrupt sequence be generated within the microprocessor. The \overline{NMI} is sampled during θ_2 ; the current instruction is completed and the interrupt sequence begins during θ_1 . The program counter is loaded with the interrupt vector from locations FFFA (low byte) and FFFB (high byte), thereby transferring program control to the non-maskable interrupt routine.

Note: Since this interrupt is non-maskable, another \overline{NMI} can occur before the first is finished. Care should be taken when using \overline{NMI} to avoid this.

Ready (RDY)

This input allows the user to single-cycle the microprocessor on all cycles including write cycles. A negative transition to the low state, during or coincident with phase one (θ_1), will halt the microprocessor with the output address lines reflecting the current address being fetched. This condition will remain through a subsequent phase two (θ_2) in which the ready signal is low. This feature allows microprocessor interfacing with low-speed memory as well as direct memory access (DMA).

Reset (\overline{RES})

This input is used to reset the microprocessor. Reset must be held low for at least two clock cycles after V_{DD} reaches operating voltage from a power down. A positive transition on this pin will then cause an initialization sequence to begin. Likewise, after the system has been operating, a low on this line of at least two cycles will cease microprocessing activity, followed by initialization after the positive edge on \overline{RES} .

When a positive edge is detected, there is an initialization sequence lasting six clock cycles. Then the interrupt mask flag is set, the decimal mode is cleared, and the program counter is loaded with the restart vector from locations FFFC (low byte) and FFFD (high byte). This is the start location for program control. This input should be high in normal operation.

Read/Write (R/\overline{W})

This signal is normally in the high state indicating that the microprocessor is reading data from memory or I/O bus. In the low state the data bus has valid data from the microprocessor to be stored at the addressed memory location.

Set Overflow (\overline{SO})

A negative transition on this line sets the overflow bit in the status code register. The signal is sampled on the trailing edge of θ_1 .

Synchronize (SYNC)

This output line is provided to identify those cycles during which the microprocessor is doing an OP CODE fetch. The SYNC line goes high during θ_1 of an OP CODE fetch and stays high for the remainder of that cycle. If the RDY line is pulled low during the θ_1 clock pulse in which SYNC went high, the processor will stop in its current state and will remain in the state until the RDY line goes high. In this manner, the SYNC signal can be used to control RDY to cause single instruction execution.

■ INSTRUCTION SET — ALPHABETICAL SEQUENCE

ADC	Add Memory to Accumulator with Carry	LDX	Load Index X with Memory
AND	"AND" Memory with Accumulator	LDY	Load Index Y with Memory
ASL	Shift One Bit Left	LSR	Shift One Bit Right
BCC	Branch on Carry Clear	NOP	No Operation
BCS	Branch on Carry Set	ORA	"OR" Memory with Accumulator
BEQ	Branch on Result Zero	PHA	Push Accumulator on Stack
BIT	Test Memory Bits with Accumulator	PHP	Push Processor Status on Stack
BMI	Branch on Result Minus	* PHX	Push Index X on Stack
BNE	Branch on Result not Zero	* PHY	Push Index Y on Stack
BPL	Branch on Result Plus	PLA	Pull Accumulator from Stack
* BRA	Branch Always	PLP	Pull Processor Status from Stack
BRK	Force Break	* PLX	Pull Index X from Stack
BVC	Branch on Overflow Clear	* PLY	Pull Index Y from Stack
BVS	Branch on Overflow Set	ROL	Rotate One Bit Left
CLC	Clear Carry Flag	ROR	Rotate One Bit Right
CLD	Clear Decimal Mode	RTI	Return from Interrupt
CLI	Clear Interrupt Disable Bit	RTS	Return from Subroutine
CLV	Clear Overflow Flag	SBC	Subtract Memory from Accumulator with Borrow
CMP	Compare Memory and Accumulator	SEC	Set Carry Flag
CPX	Compare Memory and Index X	SED	Set Decimal Mode
CPY	Compare Memory and Index Y	SEI	Set Interrupt Disable Bit
* DEA	Decrement Accumulator	STA	Store Accumulator in Memory
DEC	Decrement by One	STX	Store Index X in Memory
DEX	Decrement Index X by One	STY	Store Index Y in Memory
DEY	Decrement Index Y by One	* STZ	Store Zero in Memory
EOR	"Exclusive-or" Memory with Accumulator	TAX	Transfer Accumulator to Index X
* INA	Increment Accumulator	TAY	Transfer Accumulator to Index Y
INC	Increment by One	* TRB	Test and Reset Memory Bits with Accumulator
INX	Increment Index X by One	* TSB	Test and Set Memory Bits with Accumulator
INY	Increment Index Y by One	TSX	Transfer Stack Pointer to Index X
JMP	Jump to New Location	TXA	Transfer Index X to Accumulator
JSR	Jump to New Location Saving Return Address	TXS	Transfer Index X to Stack Pointer
LDA	Load Accumulator with Memory	TYA	Transfer Index Y to Accumulator

Note: * = New Instruction

■ MICROPROCESSOR OP CODE TABLE

S D	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	BRK rel	ORA ind, X			TSB* zpg	ORA zpg	ASL zpg		PHP	ORA imm	ASL A		TSB* abs	ORA abs	ASL abs	0
1	BPL rel	ORA ind, Y	ORA*† (zpg)		TRB* zpg	ORA zpg, X	ASL zpg, X		CLC	ORA abs, Y	INA* A		TRB* abs	ORA abs, X	ASL abs, X	1
2	JSR abs	AND ind, X			BIT zpg	AND zpg	ROL zpg		PLP	AND imm	ROL A		BIT abs	AND abs	ROL abs	2
3	BMI rel	AND ind, Y	AND*† (zpg)		BIT* zpg, X	AND zpg, X	ROL zpg, X		SEC	AND abs, Y	DEA* A		BIT*† abs, X	AND abs, X	ROL abs, X	3
4	RTI	EOR ind, X			EOR zpg	LSR zpg			PHA	EOR imm	LSR A		JMP abs	EOR abs	LSR abs	4
5	BVC rel	EOR ind, Y	EOR*† (zpg)		EOR zpg, X	LSR zpg, X			CLI	EOR abs, Y	PHY* A			EOR abs, X	LSR abs, X	5
6	RTS	ADC ind, X			STZ* zpg	ADC zpg	ROR zpg		PLA	ADC imm	ROR A		JMP (abs)	ADC abs	ROR abs	6
7	BVS rel	ADC ind, Y	ADC*† (zpg)		STZ* zpg, X	ADC zpg, X	ROR zpg, X		SEI	ADC abs, Y	PLY* A		JMP*† abs(ind, X)	ADC abs, X	ROR abs, X	7
8	BRA* rel	STA ind, X			STY zpg	STA zpg	STX zpg		DEY	BIT* imm	TXA		STY abs	STA abs	STX abs	8
9	BCC rel	STA ind, Y	STA*† (zpg)		STY zpg, X	STA zpg, X	STX zpg, Y		TYA	STA abs, Y	TXS		STZ* abs	STA abs, X	STZ* abs, X	9
A	LDY imm	LDA ind, X	LDX imm		LDY zpg	LDA zpg	LDX zpg		TAY	LDA imm	TAX		LDY abs	LDA abs	LDX abs	A
B	BCS rel	LDA ind, Y	LDA*† (zpg)		LDY zpg, X	LDA zpg, X	LDX zpg, Y		CLV	LDA abs, Y	TSX		LDY abs, X	LDA abs, X	LDX abs, Y	B
C	CPY imm	CMP ind, X			CPY zpg	CMP zpg	DEC zpg		INY	CMP imm	DEX		CPY abs	CMP abs	DEC abs	C
D	BNE rel	CMP ind, Y	CMP*† (zpg)		CMP zpg, X	DEC zpg, X			CLD	CMP abs, Y	PHX* A			CMP abs, X	DEC abs, X	D
E	CPX imm	SBC ind, X			CPX zpg	SBC zpg	INC zpg		INX	SBC imm	NOP		CPX abs	SBC abs	INC abs	E
F	BEQ rel	SBC ind, Y	SBC*† (zpg)		SBC zpg, X	INC zpg, X			SED	SBC abs, Y	PLX* A			SBC abs, X	INC abs, X	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Note: * = New OP Codes

Note: † = New Address Modes

