

USP - ICMC - SSC

SSC 0511 - Sist. Informação - 2o. Semestre 2014

Disciplina de Organização de Computadores Digitais

Prof. Fernando Santos Osório

Email: fosorio [at] { icmc. usp. br , gmail. com }

Página Pessoal: <http://www.icmc.usp.br/~fosorio/>

Material on-line: Wiki ICMC:

[http://wiki.icmc.usp.br/index.php/SSC-511-2014\(fosorio\)](http://wiki.icmc.usp.br/index.php/SSC-511-2014(fosorio))



Lab. de Robótica Móvel



Centro de Robótica da USP

Conteúdos Abordados:

1. Microprocessador 6502

- **Microprocessador Comercial MOSTEK 6502**
- **Usado no Apple II, Atari (400, 800, 2600), Commodore (VIC-20)**
- **Emulador do Apple II**

- **Arquitetura da CPU: Registradores, ULA, Barramento**
- **CPU: Principais Características**
- **Modos de Endereçamento**
- **Principais Instruções**
- **Simulador do 6502**

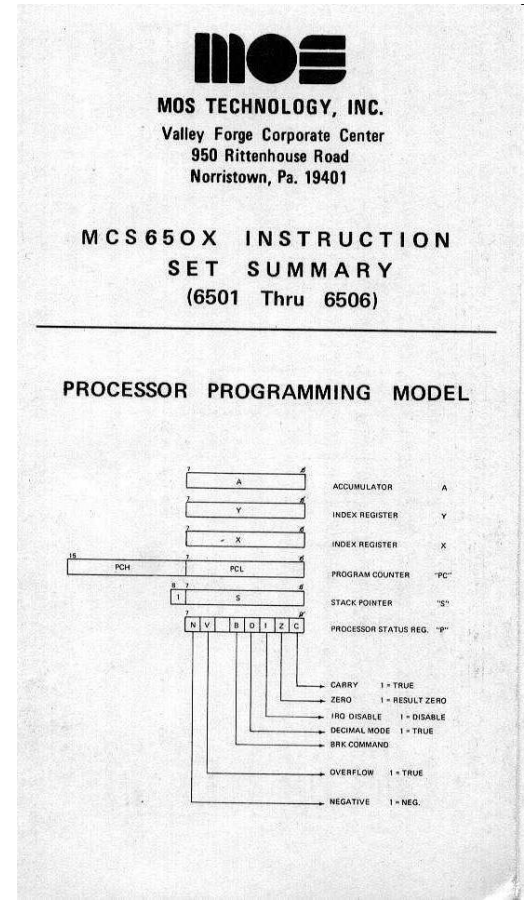
MosTech 6502 - Adotado no Apple II

Microprocessador de 8 bits dados e 16 bits de endereço



VSS	1	40	RES
RDY	2	39	ϕ_2 (OUT)
ϕ_1 (OUT)	3	38	S0
IRQ	4	37	ϕ_0 (IN)
N.C.	5	36	N.C.
$\overline{\text{NMI}}$	6	35	N.C.
SYNC	7	34	R/W
VCC	8	33	D0
A0	9	32	D1
A1	10	31	D2
A2	11	30	D3
A3	12	29	D4
A4	13	28	D5
A5	14	27	D6
A6	15	26	D7
A7	16	25	A15
A8	17	24	A14
A9	18	23	A13
A10	19	22	A12
A11	20	21	VSS

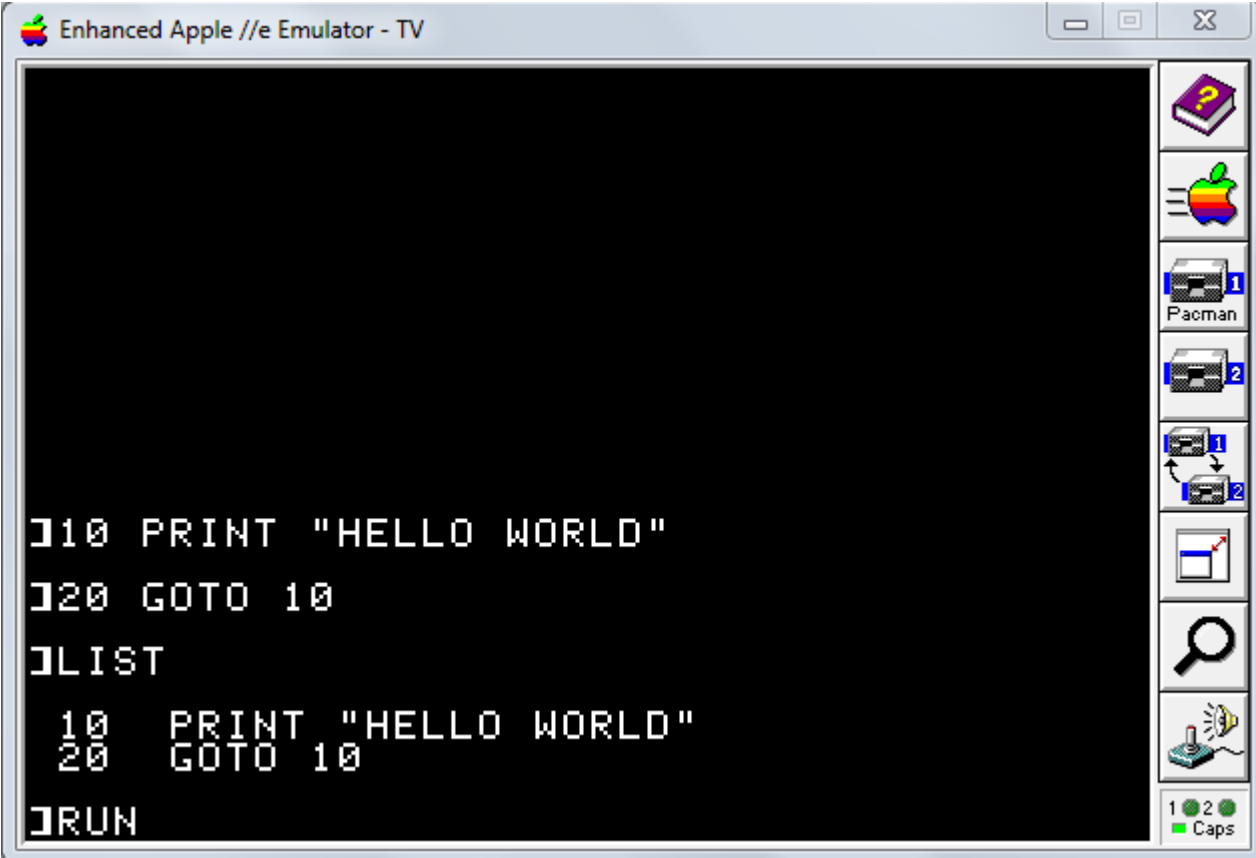
6502 Pin configuration (40-Pin DIP)



MosTech 6502 - Adotado no Apple II

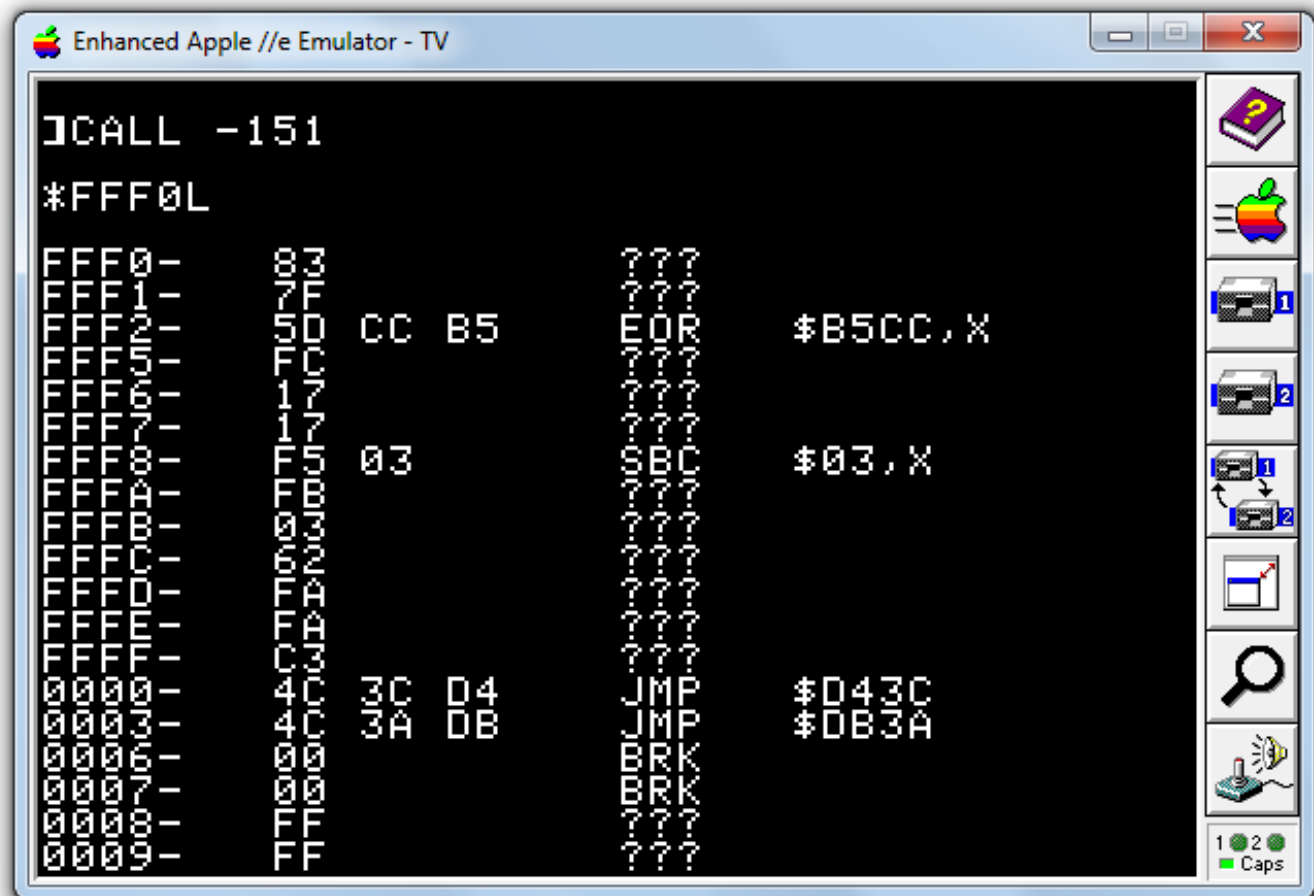


MosTech 6502 - Adotado no Apple II

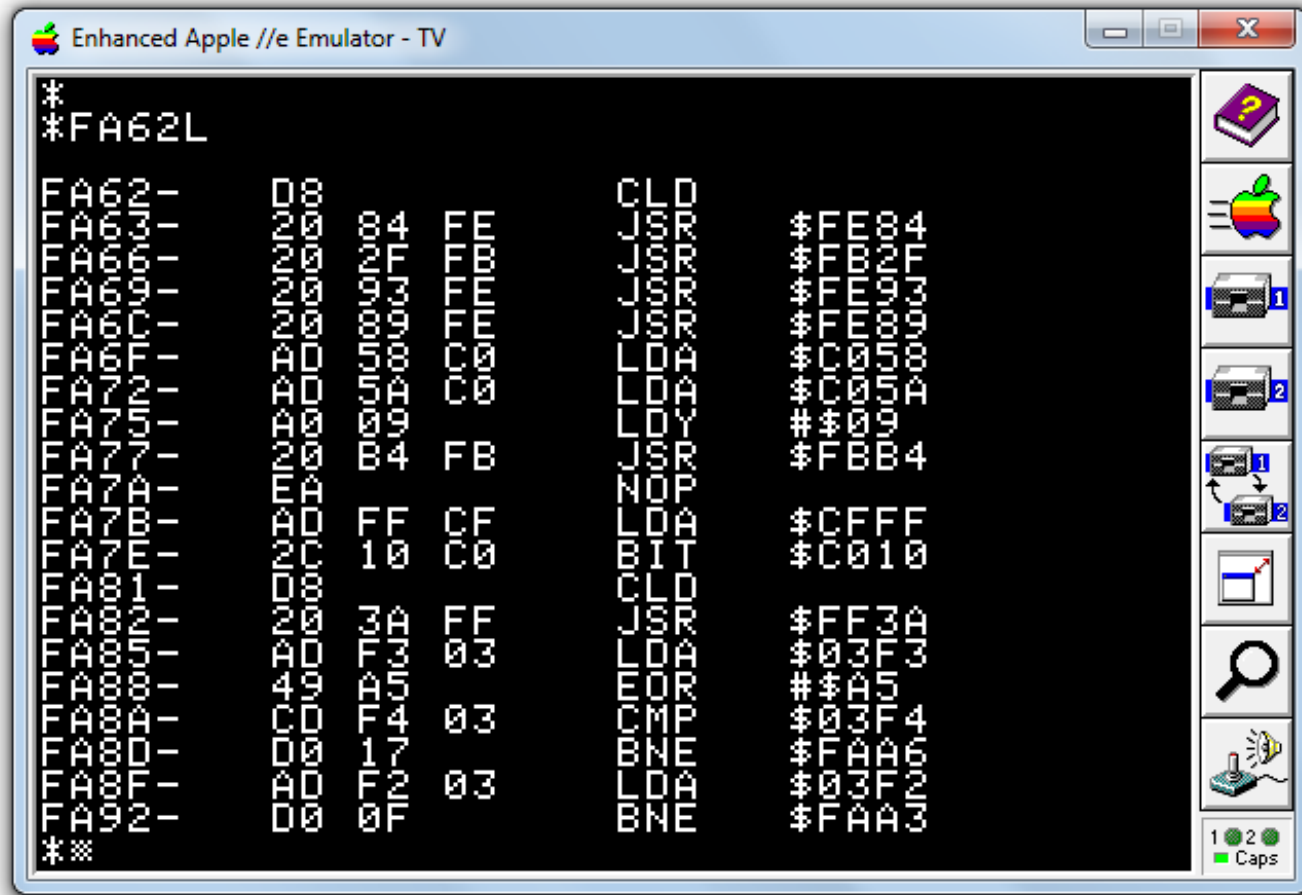


```
Enhanced Apple //e Emulator - TV  
]10 PRINT "HELLO WORLD"  
]20 GOTO 10  
]LIST  
 10 PRINT "HELLO WORLD"  
 20 GOTO 10  
]RUN
```

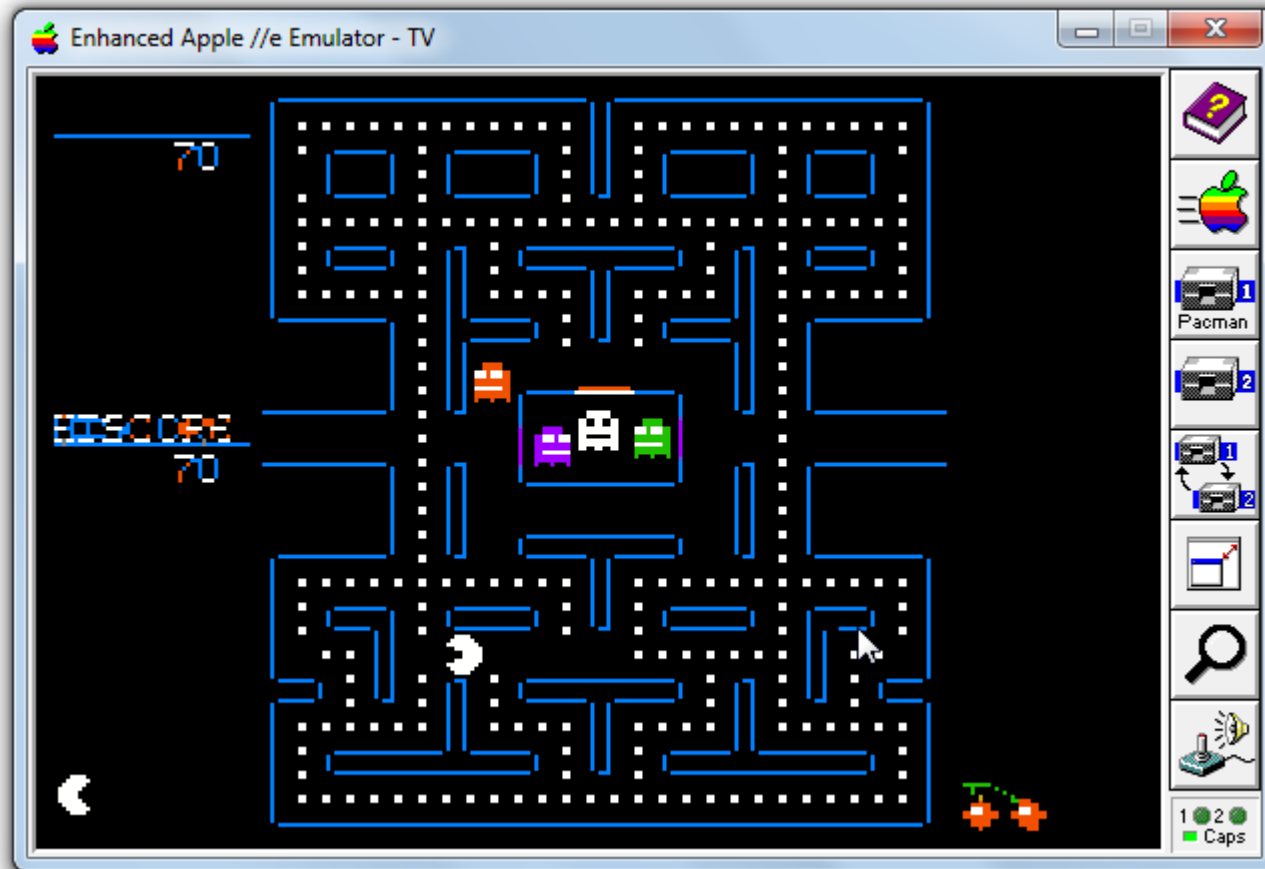
MosTech 6502 - Adotado no Apple II



MosTech 6502 - Adotado no Apple II



MosTech 6502 - Adotado no Apple II



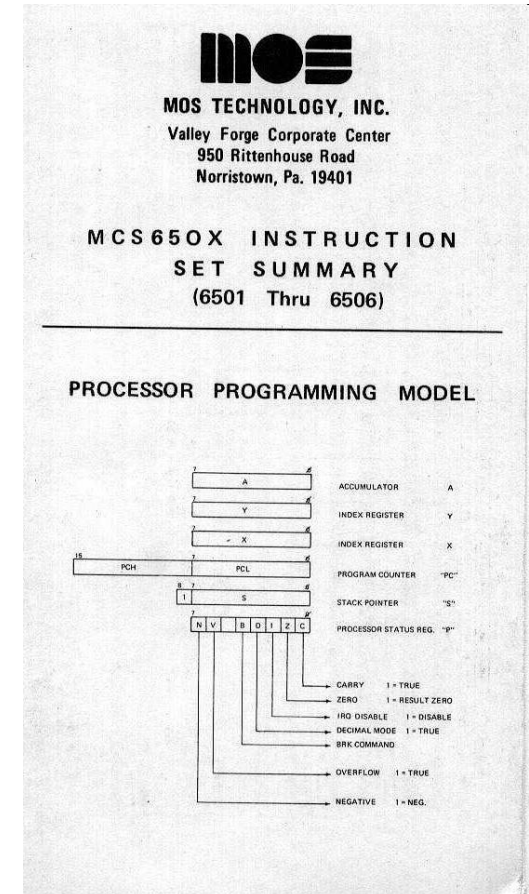
MosTech 6502 - Adotado no Apple II

Microprocessador de 8 bits dados e 16 bits de endereço



VSS	1	40	RES
RDY	2	39	ϕ_2 (OUT)
ϕ_1 (OUT)	3	38	S0
IRQ	4	37	ϕ_0 (IN)
N.C.	5	36	N.C.
$\overline{\text{NMI}}$	6	35	N.C.
SYNC	7	34	R/W
VCC	8	33	D0
A0	9	32	D1
A1	10	31	D2
A2	11	30	D3
A3	12	29	D4
A4	13	28	D5
A5	14	27	D6
A6	15	26	D7
A7	16	25	A15
A8	17	24	A14
A9	18	23	A13
A10	19	22	A12
A11	20	21	VSS

6502 Pin configuration (40-Pin DIP)



MosTech 6502 - Adotado no Apple II

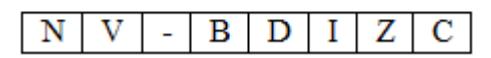
Microprocessador de 8 bits dados e 16 bits de endereço

Registers

The 6502 only has 6 registers. Five are 8 bits wide, one is 16 bits wide.

A	Accumulator
X	Index Register X
Y	Index Register Y
PCH PCL	Program Counter
S	Stack Pointer
P	Processor Status (Flags)

Processor Status Register



- N - Negative Flag
- V - Overflow Flag
- B - Break Command
- D - Decimal Mode
- I - Int. Disable
- Z - Zero Flag
- C - Carry Flag

Accumulator 8 bits A

MosTech 6502 - Adotado no Apple II

Accumulator **8 bits** **A**

Used for all arithmetic and logical operations (apart from increments and decrements). Data must be loaded into the Acc before it can be manipulated.

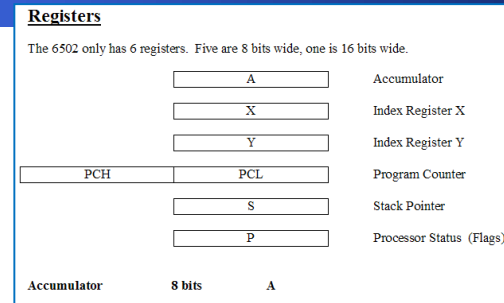
Index Register X **8 bits** **X**

Usually used to hold counters or offsets for accessing memory. Contents can be compared with memory locations and incremented and decremented.

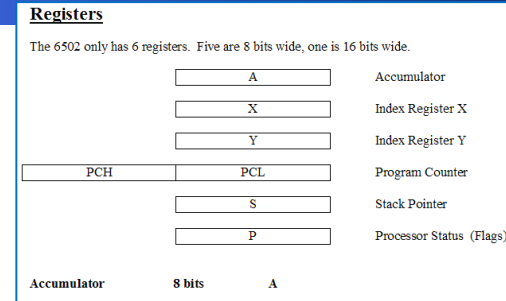
Unlike other registers (including the Y register), can be used to get a copy of the stack pointer or change its value.

Index Register Y **8 bits** **Y**

Usually used to hold counters or offsets for accessing memory. Contents can be compared with memory locations and incremented or decremented.



MosTech 6502 - Adotado no Apple II



Program Counter 16 bits PC

Contains the address of the next instruction to be executed. Automatically incremented by the hardware, but can be altered by a jump, branch or subroutine call / return.

Stack Pointer 8 bits S

The 6502 uses a 256 byte stack located on page 1 (\$0100 to \$01FF). The stack pointer is an 8 bit register that holds the least significant byte of the next free location on the stack. This means that the stack cannot be moved. The stack starts at \$01FF and grows downward. When a byte is pushed onto the stack, the stack pointer is decremented. When something is popped off the stack, it is incremented.

The hardware does not detect stack overflow. The programmer must ensure that the program does not make excessive demands on the stack space.

MosTech 6502 - Adotado no Apple II

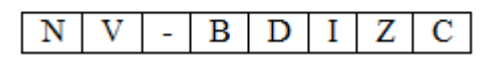
Microprocessador de 8 bits dados e 16 bits de endereço

Registers

The 6502 only has 6 registers. Five are 8 bits wide, one is 16 bits wide.

A	A	Accumulator
X	X	Index Register X
Y	Y	Index Register Y
PCH	PCL	Program Counter
S	S	Stack Pointer
P	P	Processor Status (Flags)

Processor Status Register



- N - Negative Flag
- V - Overflow Flag
- B - Break Command
- D - Decimal Mode
- I - Int. Disable
- Z - Zero Flag
- C - Carry Flag

Accumulator **8 bits** **A**

MosTech 6502 - Adotado no Apple II

Processor Status Register

N	V	-	B	D	I	Z	C
---	---	---	---	---	---	---	---

N - Negative Flag
V - Overflow Flag
B - Break Command
D - Decimal Mode
I - Int. Disable
Z - Zero Flag
C - Carry Flag

Also known as the flags register. Used to indicate the results of an operation. Each bit in the register signifies a different condition. Some of the instructions allow you to test the values of various bits, set them, clear them, and push the entire set onto the stack (or pop them off).

Carry Flag C

The carry flag is set if the last operation caused an overflow from the most significant bit (bit 7) of the result or an underflow from the least significant bit (bit 0).

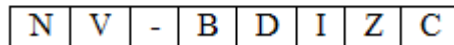
This condition is set during arithmetic instructions, comparison instructions and during logical shifts. It can be explicitly set using the 'Set Carry Flag' (SEC) instruction and cleared with 'Clear Carry Flag' (CLC).

Zero Flag Z

The zero flag is set if the result of the last operation was zero.

MosTech 6502 - Adotado no Apple II

Processor Status Register



N - Negative Flag
V - Overflow Flag
B - Break Command
D - Decimal Mode
I - Int. Disable
Z - Zero Flag
C - Carry Flag

Interrupt Disable I

The interrupt disable flag is set if the program has executed a 'Set Interrupt Disable' (SEI) instruction. While this flag is set the processor will not respond to interrupts from external devices until it is cleared by a 'Clear Interrupt Disable' (CLI) instruction.

Decimal Mode D

While the decimal mode flag is set the processor will obey the rules of Binary Coded Decimal (BCD) arithmetic during addition and subtraction. The flag can be explicitly set using 'Set Decimal Flag' (SED) and cleared with 'Clear Decimal Flag' (CLD).

Break Command B

The break command bit is set when a BRK instruction has been executed and an interrupt has been generated to process it.

MosTech 6502 - Adotado no Apple II

Processor Status Register

N	V	-	B	D	I	Z	C
---	---	---	---	---	---	---	---

N - Negative Flag
V - Overflow Flag
B - Break Command
D - Decimal Mode
I - Int. Disable
Z - Zero Flag
C - Carry Flag

Overflow Flag V

The overflow flag is set during arithmetic operations if the result has yielded an invalid 2's complement result (e.g. adding two positive numbers and ending up with a negative result: $64 + 64 \Rightarrow -128$).

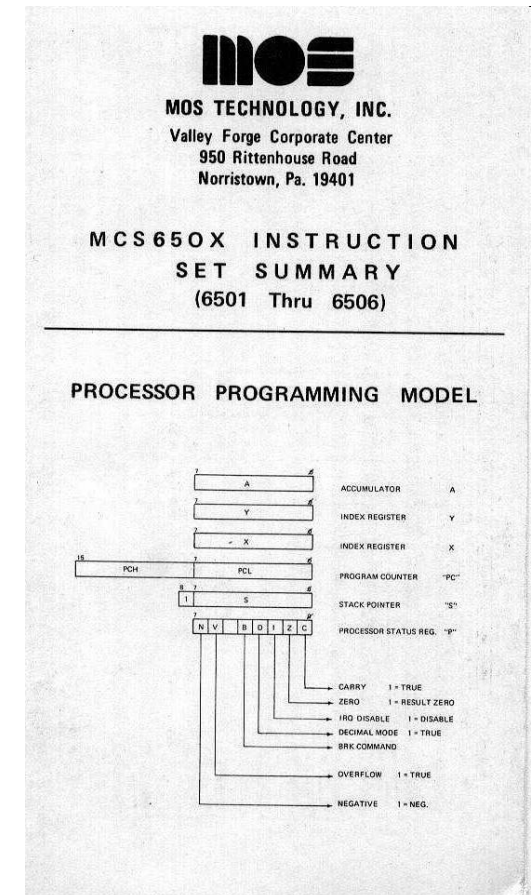
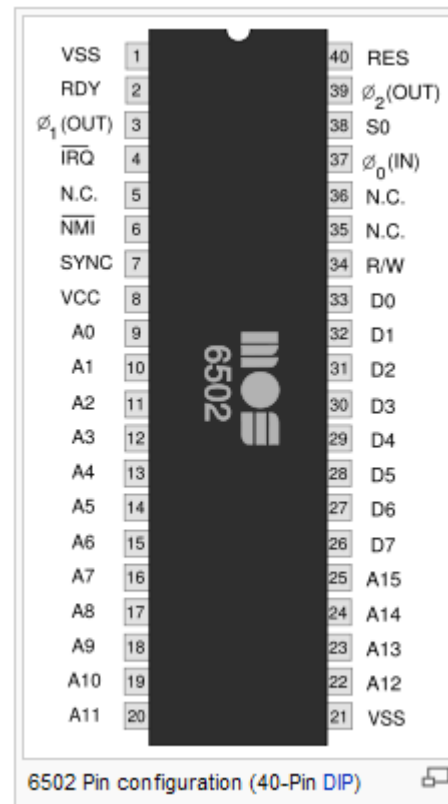
If the most significant bit of the two numbers being added is the same, and if it is different to the msb of the result, then the overflow flag will be set.

Negative Flag N

The negative flag is set if the result of the last operation had the most significant bit (bit 7) set to a one.

MosTech 6502 - Adotado no Apple II

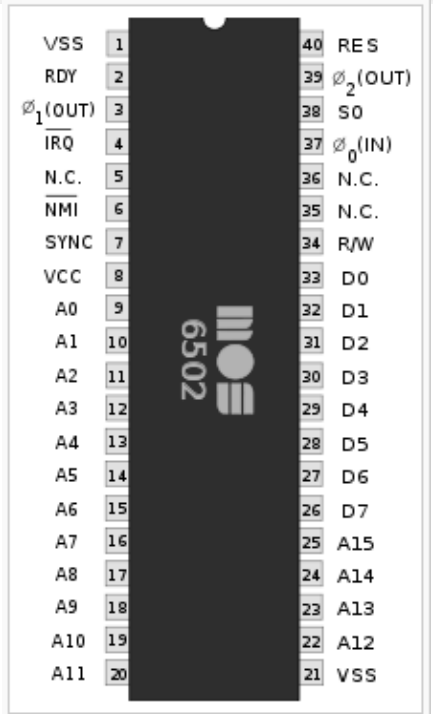
Microprocessador de 8 bits dados e 16 bits de endereço



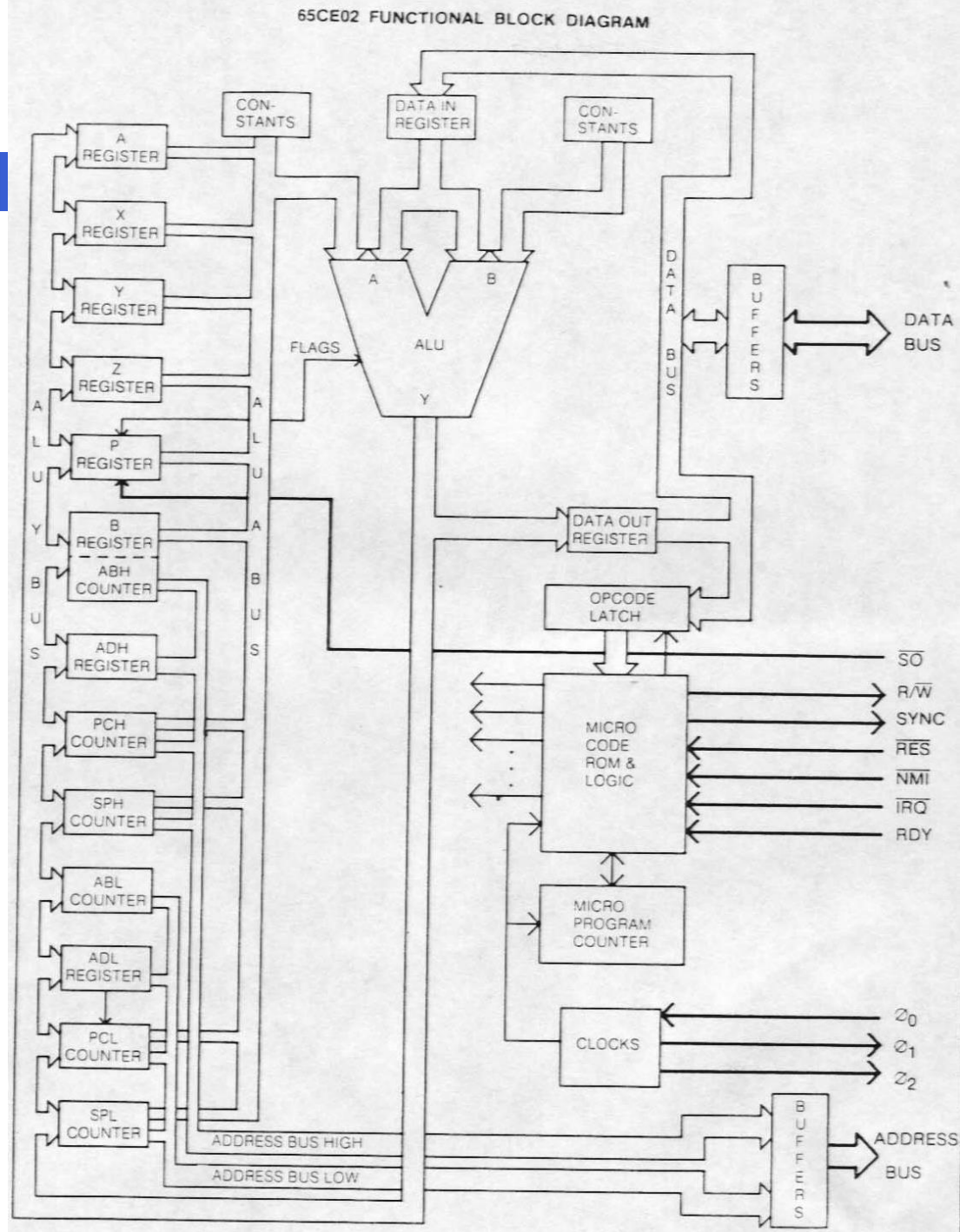
Microprocessador 6502



65CE02 MICROPROCESSOR



6502 Pin configuration (40-Pin DIP)



Microprocessador 6502

http://en.wikipedia.org/wiki/MOS_Technology_6502

MOS Technologies 6502

The MOS Technology 6502 is an 8-bit microprocessor that was designed by Chuck Peddle and Bill Mensch for MOS Technology in 1975. When it was introduced, it was the least expensive full-featured microprocessor on the market by a considerable margin, costing less than one-sixth the price of competing designs from larger companies such as Motorola and Intel. The 6502 is an 8-bit processor with a 16-bit address bus.

The 6502 was designed primarily by the same engineering team that had designed the Motorola 6800. After resigning from Motorola en masse, the team went looking for another company that would be interested in hosting a design team, and found MOS Technology, then a small chipmaking company whose main product was a single-chip implementation of the popular Pong video game.

Clock: 1Mz (6502) 2Mz (6502A) 3Mhz (6502B)

Pins: 40-pin DIP

Data Bus: 8 bits

Address Bus: 16 bits (64Kb addressable memory)

Manufacturer: MOS Tech – Year: 1975



Programação 6502 – Addressing Modes



Basic Specifications

Clock speed 1, 2 and even 3 Mhz models were available.

Wordlength 8 bits

Input / Output Memory mapped.

There are no I/O registers in the processor, so a set of memory addresses will be allocated to the I/O devices. The motherboard hardware will ensure that if a value is written to memory address \$E000 (for example), it will be sent straight to a specific output device.

Programação 6502 – Addressing Modes



Addresses

Address bus 16 bits

Addresses in the range 0000000000000000 to 1111111111111111 binary
or \$0000 to \$FFFF hex

can be accessed by the processor.

Addressable Memory is therefore 64 Kb

Little endian Addresses stored least significant byte first.

A 16 bit address needs to be stored in two consecutive bytes. A little endian processor will store the address \$458D as \$8D followed by the byte \$45.

Programação 6502 – Addressing Modes

Paged Memory

Memory is viewed as a set of 256 byte pages.

The first page (\$0000 to \$00FF) is called the ‘Zero page’, and can be accessed by using a special addressing mode which enables you to use shorter and therefore faster executing instructions.

This makes it useful for storing tables of values or addresses that are going to be accessed frequently by your program.

The second page (\$0100 to \$01FF) is used to hold the **system stack**. This is used to keep track of values, especially during subroutine calls. It cannot be moved.

Other reserved locations:

\$FFFA	Address of NMI handler
\$FFFB	ditto
\$FFFC	Address of power on reset location
\$FFFD	ditto
\$FFFE	BRK / interrupt request handler
\$FFFF	ditto

Programação 6502 – Addressing Modes

Principais Modos de endereçamento

Absolute (Direct) addressing

The operand is the address in memory where the data item can be found.

LDA	\$6F55		This instruction will access memory location \$6F55, and copy the contents into the accumulator register.

Immediate addressing

The operand is the data item.

LDA	#41		This instruction will load the number 41 (\$29) into the accumulator register.

Programação 6502

Principais Modos de endereçamento

Indexed addressing

The operand is added to the contents of the index register, and this gives the location in memory of the data.

LDA	\$4F10,X		\$34		This instruction will access the memory location \$4F44, and copy the contents into the accumulator.
			Index Register X		

Indirect addressing

The operand is the address of a memory location which contains the address of the data item.

LDA	(\$2B57)		2B57	3C	This instruction will load the data item 44 into the accumulator register.
			2B58	6A	
			6A3C	44	

Programação 6502

Principais Modos de endereçamento

Relative addressing

This is used in conjunction with branches. The operand is a 2's comp number (-128 - +127) which is added to the address of the instruction following the branch instruction to give the address of the instruction to be jumped to.

4545	BEQ	\$13	This BEQ command will cause control to be passed to the instruction in location \$4547 + \$13 = \$455A, provided that the zero flag is set.
4547	LDA	\$9333	

Resumo: (ver documentação associada ao Simulador do 6502)

Absolute	aaaa	Zero Page Indexed, X	aa, X
Zero Page	aa	Zero Page Indexed, Y	aa, Y
Immediate	#aa	Indirect Absolute	(aaaa)
Implicit		Indexed Indirect	(aa, X)
Accumulator	A	Indirect Indexed	(aa), Y
Absolute Indexed, X	aaaa, X	Relative	aa
Absolute Indexed, Y	aaaa, Y		

Programação 6502 – Instruction Set

Principais Instruções

Load and Store Group

● LDA	Load Accumulator	N,Z
LDX	Load X Register	N,Z
LDY	Load Y Register	N,Z
● STA	Store Accumulator	
STX	Store X Register	
STY	Store Y Register	

Arithmetic Group

● ADC	Add with Carry	N,V,Z,C
● SBC	Subtract with Carry	N,V,Z,C

Increment and Decrement Group

● INC	Increment a memory location	N,Z
INX	Increment the X register	N,Z
INY	Increment the Y register	N,Z
● DEC	Decrement a memory location	N,Z
DEX	Decrement the X register	N,Z
DEY	Decrement the Y register	N,Z

Register Transfer Group

TAX	Transfer accumulator to X	N,Z
TAY	Transfer accumulator to Y	N,Z
TXA	Transfer X to accumulator	N,Z
TYA	Transfer Y to accumulator	N,Z

Logical Group

AND	Logical AND	N,Z
EOR	Exclusive OR	N,Z
ORA	Logical Inclusive OR	N,Z

Compare and Bit Test Group

● CMP	Compare accumulator	N,Z,C
CPX	Compare X register	N,Z,C
CPY	Compare Y register	N,Z,C
BIT	Bit Test	N,V,Z

Shift and Rotate Group

ASL	Arithmetic Shift Left	N,Z,C
LSR	Logical Shift Right	N,Z,C
ROL	Rotate Left	N,Z,C
ROR	Rotate Right	N,Z,C

Programação 6502

Instruction Set

Principais Instruções

Subroutine and Interrupt Group

● JSR	Jump to a subroutine	
● RTS	Return from subroutine	
BRK	Force an interrupt	B
RTI	Return from Interrupt	All
NOP	No Operation	

Jump and Branch Group

● JMP	Jump to another location	
BCC	Branch if carry flag clear	
BCS	Branch if carry flag set	
● BEQ	Branch if zero flag set	
BMI	Branch if negative flag set	
● BNE	Branch if zero flag clear	
BPL	Branch if negative flag clear	
BVC	Branch if overflow flag clear	
BVS	Branch if overflow flag set	

Stack Group

TSX	Transfer stack pointer to X	N,Z
TXS	Transfer X to stack pointer	
● PHA	Push accumulator on stack	
PHP	Push processor status on stack	
● PLA	Pull accumulator from stack	N,Z
PLP	Pull processor status from stack	All

Status Flag Change Group

CLC	Clear carry flag	C
CLD	Clear decimal mode flag	D
CLI	Clear interrupt disable flag	I
CLV	Clear overflow flag	V
SEC	Set carry flag	C
SED	Set decimal mode flag	D
SEI	Set interrupt disable flag	I

Microprocessador 6502

MosTech 6502 - Adotado no Apple II

The screenshot shows the 6502 Simulator interface with the following components:

- Main Editor:** Displays assembly code for adding two 8-bit numbers (4 and 6).

```
; Program to add two 8 bit numbers  
; The numbers being added are 4 and 6  
; Add0801.65s  
  
.ORG $0200 ; Store machine code starting here  
  
LDA #$04 ; Store first number (4) in  
STA no1 ; byte labelled no1  
LDA #$06 ; Store second number (6) in  
STA no2 ; byte labelled no2  
  
CLC ;  
LDA no1 ;  
ADC no2 ;  
STA res ;  
  
BRK ; S  
  
no1: .DB $00 ; T  
; r  
no2: .DB $00 ; ;  
; t  
res: .DB $00 ; T
```
- 6502 μP Registers & Status:** Shows register values and status flags.

A= \$00	0, '?',00000000	CLK: 0	<- zero		
X= \$00	0, '?',00000000	N	Z	V	C
Y= \$00	0, '?',00000000	P= \$20	I	B	D
S=	\$FF	- empty stack -			
PC=	\$0200	LDA #\$04	Arg:	4, '?',00000100	
Stat:	OK				
- 6502 μP Memory:** Shows memory contents starting at address 0200.

01F8	00	00	00	00	00	00	00	00	00	>	<
0200	A9	04	8D	16	02	A9	06	8D		>	<
0208	17	02	18	AD	16	02	6D	17		>	<
0210	02	8D	18	02	00	00	00	00		>	<
0218	00	00	00	00	00	00	00	00		>	<
0220	00	00	00	00	00	00	00	00		>	<
0228	00	00	00	00	00	00	00	00		>	<
0230	00	00	00	00	00	00	00	00		>	<
0238	00	00	00	00	00	00	00	00		>	<
0240	00	00	00	00	00	00	00	00		>	<
0248	00	00	00	00	00	00	00	00		>	<
0250	00	00	00	00	00	00	00	00		>	<
0258	00	00	00	00	00	00	00	00		>	<
0260	00	00	00	00	00	00	00	00		>	<
0268	00	00	00	00	00	00	00	00		>	<
0270	00	00	00	00	00	00	00	00		>	<
0278	00	00	00	00	00	00	00	00		>	<
0280	00	00	00	00	00	00	00	00		>	<
0288	00	00	00	00	00	00	00	00		>	<
0290	00	00	00	00	00	00	00	00		>	<
0298	00	00	00	00	00	00	00	00		>	<
02A0	00	00	00	00	00	00	00	00		>	<
- 6502 μP Stack:** Shows an empty stack starting at address 1FF.

1FF	00	'	'	0000	00	00	00	00	00	>	<
1FE	00	'	'	0000	00	00	00	00	00	>	<
1FD	00	'	'	0000	00	00	00	00	00	>	<
1FC	00	'	'	0000	00	00	00	00	00	>	<
1FB	00	'	'	0000	00	00	00	00	00	>	<
1FA	00	'	'	0000	00	00	00	00	00	>	<
1F9	00	'	'	0000	00	00	00	00	00	>	<
1F8	00	'	'	0000	00	00	00	00	00	>	<
1F7	00	'	'	0000	00	00	00	00	00	>	<
1F6	00	'	'	0000	00	00	00	00	00	>	<
1F5	00	'	'	0000	00	00	00	00	00	>	<
1F4	00	'	'	0000	00	00	00	00	00	>	<
1F3	00	'	'	0000	00	00	00	00	00	>	<
1F2	00	'	'	0000	00	00	00	00	00	>	<
1F1	00	'	'	0000	00	00	00	00	00	>	<
1F0	00	'	'	0000	00	00	00	00	00	>	<
1EF	00	'	'	0000	00	00	00	00	00	>	<
1EE	00	'	'	0000	00	00	00	00	00	>	<
1ED	00	'	'	0000	00	00	00	00	00	>	<
1EC	00	'	'	0000	00	00	00	00	00	>	<
1EB	00	'	'	0000	00	00	00	00	00	>	<
1EA	00	'	'	0000	00	00	00	00	00	>	<
1E9	00	'	'	0000	00	00	00	00	00	>	<

MosTech 6502 - Adotado no Apple II

6502 Simulator - [Ex1-Add8.65s]

```
; Program to add two 8 bit numbers
; The numbers being added are 4 and 6
; Add0801.65s

.ORG $0200      ; Store machine code starting here

LDA #$04        ; Store first number (4) in
STA no1         ; byte labelled no1
LDA #$06        ; Store second number (6) in
STA no2         ; byte labelled no2

CLC             ;
LDA no1         ;
ADC no2         ;
STA res         ;

BRK             ;

no1: .DB $00    ;
no2: .DB $00    ;
res: .DB $00    ;
```

6502 μP Registers & Status

A= \$00 0, '?',00000000 CLK: 0 <- zero
X= \$00 0, '?',00000000 N Z V C
Y= \$00 0, '?',00000000 P= \$20 I B D
S= \$FF - empty stack -
PC= \$0200 LDA #\$04 Arg: 4, '?',00000100
Stat: OK

6502 μP Memory

01F8	00	00	00	00	00	00	00	00	00	>
0200	A9	04	8D	16	02	A9	06	8D	>	
0208	17	02	18	AD	16	02	6D	17	>	
0210	02	8D	18	02	00	00	00	00	>	
0218	00	00	00	00	00	00	00	00	>	
0220	00	00	00	00	00	00	00	00	>	
0228	00	00	00	00	00	00	00	00	>	
0230	00	00	00	00	00	00	00	00	>	
0238	00	00	00	00	00	00	00	00	>	
0240	00	00	00	00	00	00	00	00	>	
0248	00	00	00	00	00	00	00	00	>	
0250	00	00	00	00	00	00	00	00	>	
0258	00	00	00	00	00	00	00	00	>	
0260	00	00	00	00	00	00	00	00	>	
0268	00	00	00	00	00	00	00	00	>	
0270	00	00	00	00	00	00	00	00	>	
0278	00	00	00	00	00	00	00	00	>	
0280	00	00	00	00	00	00	00	00	>	
0288	00	00	00	00	00	00	00	00	>	
0290	00	00	00	00	00	00	00	00	>	
0298	00	00	00	00	00	00	00	00	>	
02A0	00	00	00	00	00	00	00	00	>	

6502 μP Stack

1FF	00	00	00	00	00	00	00	00	00
-----	----	----	----	----	----	----	----	----	----

- Editor de Textos (com Help)
 - Assembler (Montador)
 - Debugger (Passo-a-passo)
- Exibir/Modificar Registradores
 - Exibir/Modificar Memória
- Disassembler (Desmontador)
 - Console de Entrada/Saída

MosTech 6502 - Adotado no Apple II

Programação do 6502 – Exercícios

- 1) Somar 2 valores de 8 bits
- 2) Somar 2 valores de 16 bits (considerando o “vai um”)
- 3) Somar vários valores de 8 bits ($A + B + C + D + E$)
- 4) Subtrair valores de 8 bits ($A - B$)
- 5) Contador: Laço de contagem até 10
- 6) Somar os dados de um vetor
- 7) Somar valores com mais de 8 bits (!)
- 8) Multiplicar 2 valores
- 9) Pesquisar um dado em uma tabela

MosTech 6502 - Adotado no Apple II

Programação do 6502 – Exercícios

1) Somar 2 valores de 8 bits

```
;  
; Programa para somar dois valores de 8 bits  
;  
                .ORG      $1000  
inicio:  LDA      #$05  
         ADC      #$34  
         STA      Result  
         BRK  
  
Result:  .DB      $00
```

MosTech 6502 - Adotado no Apple II

Programação do 6502 – Exercícios

1) Somar 2 valores de 16 bits

	.ORG \$1000		
	Resultado = \$4000		
inicio:	CLC		.ORG \$2000
	LDA V1L	V1L:	.DB \$FF
	ADC V2L	V1H:	.DB \$01
	STA ResL	V2L:	.DB \$01
	LDA V1H	V2H:	.DB \$01
	ADC V2H	ResL:	.DB \$00
	STA ResH	ResH:	.DB \$00
	BRK		

INFORMAÇÕES SOBRE A DISCIPLINA

USP - Universidade de São Paulo - São Carlos, SP
ICMC - Instituto de Ciências Matemáticas e de Computação
SSC - Departamento de Sistemas de Computação

Prof. Fernando Santos OSÓRIO

Web institucional: <http://www.icmc.usp.br/ssc/>

Página pessoal: <http://www.icmc.usp.br/~fosorio/>

E-mail: [fosorio \[at\] icmc. usp. br](mailto:fosorio@icmc.usp.br) ou [fosorio \[at\] gmail. com](mailto:fosorio@gmail.com)

Disciplina de Organização de Computadores Digitais / BSI

Web disciplina: Wiki ICMC - [Http://wiki.icmc.usp.br](http://wiki.icmc.usp.br)

> Programa, Material de Aulas, Critérios de Avaliação,

> Lista de Exercícios, Trabalhos Práticos, Datas das Provas