CPU6502 Instruction Manual v2.1

09/13/2005

SUNPLUS TECHNOLOGY CO., LTD.

CPU6502 Instruction Manual v2.1

Sunplus Technology reserves the right to change this documentation without prior notice. Information provided by Sunplus Technology is believed to be accurate and reliable. However, Sunplus Technology makes no warranty for any errors which may appear in this document. Contact Sunplus Technology to obtain the latest version of device specifications before placing your order.

No responsibility is assumed by Sunplus Technology for any infringement of patent or other rights of third parties which may result from its use. In addition, Sunplus products are not authorized for use as critical components in life support devices/ systems or aviation devices/systems, where a malfunction or failure of the product may reasonably be expected to result in significant injury to the user, without the express written approval of Sunplus.

If you have suggestions on this documentation which can better serve your needs, please contact:

Sunplus Technology
No. 19, Innovation Road 1, Science-Based Industrial Park,
Hsin-Chu, Taiwan, R.O.C.
FAX: 886-3-578-4418

CONTENT

Revision History

Revision	Date	By	Remark
V2.1	09/13/2005	Joe Chuang	1. Added SPCP bodies in the 65N02 and CPU12 body lists (page.7) 2. Changed the name "Sunplus code" to "CPU12" in the tables of the Instructions
V2.0	12/08/2004	Bean Wang	Page 9: Remove B Flag of status register(P) Was: P: {N,V,X,B,D,I,Z,C}. (X: Not Used) Now: P: {N,V,X,X,D,I,Z,C}. (X: Not Used) Page 27: Remove BRK instruction Page 23~57: Add 65b02 instruction. Page 25: ASL aaaa, X Modify: No. Cycle of 65n02 from 7 to 6*. Page 36: JMP(aaaa) Modify: No. Cycle of 65n02, 65r02, 65s02, Sunplus Code from 3 to 6. Page 39: LSR aaaa, X Modify: No. Cycle of 65n02 from 7 to 6*. Page 42: ROL aaaa, X Modify: No. Cycle of 65n02 from 7 to 6*. Page 43: ROR aaaa, X Modify: No. Cycle of 65n02 from 7 to 6*.
V1.9	02/06/2004	J. K. Chen	Page 30: CMP Was: C: Set if a "borrow" not occurred. (A > M) Now: C: Set if a "borrow" not occurred. (A > = M) Page 31: CPX Was: C: Set if a "borrow" occurred (data > X). Now: C: Set if a "borrow" not occurred. (X > = data) Page 31: CPY Was: C: Set if a "borrow" occurred (data > Y). Now: C: Set if a "borrow" not occurred. (Y > = data) Page 7: remove the CPU type table Page 38: LDX Zero Page, (LDX aa, Y), Sunplus Opcode changed from E9H to B9H. Page 38: 65r02 and Sunplus Code supports LDX aaaa. Page 44: SBC Was: C: Set if there is no "borrow" occurred. (M > A). Now: C: Set if there is no "borrow" occurred. (A > M).
V1.8	12/25/2002	Michael Lin	Page 27: BIT: Set if the bit7 of the result is 1 → set if the memory bit7 of the result is 1. Set if the bit6 of the result is 1 → set if the memory bit6 of the result is 1. Correct the STA, STX, STY, TXS to no effect on status register
V1.7	01/07/2002	Michael Lin	Page 41 Origin: "PLA" takes no effect on any status flag Modified: "PLA" affect the "N" and "Z" flags. Origin: "PLP" takes no effect on any status flag. Modified: "PLP" affects all status flags.
V1.6	08/30/2001	Michael Lin	Page 31 Modify: From: C: Set if a "borrow" occurred. (M > A) To: C: Set if a "borrow" not occurred (A > M)

Revision	Date	By	Remark
V1.5	03/19/2001	Michael Lin	Modify: From: ADC (aa), Y: 6502 Opcode = 1EH To: ADC (aa), Y: 6502 Opcode = 71H
V1.4	12/12/2000	Michael Lin	Page 44, SBC Origin: C: Set if there is a "borrow" occurred. (M > A). Modify to: C: Set if there is ***no*** "borrow" occurred. (M > A).
V1.3	09/08/2000	Michael Lin	Page 7 Update the CPU type of IC. The X2s.,exe is updated to v2.78, 09/08/2000
V1.2	07/18/2000	Michael Lin	Page 44 Origin: SBC: (A-M -C) → A, C Modify to: SBC: (A-M - $\overline{C}$) → A, C

General Description

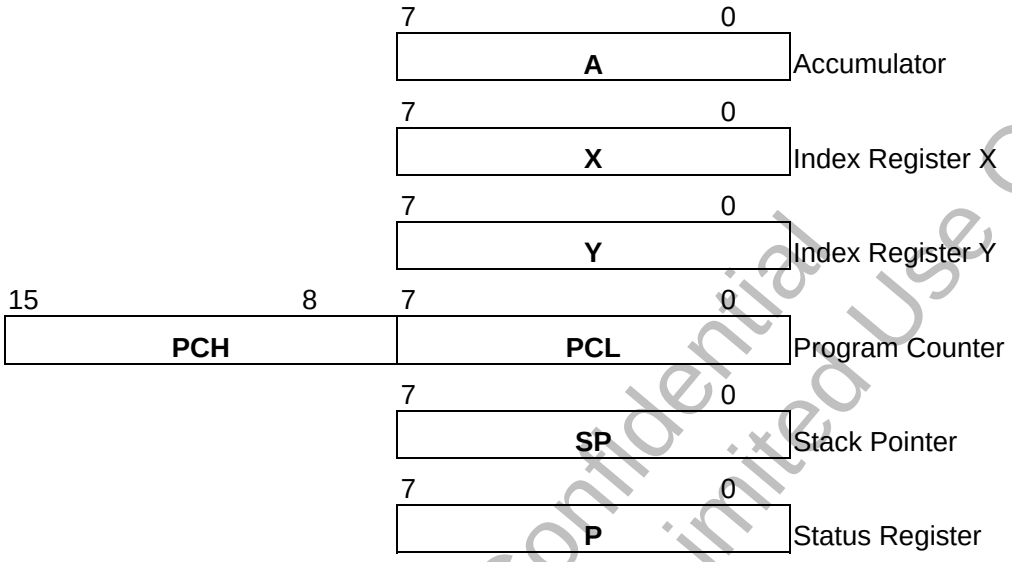
This manual intends to guide users through the 6502 Instruction sets. All 6502 instructions are listed in alphabetical order. However, not all 6502 instructions or addressing modes are available in all SUNPLUS CPUs. To list the types of SUNPLUS CPU, please obtain a tool, named x2s.exe, and apply the following syntax in a DOS command line:

```
C:\>x2s /s
Patch-up Tool CopyRight(c) Sep 08 2000 by SUNPLUS.
2500AD Object Code Convert Program  Ver 2.78
The corresponding Instruction Set to each body is:
   65B02   (Full Set):                    SPMC652,ECMC653

   65N02   (Full Set): (see note.2)       SPL61A,SPL130A,SPL191A,
                                          SPDC256A,SPDC512A,SPDC512B,
                                          SPDC1000A,SPDC1000B
                                          SPLB10A,
                                          SPCP05A, SPCP06A, SPCP08A,
                                          SPCP16A, SPCP18A, SPCP25A, SPCP26A,
                                          SPCP825A, SPCP826A, SPCP835A

   65R02   (Reduce+BIT+TXA+TAX):  SPF02A,SPL02C,SPL02D,
                                          SPL03B,SPL03C,
                                          SPL05A,SPL05B,
                                          SPL06A,SPL06B,
                                          SPLB20A,SPLB20A1,
                                          SPLB21A,SPLB22A,
                                          SPLB23A,SPLB24A,
                                          SPLB25A, SPLB26A
                                          SPL128A,SPLG01

   65S02    (Only Reduce Set):          SPF06A1,SPF18A1,SPF20A,
                                          SPF30A1,SPF30B,
                                          SPL02A

   CPU12 / CPU8: (Reduced instructions+BIT+TXA+TAX)
                                          SPCxxx,SPCRxx,SPMCxx,
                                          SPFA64A,SPFA120A,
                                          SPL08A,SPL08A1,SPL081A,
                                          SPL10A,SPL15A,SPL15B,
                                          SPL25B,SPL25C,SPL30A,
                                          SPL31A,SPL60A,SPL190A,
                                          SPCP02A
```

Note:

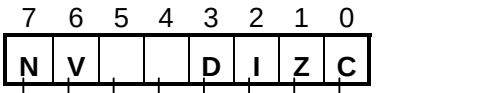
1. Since x2s.exe is updated from time to time, be sure to use the newest version of x2s.exe
2. The CPU type 65N02 has two kinds of body, the one uses standard 6502 OP code, the other one uses Sunplus OP code. Please refer to the programming guide of the individule body.

Register



Register	Size	Description
Accumulator (A)	8 Bit	Accumulator is the only register that can be used for arithmetic or logic operation such as ADD, SUB, AND, OR and EOR and store the result in it.
Index Register X	8 Bit	X is an index register which can be used as a memory buffer, a offset, or a counter.
Index Register Y	8 Bit	Y is an index register which can be used as a memory buffer, a offset, or a counter.
Program Counter(PC)	16 Bit	PC is a 16-bit register. Program Counter points to an address location where an instruction is held and waits to be executed by CPU next. When CPU fetches one instruction to execute, PC is incremented to the next location in memory from which the next instruction to be executed will be taken unless a branch is occurred that will lead PC points to the specified address location.
Stack Pointer(SP)	8 Bit	Stack Pointer is an 8-bit register. Normally, SP is used for storing return address, data of status register or temporary data.
Status Register (P)	8 Bit	Status Register usually offers information on result of previous instruction executed.

Status Register (P)



Carry Flag, If Carry flag is set, C=1

Zero Flag, If arithmetic or logic operation results to zero, Z is set to 1; otherwise, Z=0

Interrupt Disable Flag
If interrupt disable flag is set, I=1, CPU will ignore interrupt signal.
If interrupt disable flag is clear, I=0, CPU will accept interrupt signal.

In CPU6502, this is Decimal Mode Flag
In CPU12, this bit is not used.

Not Used

Not Used

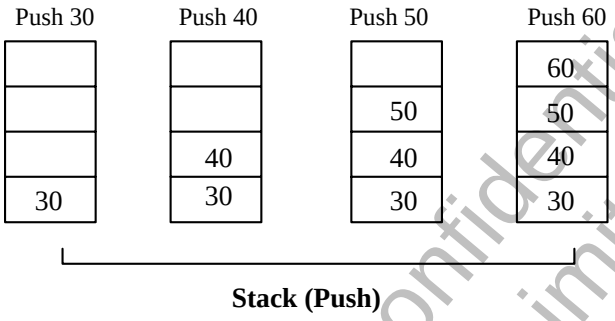
OverFlow Flag
If OverFlow is set, V=1
If OverFlow is clear, V=0

Negative Flag
If arithmetic or logic operation results to negative, N is set to 1. otherwise, N=0.

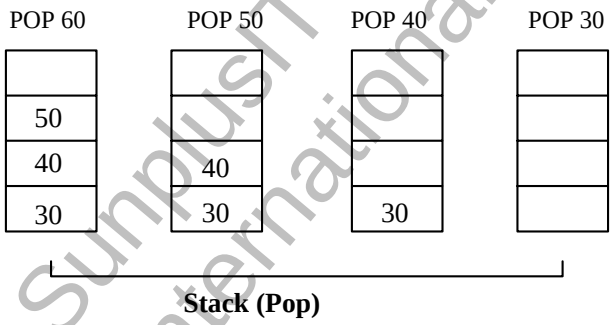
* Note: Not all instructions affect Status Register. A detailed instruction description will be discussed in later section.

Stack

In normal use, stack can be used as storing return address, temporary data or register's content. A stack has the property that the last item placed on the stack will be the first item removed. This property is commonly referred to as last in, first out, or simply **LIFO**. A diagram is shown as follows:



Stack (Push)



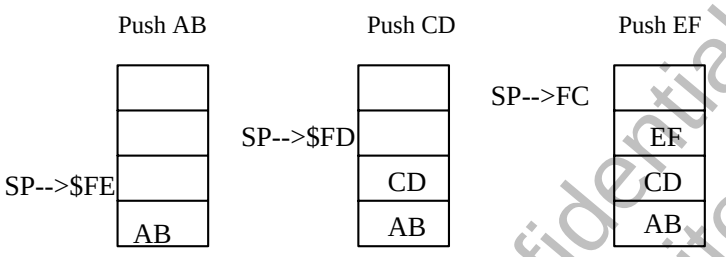
Stack (Pop)

In push activity, a value of 30 is pushed first. Then, a value of 40 is pushed. Thus, the value of 40 is now stored on the top on stack. After all values stored in the stack, the value order is 60, 50, 40, 30.

Now, in pop activity, the value of 60 will be popped out first. Second, the value of 50 will be popped. Then, 40 and 30 will be popped out in order. Stack is empty after all the values are popped.

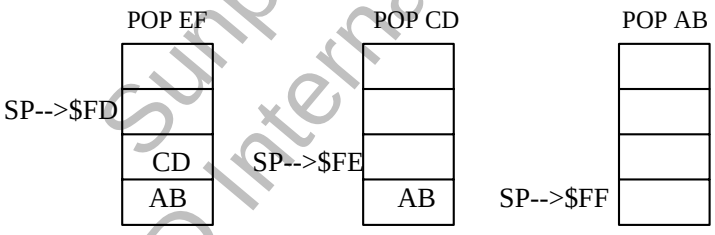
Stack Pointer (SP)

Stack Pointer is a pointer which usually points to an available location where can be stored pushed data. Normally, stack pointer is extended from FF to 00 in CPU 12. When data is pushed onto stack, stack pointer will decrease by 1. When data is pulled (popped) from stack, stack pointer is increased by 1.



Stack Pointer, Push

First of all, a data 0ABH is pushed onto stack; then, the stack pointer points to the address location $FE. Second, a data of 0CDH is pushed onto stack and the stack pointer then points to the address location $FD. Third, a data of 0EFH is pushed onto stack and the stack pointer is now pointing to the address location $FC.



Stack Pointer, Pop

In the pop activity, the stack pointer will be increased by 1 first; then stack pops the value of 0EFH. The stack pointer is now pointing to the address location $FD. When pop acts again, stack pointer will be increased by 1 again; then pops the value of 0CDH. At this moment, the stack pointer is pointing to the address location $FE. Finally, the stack pointer is increased by 1 and pops the value of 0ABH. Now, the stack pointer is pointing to the original address location $FF. Note that if stack now pops again, the stack pointer will point to location $00. This is an illegal stack activity since the bottom of stack is $FF.

Addressing Mode

Immediate addressing mode

There is one byte in an immediate addressing mode.

Operation: **OP-code** **#dd**

where #dd can be :

binary: #%00000001 or #00000001B

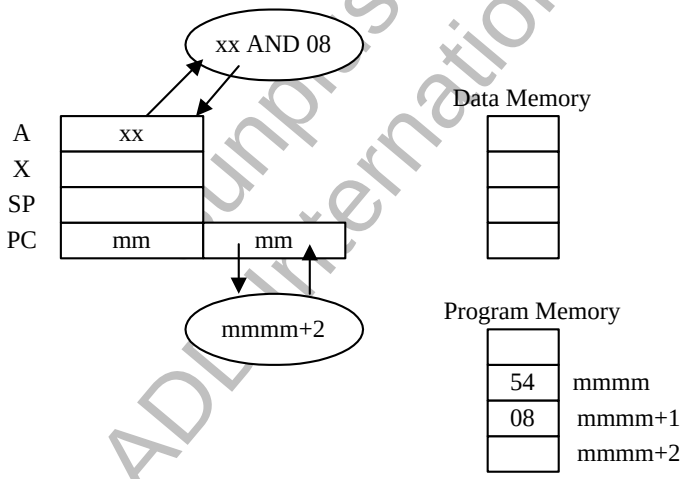
decimal: #01 or #01D

hexdecimal: #01H or #$01

Example:

AND #$08

N	V			D	I	Z	C
!	-			-	-	!	-



Example:

Given: A=7EH

AND #88H

Result:

88 AND 7E ➔ 08H

08H → A (08H stored in A)

Absolute addressing mode

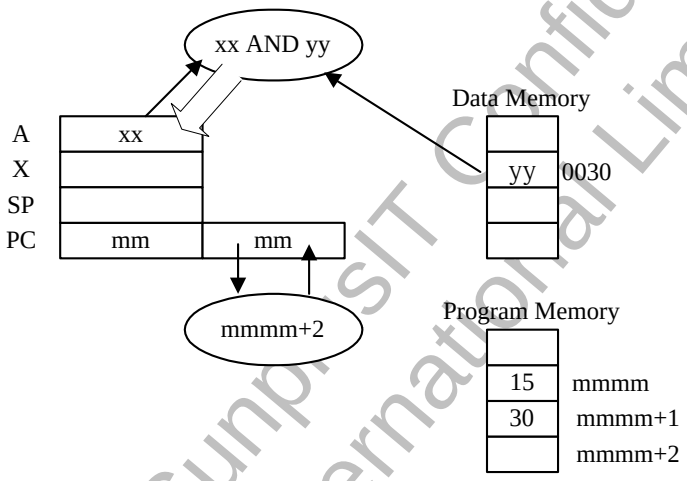
The absolute addressing mode uses two bytes (adr 16) to specify a memory address. The adr 16 may be the address of a byte of data or the beginning address for the next instruction.

Operation: **OP-code** **Adr16**

Example:

AND $0030

N	V			D	I	Z	C
!	-			-	-	!	-



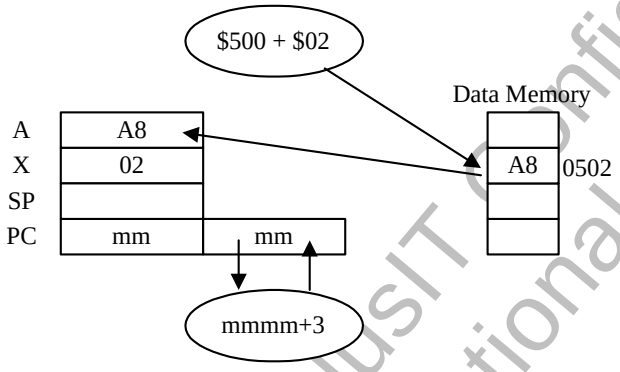
Absolute indexed addressing mode

The absolute indexed addressing mode uses two-part (adr 16 and X) to specify a memory address.

Operation : **OP-code** **Adr 16, X**

Example:
LDA $0500,X

N	V			D	I	Z	C
!	-			-	-	!	-



The new address is $500 + $02 = $502. This operation will copy the data of $502 to Accumulator. Therefore, Accumulator contains A8.

Zero Page Addressing Mode

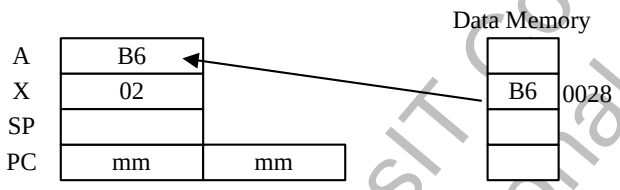
The zero page addressing mode uses the low-order byte of the address in page zero (adr 08) to specify a memory address.

Operation: **OP-Code** **Adr 08**

Example:

LDA $28

N	V			D	I	Z	C
!	-			-	-	!	-



Copy data from location $28 to Accumulator.

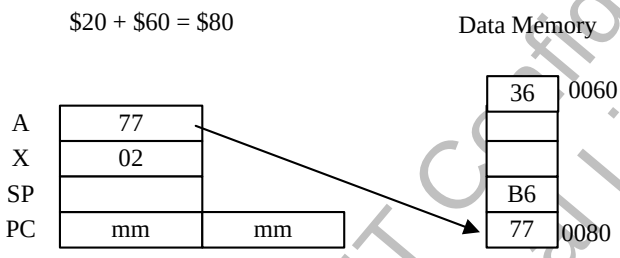
Zero Page Indexed addressing Mode

The zero page indexed addressing mode uses two-part (adr 08 and X) to specify a memory address.

Operation: **OP-Code** **Adr 08, X**

Example:

```
LDX     #$20
LDA     #$77
STA     $60, X
```



The new address = $60 + $20 = $80

Store #77H into $80.

Implied addressing mode

The implied addressing mode does not have any address.

Operation: **OP-Code**

Example:

```
TAX     ; To transfer data from accumulator to register X.
CLC     ; To clear carry
```

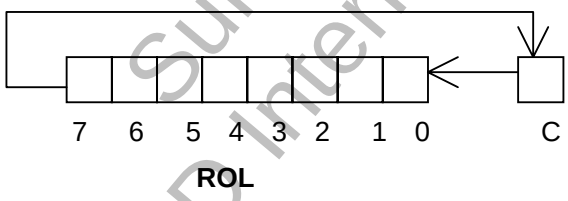
Accumulator addressing mode

The accumulator addressing mode does not have any address. The instruction operates on the data in the accumulator.

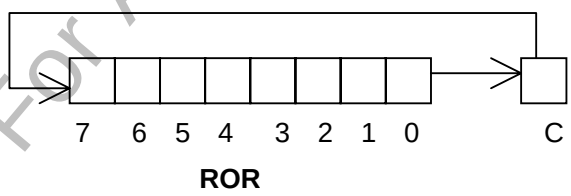
Operation: **OP-Code**

Example:

ROL Rotate Left with Carry
ROR Rotate Right with Carry



ROL



ROR

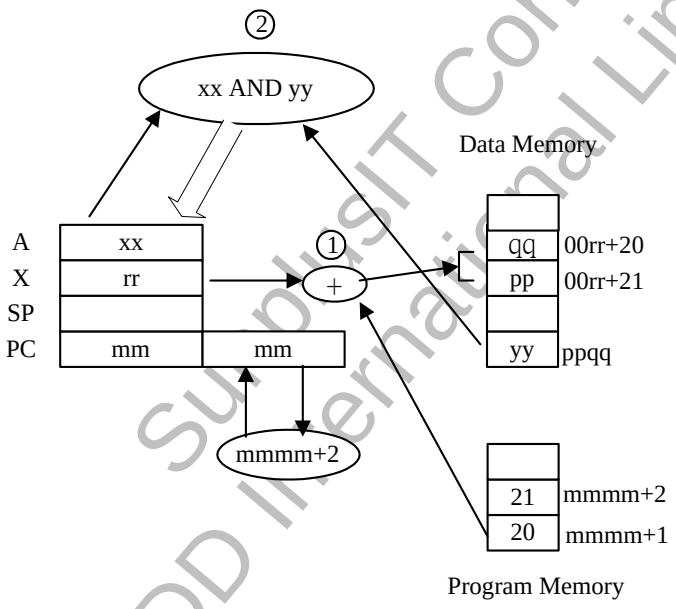
Indexed indirect addressing mode

The pre-indexed indirect addressing mode uses " (adr 08 and X)" to specify a memory address. Only register X can be used in this mode. The pre-indexed indirect address is a zero-page indexed direct address. Thus, the valid address must be on page zero.

Operation : **OP-Code (Adr 08, X)**

Example:
AND ($20, X)

N	V			D	I	Z	C
!	-			-	-	!	-



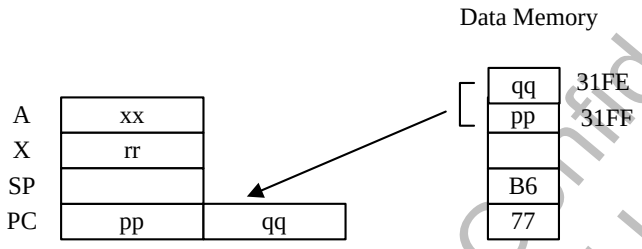
Indirect addressing mode

Index addressing mode can only use JMP instruction.

Operation: **JMP** **(Adr)**

Example:

JMP ($31FE)



PC=ppqq

Indirect Indexed addressing mode

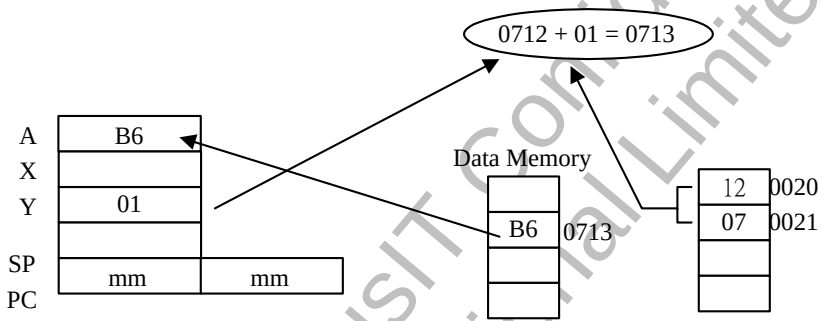
Indirect Indexed addressing mode can only be applied for Y index register.

Operation: Opcode **(aa), Y**

Example:

LDA ($20), Y

N	V			D	I	Z	C
!	-			-	-	!	-



Relative addressing mode

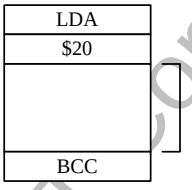
The relative addressing mode uses (adr 08) to specify a memory address. The relative addressing mode only uses with the branch instructions. The maximum branch forward is 127 bytes and the maximum branch backward is 128 bytes.

Operation: **OP-Code** **Adr 08**

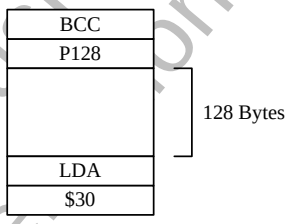
Example:

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

```
M127:   LDA     #$20
          .
          .
          .
        BCC     M127
```




```
        BCC     P128
          .
          .
          .
P128:   LDA     #$30
```



Format of Assembly Language Instruction

There are four parts of assembly language instruction.

[label :] OP-code [operand] [; comment]

[ ] : represents optional item.

Label field It labels an instruction. Programmers are able to use the label as an address. Some rules should be applied:

- Start in column 1 or use a colon (:) at the end of a label.
- Start with a letter.
- Do not use the name of OP-code or register.
- 1 to 32 characters
- Avoid special symbols

OP-Code field It is an instruction field.

Operand field It can be data or addresses used in the program. When OP-Code is a single byte, operand field is omitted. When the address mode is immediate, it is a byte of data. It is a symbol for a location where a byte of data is found. It is a label when it refers to a program address.

Comment field The comment field will increase the program's readability. A semicolon (;) should be placed at the beginning of comment.
For example:

```
LDA   #00        ; load data 00 to A
STA   Counter    ; load value of A into Counter
```

Note: A space is needed between two fields.

Instructions

ADC

Add to Accumulator with Carry, (A+M+C) → A, C

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	ADC #dd	69H	56H	2	2	2	2	2	2
Zero Page	ADC aa	65H	17H	2	3	3	3	3	3
Zero Page, X	ADC aa, X	75H	1FH	2	4	4	X	X	X
Absolute	ADC aaaa	6DH	57H	3	4	4	X	X	X
Absolute, X	ADC aaaa, X	7DH	5FH	3	4	4*	X	X	X
Absolute, Y	ADC aaaa, Y	79H	5EH	3	4	4*	X	X	X
(Indirect, X)	ADC (aa, X)	61H	16H	2	6	6	X	X	X
(Indirect), Y	ADC (aa), Y	71H	1EH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	!			*	-	!	!

N: Set if result is negative

V: Set if arithmetic overflow occurs.

Z: Set if result is 0

C: Set if there is a carry from the most significant bit of the result.

D: * if set to 1, the ADC performs decimal operation.

AND

AND memory data with Accumulator, (A^M) → A

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	AND #dd	29H	54H	2	2	2	2	2	2
Zero Page	AND aa	25H	15H	2	3	3	3	3	3
Zero Page, X	AND aa, X	35H	1DH	2	4	4	X	X	X
Absolute	AND aaaa	2DH	55H	3	4	4	X	X	X
Absolute, X	AND aaaa, X	3DH	5DH	3	4	4*	X	X	X
Absolute, Y	AND aaaa, Y	39H	5CH	3	4	4*	X	X	X
(Indirect, X)	AND (aa, X)	21H	14H	2	6	6	X	X	X
(Indirect), Y	AND (aa), Y	31H	1CH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

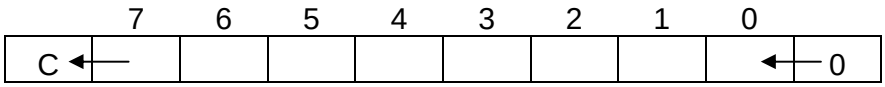
N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

ASL

Accumulator Shift Left



Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Accumulator	ASL A	0AH	C0H	1	2	2	X	X	X
Zero Page	ASL aa	06H	81H	2	5	5	X	X	X
Zero Page, X	ASL aa, X	16H	89H	2	6	6	X	X	X
Absolute	ASL aaaa	0EH	C1H	3	6	6	X	X	X
Absolute, X	ASL aaaa, X	1EH	C9H	3	6	6*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if the bit shifted from the most significant bit is 1 .

BCC/BCS/BEQ/BMI/BNE/BPL/BVC/BVS

Branch to aa if condition is true.

The range of relative addressing is –128 (backward) and +127 (forward) bytes.

Assembly Language Form	Condition	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
BCC aa	C=0	90H	28H	2	2***	2**	2**	2**	2**
BCS aa	C=1	B0H	38H	2	2***	2**	2**	2**	2**
BEQ aa	Z=1	F0H	3AH	2	2***	2**	2**	2**	2**
BMI aa	N=1	30H	18H	2	2***	2**	2**	2**	2**
BNE aa	Z=0	D0H	2AH	2	2***	2**	2**	2**	2**
BPL aa	N=0	10H	08H	2	2***	2**	2**	2**	2**
BVC aa	V=0	50H	0AH	2	2***	2**	2**	2**	2**
BVS aa	V=1	70H	1AH	2	2***	2**	2**	2**	2**

** Add 1 clock cycle if branch occurs to the same page.
Add 2 clock cycles if branch occurs to different page.

*** Add 1 clock cycle if branch occurs.

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

BIT

Test bit in memory with Accumulator

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	BIT aa	24H	11H	2	3	3	3	X	3
Absolute	BIT aaaa	2CH	51H	3	4	4	4	X	4

X: Not available.

N	V			D	I	Z	C
!	!			-	-	!	-

N: Set if memory bit7 of the result is 1

V: Set if memory bit 6 of the result is 1.

Z: Set if result is 0

CLC

Clear Carry flag

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	CLC	18H	48H	1	2	2	2	2	2

N	V			D	I	Z	C
-	-			-	-	-	!

C: Unconditionally cleared.

CLD

Clear Decimal mode

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	CLD	D8H	6AH	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
-	-			!	-	-	-

D: Unconditionally cleared.

CLI

Clear Interrupt mask. (enable interrupt)

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	CLI	58H	4AH	1	2	2	2	2	2

N	V			D	I	Z	C
-	-			-	!	-	-

I: Unconditionally cleared.

CLR

Bit Clear.

Clear BITn of $aa as “0”.

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	CLR aa, 0	0FH	C5H	2	5	X	X	X	X
Zero Page	CLR aa, 1	1FH	CDH	2	5	X	X	X	X
Zero Page	CLR aa, 2	2FH	D5H	2	5	X	X	X	X
Zero Page	CLR aa, 3	3FH	DDH	2	5	X	X	X	X
Zero Page	CLR aa, 4	4FH	C7H	2	5	X	X	X	X
Zero Page	CLR aa, 5	5FH	CFH	2	5	X	X	X	X
Zero Page	CLR aa, 6	6FH	D7H	2	5	X	X	X	X
Zero Page	CLR aa, 7	7FH	DFH	2	5	X	X	X	X

X: Not available.

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

CLV

Clear overflow

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	CLV	B8H	78H	1	2	2	2	2	2

N	V			D	I	Z	C
-	!			-	-	-	-

V: Unconditionally cleared.

CMP

Compare memory data with Accumulator, A - M

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	CMP #dd	C9H	66H	2	2	2	2	2	2
Zero Page	CMP aa	C5H	27H	2	3	3	3	3	3
Zero Page, X	CMP aa, X	D5H	2FH	2	4	4	4	4	4
Absolute	CMP aaaa	CDH	67H	3	4	4	X	X	X
Absolute, X	CMP aaaa, X	DDH	6FH	3	4	4*	X	X	X
Absolute, Y	CMP aaaa, Y	D9H	6EH	3	4	4*	X	X	X
(Indirect, X)	CMP (aa, X)	C1H	26H	2	6	6	X	X	X
(Indirect), Y	CMP (aa), Y	D1H	2EH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if a “borrow” not occurred. (A > = M)

CPX

Compare memory data with Register X, X - data

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	CPX #dd	E0H	32H	2	2	2	2	2	2
Zero Page	CPX aa	E4H	33H	2	3	3	3	3	3
Absolute	CPX aaaa	ECH	73H	3	4	4	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if a “borrow” not occurred. (X > = data)

CPY

Compare memory data with Register Y, Y - data

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	CPY #dd	C0H	22H	2	2	2	X	X	X
Zero Page	CPY aa	C4H	23H	2	3	3	X	X	X
Absolute	CPY aaaa	CCH	63H	3	4	4	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if a “borrow” not occurred (Y > = data)

DEC

Decrement memory by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	DEC aa	C6H	A3H	2	5	5	5	5	5
Zero Page, X	DEC aa, X	D6H	ABH	2	6	6	6	X	6
Absolute	DEC aaaa	CEH	E3H	3	6	6	X	X	X
Absolute, X	DEC aaaa, X	DEH	EBH	3	6	7	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

DEX

Decrement Register X by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	DEX	CAH	E2H	1	2	2	2	2	2

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

DEY

Decrement Register Y by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	DEY	88H	60H	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

EOR

Exclusive-OR memory with Accumulator, A ← A XOR memory

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	EOR #dd	49H	46H	2	2	2	2	2	2
Zero Page	EOR aa	45H	07H	2	3	3	3	3	3
Zero Page, X	EOR aa, X	55H	0FH	2	4	4	4	X	4
Absolute	EOR aaaa	4DH	47H	3	4	4	X	X	X
Absolute, X	EOR aaaa, X	5DH	4FH	3	4	4*	X	X	X
Absolute, Y	EOR aaaa, Y	59H	4EH	3	4	4*	X	X	X
(Indirect, X)	EOR (aa, X)	41H	06H	2	6	6	X	X	X
(Indirect), Y	EOR (aa), Y	51H	0EH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

INC

Increment memory by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	INC aa	E6H	B3H	2	5	5	5	5	5
Zero Page, X	INC aa, X	F6H	BBH	2	6	6	X	X	X
Absolute	INC aaaa	EEH	F3H	3	6	6	X	X	X
Absolute, X	INC aaaa, X	FEH	FBH	3	6	7	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

INV

Bit Inverse.

Toggle BITn of $aa.

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	INV aa, 0	87H	A5H	2	5	X	X	X	X
Zero Page	INV aa, 1	97H	ADH	2	5	X	X	X	X
Zero Page	INV aa, 2	A7H	B5H	2	5	X	X	X	X
Zero Page	INV aa, 3	B7H	BDH	2	5	X	X	X	X
Zero Page	INV aa, 4	C7H	A7H	2	5	X	X	X	X
Zero Page	INV aa, 5	D7H	AFH	2	5	X	X	X	X
Zero Page	INV aa, 6	E7H	B7H	2	5	X	X	X	X
Zero Page	INV aa, 7	F7H	BFH	2	5	X	X	X	X

X: Not available.

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

INX

Increment Register X by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	INX	E8H	72H	1	2	2	2	2	2

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

INY

Increment Register Y by one

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	INY	C8H	62H	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

JMP

Jump to specified location

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Absolute	JMP aaaa	4CH	43H	3	3	3	3	3	3
Indirect	JMP (aaaa)	6CH	53H	3	5	6	6	6	6

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

JSR

Jump to subroutine

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Absolute	JSR aaaa	20H	10H	3	6	6	6	6	6

With JSR instruction, the current address will be pushed on stack and then jumps to the specified subroutine. At the end of subroutine procedure, the RTS (return from subroutine) instruction can be used to return to the original program flow by popping saved address from stack.

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

LDA

Load memory data or data into Accumulator, A ← data

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	LDA #dd	A9H	74H	2	2	2	2	2	2
Zero Page	LDA aa	A5H	35H	2	3	3	3	3	3
Zero Page, X	LDA aa, X	B5H	3DH	2	4	4	4	4	4
Absolute	LDA aaaa	ADH	75H	3	4	4	4	4	4
Absolute, X	LDA aaaa, X	BDH	7DH	3	4	4*	4*	4*	4*
Absolute, Y	LDA aaaa, Y	B9H	7CH	3	4	4*	X	X	X
(Indirect, X)	LDA (aa, X)	A1H	34H	2	6	6	6	6	6
(Indirect), Y	LDA (aa), Y	B1H	3CH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

LDX

Load memory data or data into Register X, X ← data

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	LDX #dd	A2H	B0H	2	2	2	2	2	2
Zero Page	LDX aa	A6H	B1H	2	3	3	3	3	3
Zero Page, Y	LDX aa, Y	B6H	B9H	2	4	4	X	X	X
Absolute	LDX aaaa	AEH	F1H	3	4	4	4	X	4
Absolute, Y	LDX aaaa, Y	BEH	F9H	3	4	4*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

LDY

Load memory data or data into Register Y, Y ← data

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	LDY #dd	A0H	30H	2	2	2	X	X	X
Zero Page	LDY aa	A4H	31H	2	3	3	X	X	X
Zero Page, X	LDY aa, X	B4H	39H	2	4	4	X	X	X
Absolute	LDY aaaa	ACH	71H	3	4	4	X	X	X
Absolute, X	LDY aaaa, X	BCH	79H	3	4	4*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

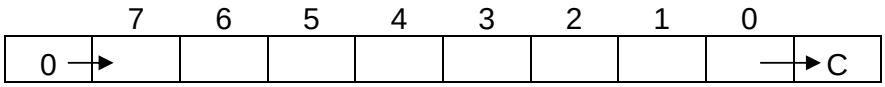
N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

LSR

Local Shift Right



Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Accumulator	LSR A	4AH	C2H	1	2	2	X	X	X
Zero Page	LSR aa	46H	83H	2	5	5	X	X	X
Zero Page, X	LSR aa, X	56H	8BH	2	6	6	X	X	X
Absolute	LSR aaaa	4EH	C3H	3	6	6	X	X	X
Absolute, X	LSR aaaa, X	5EH	CBH	3	6	6*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if the bit shifted from the least significant bit is 1.

NOP

No operation

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	NOP	EAH	F2H	1	2	2	2	2	2

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

ORA

OR memory with Accumulator, A ← A OR memory

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	ORA #dd	09H	44H	2	2	2	2	2	2
Zero Page	ORA aa	05H	05H	2	3	3	3	3	3
Zero Page, X	ORA aa, X	15H	0DH	2	4	4	X	X	X
Absolute	ORA aaaa	0DH	45H	3	4	4	X	X	X
Absolute, X	ORA aaaa, X	1DH	4DH	3	4	4*	X	X	X
Absolute, Y	ORA aaaa, Y	19H	4CH	3	4	4*	X	X	X
(Indirect, X)	ORA (aa, X)	01H	04H	2	6	6	X	X	X
(Indirect), Y	ORA (aa), Y	11H	0CH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if result is negative

Z: Set if result is 0

PHA

Push Accumulator on Stack

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	PHA	48H	42H	1	3	3	3	3	3

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

PHP

Push Status Flag on Stack

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	PHP	08H	40H	1	3	3	3	3	3

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

PLA

Pull Accumulator from Stack

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	PLA	68H	52H	1	4	4	4	4	4

N	V			D	I	Z	C
!	-			-	-	!	-

PLP

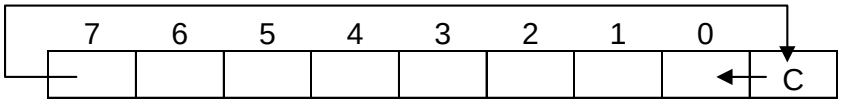
Pull Status Flag from Stack

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	PLP	28H	50H	1	4	4	4	4	4

N	V			D	I	Z	C
!	!			!	!	!	!

ROL

Rotate Left



Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Accumulator	ROL A	2AH	D0H	1	2	2	2	2	2
Zero Page	ROL aa	26H	91H	2	5	5	5	5	5
Zero Page, X	ROL aa, X	36H	99H	2	6	6	X	X	X
Absolute	ROL aaaa	2EH	D1H	3	6	6	X	X	X
Absolute, X	ROL aaaa, X	3EH	D9H	3	6	6*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

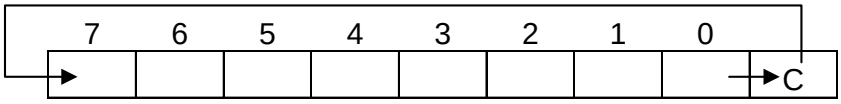
N: Set if result is negative

Z: Set if result is 0

C: Set if the bit shifted from the most significant bit position is 1.

ROR

Rotate Right



Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Accumulator	ROR A	6AH	D2H	1	2	2	2	2	2
Zero Page	ROR aa	66H	93H	2	5	5	5	5	5
Zero Page, X	ROR aa, X	76H	9BH	2	6	6	X	X	X
Absolute	ROR aaaa	6EH	D3H	3	6	6	X	X	X
Absolute, X	ROR aaaa, X	7EH	DBH	3	6	6*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	!

N: Set if result is negative

Z: Set if result is 0

C: Set if the bit shifted from the least significant bit position is 1.

RTI

Return from Interrupt

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	RTI	40H	02H	1	6	6	6	6	6

N	V			D	I	Z	C

N: Restored from stack

V: Restored from stack

D, I: Restored from stack

Z: Restored from stack

C: Restored from stack

RTS

Return from Subroutine

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	RTS	60H	12H	1	6	6	6	6	6

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

SBC

Subtract from Accumulator with Carry's complement, (A-M - $\overline{C}$) → A, C

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Immediate	SBC #dd	E9H	76H	2	2	2	2	2	2
Zero Page	SBC aa	E5H	37H	2	3	3	3	3	3
Zero Page, X	SBC aa, X	F5H	3FH	2	4	4	X	X	X
Absolute	SBC aaaa	EDH	77H	3	4	4	X	X	X
Absolute, X	SBC aaaa, X	FDH	7FH	3	4	4*	X	X	X
Absolute, Y	SBC aaaa, Y	F9H	7EH	3	4	4*	X	X	X
(Indirect, X)	SBC (aa, X)	E1H	36H	2	6	6	X	X	X
(Indirect), Y	SBC (aa), Y	F1H	3EH	2	6	5*	X	X	X

* Add 1 clock cycle if page boundary is crossed.

X: Not available.

N	V			D	I	Z	C
!	!			*	-	!	!

N: Set if result is negative

V: Set if arithmetic overflow occurs.

Z: Set if result is 0

C: Set if there is no "borrow" occurred. (A > M).

D: * if set to 1, the ADC performs decimal operation.

SEC

Set Carry Flag to 1, C ←1

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	SEC	38H	58H	1	2	2	2	2	2

N	V			D	I	Z	C
-	-			-	-	-	!

C: Unconditionally Set

SED

Set Decimal Mode to 1, D ←1

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	SED	F8H	7AH	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
-	-			!	-	-	-

D: Unconditionally Set

SEI

Set Interrupt Disable flag to 1, I ←1 (Disable Interrupt)

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	SEI	78H	5AH	1	2	2	2	2	2

N	V			D	I	Z	C
-	-			-	!	-	-

I: Unconditionally Set

SET

Bit Set.

Set BITn of $aa as “1”.

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	SET aa, 0	8FH	E5H	2	5	X	X	X	X
Zero Page	SET aa, 1	9FH	EDH	2	5	X	X	X	X
Zero Page	SET aa, 2	AFH	F5H	2	5	X	X	X	X
Zero Page	SET aa, 3	BFH	FDH	2	5	X	X	X	X
Zero Page	SET aa, 4	CFH	E7H	2	5	X	X	X	X
Zero Page	SET aa, 5	DFH	EFH	2	5	X	X	X	X
Zero Page	SET aa, 6	EFH	F7H	2	5	X	X	X	X
Zero Page	SET aa, 7	FFH	FFH	2	5	X	X	X	X

X: Not available.

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

STA

Store Accumulator in memory, M ← A

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	STA aa	85H	25H	2	3	3	3	3	3
Zero Page, X	STA aa, X	95H	2DH	2	4	4	4	4	4
Absolute	STA aaaa	8DH	65H	3	4	4	X	X	X
Absolute, X	STA aaaa, X	9DH	6DH	3	4	5	X	X	X
Absolute, Y	STA aaaa, Y	99H	6CH	3	4	5	X	X	X
(Indirect, X)	STA (aa, X)	81H	24H	2	6	6	6	6	6
(Indirect), Y	STA (aa), Y	91H	2CH	2	6	6	X	X	X

X: Not available.

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

STX

Store Register X in memory, M ← X

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	STX aa	86H	A1H	2	3	3	3	3	3
Zero Page, Y	STX aa, Y	96H	A9H	2	4	4	X	X	X
Absolute	STX aaaa	8EH	E1H	3	4	4	4	4	4

X: Not available.

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

STY

Store Register Y in memory, M ← Y

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	STY aa	84H	21H	2	3	3	X	X	X
Zero Page, X	STY aa, X	94H	29H	2	4	4	X	X	X
Absolute	STY aaaa	8CH	61H	3	4	4	X	X	X

X: Not available.

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

TAX

Transfer Accumulator to Index X, X ← A

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TAX	AAH	F0H	1	2	2	2	X	2

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if the result is negative

Z: Set if the result is 0

TAY

Transfer Accumulator to Index Y, Y ← A

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TAY	A8H	70H	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if the result is negative

Z: Set if the result is 0

TST

Bit Test.

Read and judge BITn of $aa.

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Zero Page	TST aa, 0	07H	85H	2	3	X	X	X	X
Zero Page	TST aa, 1	17H	8DH	2	3	X	X	X	X
Zero Page	TST aa, 2	27H	95H	2	3	X	X	X	X
Zero Page	TST aa, 3	37H	9DH	2	3	X	X	X	X
Zero Page	TST aa, 4	47H	87H	2	3	X	X	X	X
Zero Page	TST aa, 5	57H	8FH	2	3	X	X	X	X
Zero Page	TST aa, 6	67H	97H	2	3	X	X	X	X
Zero Page	TST aa, 7	77H	9FH	2	3	X	X	X	X

X: Not available.

N	V			D	I	Z	C
-	-			-	-	!	-

Z: Set if BITn of $aa is 0.

TSX

Transfer Stack to Index X, X ← S

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TSX	BAH	F8H	1	2	2	2	2	2

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if the result is negative

Z: Set if the result is 0

TXA

Transfer Register X to Accumulator, A ← X

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TXA	8AH	E0H	1	2	2	2	X	2

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if the result is negative

Z: Set if the result is 0

TXS

Transfer Register X to Stack, S ← X

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TXS	9AH	E8H	1	2	2	2	2	2

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

TYA

Transfer Register Y to Accumulator, A ← Y

Addressing mode	Assembly Language Form	6502 Opcode	Sunplus Opcode	No. Bytes	Available Instruction & No. Cycles				
					65b02	65n02	65r02	65s02	CPU12
Implied	TYA	98H	68H	1	2	2	X	X	X

X: Not available.

N	V			D	I	Z	C
!	-			-	-	!	-

N: Set if the result is negative

Z: Set if the result is 0

Summary of Available Instruction set for each CPU Type

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
1.	ADC #dd	Immediate	√	√	√	√	√
2.	ADC aa	Zero page	√	√	√	√	√
3.	ADC aa, X	Zero page Indexed X	√	√			
4.	ADC aaaa	Absolute	√	√			
5.	ADC aaaa,X	Absolute Indexed X	√	√			
6.	ADC aaaa,Y	Absolute Indexed Y	√	√			
7.	ADC (aa,X)	Indexed Indirect X	√	√			
8.	ADC (aa), Y	Indirect Indexed Y	√	√			
9.	AND #dd	Immediate	√	√	√	√	√
10.	AND aa	Zero page	√	√	√	√	√
11.	AND aa, X	Zero page Indexed X	√	√			
12.	AND aaaa	Absolute	√	√			
13.	AND aaaa,X	Absolute Indexed X	√	√			
14.	AND aaaa,Y	Absolute Indexed Y	√	√			
15.	AND (aa,X)	Indexed Indirect X	√	√			
16.	AND (aa), Y	Indirect Indexed Y	√	√			
17.	ASL A	accumulator	√	√			
18.	ASL aa	Zero page	√	√			
19.	ASL aa,X	Zero page Indexed x	√	√			
20.	ASL aaaa	Absolute	√	√			
21.	ASL aaaa,X	Absolute Indexed x	√	√			
22.	BCC ??	Relative	√	√	√	√	√

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
23.	BCS ??	Relative	√	√	√	√	√
24.	BEQ ??	Relative	√	√	√	√	√
25.	BIT aa	Zero page	√	√	√		√
26.	BIT aaaa	Absolute	√	√	√		√
27.	BMI ??	Relative	√	√	√	√	√
28.	BNE ??	Relative	√	√	√	√	√
29.	BPL ??	Relative	√	√	√	√	√
30.	BRK	Implied	√	√	√	√	√
31.	BVC ??	Relative	√	√	√	√	√
32.	BVS ??	Relative	√	√	√	√	√
33.	CLC	Implied	√	√	√	√	√
34.	CLD	Implied	√	√			
35.	CLI	Implied	√	√	√	√	√
36.	CLR aa, BITn	Zero page	√				
37.	CLV	Implied	√	√	√	√	√
38.	CMP #dd	Immediate	√	√	√	√	√
39.	CMP aa	Zero page	√	√	√	√	√
40.	CMP aa, X	Zero page Indexed X	√	√	√	√	√
41.	CMP aaaa	Absolute	√	√			
42.	CMP aaaa,X	Absolute Indexed X	√	√			
43.	CMP aaaa,Y	Absolute Indexed Y	√	√			
44.	CMP (aa,X)	Indexed Indirect X	√	√			
45.	CMP (aa), Y	Indirect Indexed Y	√	√			
46.	CPX #dd	Immediate	√	√	√	√	√
47.	CPX aa	Zero page	√	√	√	√	√
48.	CPX aaaa	Absolute	√	√			
49.	CPY #dd	Immediate	√	√			
50.	CPY aa	Zero page	√	√			
51.	CPY aaaa	Absolute	√	√			
52.	DEC aa	Zero page	√	√	√	√	√

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
53.	DEC aa, X	Zero page Indexed X	√	√	√		√
54.	DEC aaaa	Absolute	√	√			
55.	DEC aaaa,X	Absolute Indexed X	√	√			
56.	DEX	Implied	√	√	√	√	√
57.	DEY	Implied	√	√			
58.	EOR #dd	Immediate	√	√	√	√	√
59.	EOR aa	Zero page	√	√	√	√	√
60.	EOR aa, X	Zero page Indexed X	√	√	√		√
61.	EOR aaaa	Absolute	√	√			
62.	EOR aaaa,X	Absolute Indexed X	√	√			
63.	EOR aaaa,Y	Absolute Indexed Y	√	√			
64.	EOR (aa,X)	Indexed Indirect X	√	√			
65.	EOR (aa), Y	Indirect Indexed Y	√	√			
66.	INC aa	Zero page	√	√	√	√	√
67.	INC aa, X	Zero page Indexed X	√	√			
68.	INC aaaa	Absolute	√	√			
69.	INC aaaa,X	Absolute Indexed X	√	√			
70.	INV aa, BITn	Zero page	√				
71.	INX	Implied	√	√	√	√	√
72.	INY	Implied	√	√			
73.	JMP aaaa	Absolute	√	√	√	√	√
74.	JMP (aaaa)	Indirect absolute	√	√	√	√	√
75.	JSR aaaa	Absolute	√	√	√	√	√
76.	LDA #dd	Immediate	√	√	√	√	√
77.	LDA aa	Zero page	√	√	√	√	√
78.	LDA aa, X	Zero page Indexed X	√	√	√	√	√

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
79.	LDA aaaa	Absolute	√	√	√	√	√
80.	LDA aaaa,X	Absolute Indexed X	√	√	√	√	√
81.	LDA aaaa,Y	Absolute Indexed Y	√	√			
82.	LDA (aa,X)	Indexed Indirect X	√	√	√	√	√
83.	LDA (aa), Y	Indirect Indexed Y	√	√			
84.	LDX #dd	Immediate	√	√	√	√	√
85.	LDX aa	Zero page	√	√	√	√	√
86.	LDX aa, Y	Zero page Indexed Y	√	√			
87.	LDX aaaa	Absolute	√	√	√		√
88.	LDX aaaa,Y	Absolute Indexed Y	√	√			
89.	LDY #dd	Immediate	√	√			
90.	LDY aa	Zero page	√	√			
91.	LDY aa, X	Zero page Indexed X	√	√			
92.	LDY aaaa	Absolute	√	√			
93.	LDY aaaa,X	Absolute Indexed X	√	√			
94.	LSR A	Accumulator	√	√			
95.	LSR aa	Zero page	√	√			
96.	LSR aa, X	Zero page Indexed X	√	√			
97.	LSR aaaa	Absolute	√	√			
98.	LSR aaaa,X	Absolute Indexed X	√	√			
99.	NOP	Implied	√	√	√	√	√
100.	ORA #dd	Immediate	√	√	√	√	√
101.	ORA aa	Zero page	√	√	√	√	√
102.	ORA aa, X	Zero page Indexed X	√	√			
103.	ORA aaaa	Absolute	√	√			

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
104.	ORA aaaa,X	Absolute Indexed X	√	√			
105.	ORA aaaa,Y	Absolute Indexed Y	√	√			
106.	ORA (aa,X)	Indexed Indirect X	√	√			
107.	ORA (aa), Y	Indirect Indexed Y	√	√			
108.	PHA	Implied	√	√	√	√	√
109.	PHP	Implied	√	√	√	√	√
110.	PLA	Implied	√	√	√	√	√
111.	PLP	Implied	√	√	√	√	√
112.	ROL A	Accumulator	√	√	√	√	√
113.	ROL aa	Zero page	√	√	√	√	√
114.	ROL aa, X	Zero page Indexed X	√	√			
115.	ROL aaaa	Absolute	√	√			
116.	ROL aaaa,X	Absolute Indexed X	√	√			
117.	ROR A	Accumulator	√	√	√	√	√
118.	ROR aa	Zero page	√	√	√	√	√
119.	ROR aa, X	Zero page Indexed X	√	√			
120.	ROR aaaa	Absolute	√	√			
121.	ROR aaaa,X	Absolute Indexed X	√	√			
122.	RTI	Implied	√	√	√	√	√
123.	RTS	Implied	√	√	√	√	√
124.	SBC #dd	Immediate	√	√	√	√	√
125.	SBC aa	Zero page	√	√	√	√	√
126.	SBC aa, X	Zero page Indexed X	√	√			
127.	SBC aaaa	Absolute	√	√			
128.	SBC aaaa,X	Absolute Indexed X	√	√			

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
129.	SBC aaaa,Y	Absolute Indexed Y	√	√			
130.	SBC (aa,X)	Indexed Indirect X	√	√			
131.	SBC (aa), Y	Indirect Indexed Y	√	√			
132.	SEC	Implied	√	√	√	√	√
133.	SED	Implied	√	√			
134.	SEI	Implied	√	√	√	√	√
135.	SET aa, BITn	Zero page	√				
136.	STA aa	Zero page	√	√	√	√	√
137.	STA aa, X	Zero page Indexed X	√	√	√	√	√
138.	STA aaaa	Absolute	√	√			
139.	STA aaaa,X	Absolute Indexed X	√	√			
140.	STA aaaa,Y	Absolute Indexed Y	√	√			
141.	STA (aa,X)	Indexed Indirect X	√	√	√	√	√
142.	STA (aa), Y	Indirect Indexed Y	√	√			
143.	STX aa	Zero page	√	√	√	√	√
144.	STX aa, Y	Zero page Indexed Y	√	√			
145.	STX aaaa	Absolute	√	√	√	√	√
146.	STY aa	Zero page	√	√			
147.	STY aa, X	Zero page Indexed X	√	√			
148.	STY aaaa	Absolute	√	√			
149.	TAX	Implied	√	√	√		√
150.	TAY	Implied	√	√			
151.	TST aa, BITn	Zero page	√				
152.	TSX	Implied	√	√	√	√	√
153.	TXA	Implied	√	√	√		√
154.	TXS	Implied	√	√	√	√	√

No.	Instruction	Address Mode	65b02	65n02	65r02	65s02	CPU12
155.	TYA	Implied	√	√			

Click below to find more

Mipaper at www.lcis.com.tw