6502 Addressing Modes

Implied/Implicit   **IMPLIED**

The operand is implicitly defined by the instruction.

```
INX             increment the X register by 1
```

Accumulator   **A**   **ACCUM.**

The operand **A** is implicitly defined as the accumulator with a 1 byte instruction.

```
ROL             rotate the contents of the accumulator left one bit
```

Immediate   **#$aa**   **IMM**

The operand **#$aa** is 1 byte of data.
The # symbol is used to distinguish data from an address/memory location.

```
LDA #$48
will load the accumulator with the value $48 => PETSCII/ASCII code for 'H'
```

Absolute   **$aaaa**   **ABS**

The operand **$aaaa** is the 2 byte effective address (EA).
The first byte of the operand in machine language is the least significant byte (LSB bits 0-7) of the EA.
The second byte of the operand in machine language is the most significant byte (MSB bits 8-15) of the EA.

```
if address $3491 contains $5C , then
LDA $3491
will load the accumulator with the contents of EA $3491 => $5C
```

Zero Page   **$aa**   **Z.PAGE**

The operand **$aa** is the LSB of the EA.
The MSB of the EA is $00. The EA is in the address range $0000–$00FF (zero page).

```
if address $0056 ($56) contains $1D , then
LDA $56
will load the accumulator with the contents of EA $0056 => $1D
```

Absolute Indexed,X $aaaa,X   ABS,X
Absolute Indexed,Y $aaaa,Y   ABS,Y

The operand **$aaaa** is added to the index register (X or Y), the result **$aaaa + X** or **Y** is the EA.

```
if LDX #$A3      load the X register with the value  $A3 , and
   LDY #$81      load the Y register with the value  $81 , and
   STY $2279     store the contents of the Y rgsr => $81 in address $2279 ,then

   LDA $21D6,X  will load the contents of $aaaa + X => ($21D6 + $A3) => $2279
                                                    => EA $81 into the accum.
```

Zero Page Indexed,X $aa,X   Z.PAGE,X
Zero Page Indexed,Y $aa,Y   Z.PAGE,Y

The operand **$aa** is added with no carry (NC) to the index register (X or Y),
the result   **$aa + X** or **Y (NC)** is the LSB of the EA. The MSB of the EA is $00. The EA is in zero page.

```
if LDX #$E9      load the X register with the value $E9 , then
   LDA $51,X     will result in a zero page EA of
  $aa + X (NC) => ($51 + $E9) (NC) => $13A (NC)
                               =>  $3A => EA $003A
if              EA $003A contains $04 , then
                $04 will load into the accumulator
```

Absolute Indirect   ($aaaa) INDERECT

The operand   **$aaaa**   contains the LSB of the EA.
The address   **$aaaa + $0001**   contains the MSB of the EA.
The 2 byte address   **(($aaaa + $0001) ∗ $100) + $aaaa** contains the EA.
JMP is the only instruction to use this addressing mode.

if **$aaaa**   => $0237   contains $31 => EA LSB , and
   $aaaa + $0001 => ($0237 + $0001) => $0238 contains $88 => EA MSB ,
 (($aaaa + $0001) ∗ $100) + $aaaa   => ($88 ∗ $100) + $31 => EA $8831 , then

JMP ($0237)
will jump the Program Counter (PC) to ($88 ∗ $100) + $31 => EA $8831

Indexed Indirect   ($aa,X) (IND,X)

The operand   **$aa** is added (NC) to the X register,
the result is a zero page address   **$aa + X (NC)** that contains the LSB of the EA.
The address   **$aa + X + $01** contains the MSB of the EA.

if LDX #$E9   load the X register with the value $E9 , and
   LDY #$81   load the Y register with the value $81 , and
   STY $3104   store the contents of the Y register => $81 in address $3104 ,then

   LDA ($51,X) will result in a zero page address of
   $aa + X (NC) => ($51 + $E9) (NC) => $13A (NC)
   $aa + X   => $3A

if **$aa + X**   => $3A contains $04 => EA LSB , and
   $aa + X + $01 => ($3A + $01)   => $3B contains $31 => EA MSB ,
                                             => EA $3104 , then

the accumulator will be loaded with the contents of EA $3104 => $81

Indirect Indexed   ($aa),Y (IND),Y

The operand **$aa** is a zero page address, the contents of **$aa** are added with carry (C) to the Y register
                 $aa + Y (C), the result contains the LSB of the EA.
The contents of address   **$aa + $01 + C** contain the MSB of the EA.

if LDY #$E9   load the Y register with the value $E9 , and
   LDX #$81   load the X register with the value $81 , and
   STX $3104   store the contents of the X register => $81 in address $3104 , and

if   address $A4   contains $51 , and
                 address ($A4 + $01) => $A5 contains $3F , then

LDA ($A4),Y   results in

$aa + Y (C)   => ($51 + $E9) (C) => $13A (C)   => $3A => EA LSB ,and
$aa + $01 + C => ($3F + C)   => ($3F + $01) => $40 => EA MSB
                                                      => EA $403A
if EA $403A contains $BB , then
the accumulator will be loaded with the contents of EA $403E => $BB

Relative $aa   REL

The operand **$aa** is added to the LSB of the PC as an offset, with a range of $−80 to $+7F.
Absolute addresses are converted to relative addresses by the assembler.
Branch instructions are the only instructions to use this addressing mode.