The Registers

The 6502 has only a small number of registers compared to other processor of the same era. This makes it especially challenging to program as algorithms must make efficient use of both registers and memory.

Program Counter

The program counter is a 16 bit register which points to the next instruction to be executed. The value of program counter is modified automatically as instructions are executed.

The value of the program counter can be modified by executing a jump, a relative branch or a subroutine call to another memory address or by returning from a subroutine or interrupt.

Stack Pointer

The processor supports a 256 byte stack located between $0100 and $01FF. The stack pointer is an 8 bit register and holds the low 8 bits of the next free location on the stack. The location of the stack is fixed and cannot be moved.

Pushing bytes to the stack causes the stack pointer to be decremented. Conversely pulling bytes causes it to be incremented.

The CPU does not detect if the stack is overflowed by excessive pushing or pulling operations and will most likely result in the program crashing.

Accumulator

The 8 bit accumulator is used all arithmetic and logical operations (with the exception of increments and decrements). The contents of the accumulator can be stored and retrieved either from memory or the stack.

Most complex operations will need to use the accumulator for arithmetic and efficient optimisation of its use is a key feature of time critical routines.

Index Register X

The 8 bit index register is most commonly used to hold counters or offsets for accessing memory. The value of the X register can be loaded and saved in memory, compared with values held in memory or incremented and decremented.

The X register has one special function. It can be used to get a copy of the stack pointer or change its value.

Index Register Y

The Y register is similar to the X register in that it is available for holding counter or offsets memory access and supports the same set of memory load, save and compare operations as wells as increments and decrements. It has no special functions.

Processor Status

As instructions are executed a set of processor flags are set or clear to record the results of the operation. This flags and some additional control flags are held in a special status register. Each flag has a single bit within the register.

Instructions exist to test the values of the various bits, to set or clear some of them and to push or pull the entire set to or from the stack.

- Carry Flag

  The carry flag is set if the last operation caused an overflow from bit 7 of the result or an underflow from bit 0. This condition is set during arithmetic, comparison 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

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

- Interrupt Disable

  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 devices until it is cleared by a 'Clear Interrupt Disable' (CLI) instruction.

- Decimal Mode

  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

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

- Overflow Flag

  The overflow flag is set during arithmetic operations if the result has yielded an invalid 2's complement result (e.g. adding to positive numbers and ending up with a negative result: 64 + 64 => -128). It is determined by looking at the carry between bits 6 and 7 and between bit 7 and the carry flag.

- Negative Flag

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

This page was last updated on 9th August 2003