10 Interrupt features

Subjects ***covered*** in this chapter:

* AFTER
* EVERY
* REMAIN
* The master clock

If you haven't already noticed, a major innovation in the software of the CPC464 is its unique ability to handle interrupts from BASIC - which means that AMSTRAD BASIC is capable of performing a number of simultaneous but separate operations within a program. Such a facility is sometimes referred to as multitasking, and it is implemented by the application of the new commands AFTER and EVERY.

This facility is also clearly demonstrated in the way in which sound may be handled through facilities such as queues and rendezvous.

Every aspect of timing is referred to the master system clock, which is a quartz controlled timing system within the main computer that looks after the timing and synchronisation of events that happen in the computer - things like the scanning of the display and clocking the processor. Where a function in the hardware is related to time, this can be traced back to the quartz master clock.

The software implementation is the ***AFTER*** and ***EVERY*** command, which in keeping with the user-friendly approach of AMSTBAD BASIC do precisely what they say: eg. AFTER the time you have preset in the command line, the program will divert to the designated subroutine and perform the task defined therein.

10.1 AFTER

The CPC464 maintains a real time clock. The AFTER command allows a BASIC program to arrange for subroutines to be called at some time in the future. Four delay timers are available, each of which may have a subroutine associated with it.

AFTER integer expression›[, integer expression] GOSUB ‹line number,

The first integer expression specifies how long before the subroutine should be called. This time is measured in 1/50ths of a second.

The second integer expresssion› specifies which of the four possible delay timers is to be used. The expression must yield a value in the range 0..3. If the expression is omitted, 0 is assumed.

When the time specified has passed, the subroutine is called automatically, just as if a GOS U B had been issued at the current position in the program. When the subroutine finishes, using a normal ***RETURN*** command, the main program continues running where it was interrupted.

The timers have different interrupt priorities. Timer 3 has the highest priority and timer 0 the lowest.

A**F** *T E R* commands may be issued at any time, resetting the subroutine and time associated with the given delay timer. The delay timers are the same as those used in the ***EVERY*** command, so an ***AFTER*** overrides any previous ***EVERY*** for the given timer, and vice versa.

```
10 MODE 1:X=0
20 AFTER 45 GOSUB 100
30 AFTER 100,1 GOSUB 200
40 PRINT "AMSOFT"
50 WHILE X<100
60 LOCATE #1,30,1:PRINT #1,X:X=X+1
70 WEND
80 END
100 PRINT "peripherals"
110 RETURN
200 PRINT "and software"
210 RETURN
```

Note the use of two streams (windows) to allow printing by the 'main program' (Lines 50-80) using independant cursor positioning from that used by the interrupt subroutines.

A**F**TER the time you have preset in the command line, the program diverts to the designated subroutine and performs the task defined therein:

10.2 EVERY

The E V E R Y command allows a BASIC program to arrange for subroutines to be called at regular intervals. Four delay timers are available, each of which may have a subroutine associated with it. Its form is:

E V E R Y ‹**integer** expression1 , ‹**integer** expression] G 0 S U B line number)

The first ‹**integer** expression specifies how long to wait between each call of the subroutine. This time is measured in **1/50ths** of a second.

The second <integer **expresssion**› specifies which of the four possible delay timers is to be used. The expression must yield a value in the range **0..3**. If the expression is omitted, timer 0 is assumed.

When the time specified has passed, the subroutine is called automatically, just as if a G 0 SUB had been issued at the current position in the program. When the subroutine finishes, using a normal ***RETURN*** command, the main program continues running where it was interrupted.

The timers have different interrupt priorities. Timer 3 has the highest priority and timer 0 the lowest. Immediately the timer expires, the count is reset and the count down to the next call of the subroutine begins.

E V E R Y commands may be issued at any time, resetting the subroutine and time associated with the given delay timer. The delay timers are the same as those used in the A F T E *R* command, so an *EVERY* overrides any previous A F T E R for the given timer, and vice versa.

```
10 MODE 1:X=0
20 P100=0:EVERY 10 GOSUB 100
30 P200=0:EVERY 12,1 GOSUB 200
40 PRINT "AMSOFT"
50 WHILE X<200
60 LOCATE #1,30,1:PRINT #1,X:X=X+1
70 WEND
80 LOCATE 1,20:END
100 DI:PEN P100:LOCATE 1,2:PRINT "peripherals":EI
105 IF P100=0 THEN P100=1 ELSE P100=0
110 RETURN
200 PEN P200:LOCATE 1,3:PRINT "and software"
205 IF P200=2 THEN P200=3 ELSE P200=2
210 RETURN
```

NOTE the use of *D I* and E I commands which disable and enable timer and sound interrupts whilst the commands between them are executed. This has the effect of delaying the (higher priority) interrupt of timer 1 from ever occurring during the processing of the interrupt from timer 0 (lines 100-110). Therefore, the P E N or L 0 C A T E settings are not upset before the P R I N T command.

10.3 REMAIN

This function returns the remaining count for one of the four system delay timers. It disables the timer, returning zero if the timer is already disabled. It is used in the form:

R E MA I N (‹integer expression)