There are no interrupts running when PicoDOS starts your application. Any and all interrupts that your application encounters come from features that you’ve explicitly enabled (SCIRxSetBuffered, SCITxSetBuffered, TUOpen, etc) or from code you’ve added. When your ISR gets control from an interrupt, the interrupt level sets the current mask level and only higher level interrupts can preempt your ISR.

> what are the cpu activities that disable the interrupt system?

The longest instructions are DIVS.L, DIV.U, MUL.L which I suspect may be generated with C code integer or floating point multiplication or division. The longest execution time would be on the order of 8us.

These are PicoDOS actions that disable interrupts (4.01r1, Jun05) (4.02r1, Jan 06)

searching on IEVDisableAll, IEVSaveSRThenWriteSR, IEVSaveSRThenDisable

CSSetSysWaits	602 bytes, 300 instr, copy loops, approx 250us	
FlashInit	approx 1ms	
FlashPreWrite		
SCIInit	approx 100us	
SCIConfigure	approx 100us	
SCIRxSetBuffered	approx 50us	
SCITxSetBuffered	approx 50us	
TMGSetSysClock	approx 100us + any change chores	
BIOSAddChore	approx 200us	
BIOSRemoveChore	approx 200us	
BIOSInit	many ms	
PITInit	approx 200us	
PITSet100usPeriod	approx 10us	
PITSet51msPeriod	approx 10us	
RTCInit	approx 1ms	
PPBPingPong	approx 10us	
SCITxFlush	approx 5us	
SCIRxFlush	approx 50us	
SCIRxTxISR	approx 50us	<<<< i.e. all bufferred serial
SCIRxHandshakeUpdateXOFF SCIRxSetBuffered SCIRxFlush SCIRxGetBlock SCIRxGetByteBuffed	approx 25us	called by
TLCSetTime	approx 200us	
TUTxPutByte	approx 10us	
TUTxFlush	approx 10us	
RTCGetTime	CF1:20us	
PWRSuspendTicks	on resume	