Contended memory

Contended memory is a quirk of the ZX Spectrum's hardware design which means that it is on average slower to access those memory areas which are shared with the ULA than it is to access other memory areas. This occurs because the RAM cannot be read by two devices (the ULA and the processor) at once, and the ULA is given higher priority so it can drawn the screen correctly. Therefore, programs which access this "contended memory" (which is from 0x4000 to 0x7fff on 16K or 48K models) or try to read from an I/O port where the result is provided by the ULA (any port with the low bit reset) will be slowed if the ULA is reading the screen. This effect occurs only when the actual screen is being drawn; when the border is being drawn or the TV is in either horizontal or vertical refresh, the ULA does not need to access memory and therefore no delays occur.

Contents

General principles

In order for the ULA to be able to access the memory it needs without problems being caused by the Z80 attempting to access the memory at the same time, the ULA arranges for the Z80 to be temporarily paused if the Z80 attempts to access the appropriate memory or I/O ports; the exact details of which memory is affected and at which times is given in the Details section below. For memory access, this happens on the first tstate (T1) of any instruction fetch, memory read or memory write operation.[1] For I/O operations, this can happen on all tstates; see the Contended I/O article for the specifics. The table below gives the pattern of contention that is applied for each opcode, which is essentially equivalent to when T1 operations happen in each instruction.

Details

16K and 48K

On the 16K and 48K models of ZX Spectrum, the memory from 0x4000 to 0x7fff is contended. If the contended memory is accessed 14335[2] or 14336 tstates after an interrupt (see the timing differences section below for information on the 14335/14336 issue), the Z80 will be delayed for 6 tstates. After 14336 tstates, the delay is 5 tstates. The pattern continues as follows:

Tstates	Delay
14335	6 (until 14341)
14336	5 (until 14341)
14337	4 (until 14341)
14338	3 (until 14341)
14339	2 (until 14341)
14340	1 (until 14341)
14341	No delay
14342	No delay
14343	6 (until 14349)
14344	5 (until 14349)
14345	4 (until 14349)
14346	3 (until 14349)
14347	2 (until 14349)
14348	1 (until 14349)
14349	No delay
14350	No delay

This pattern (6,5,4,3,2,1,0,0) continues until 14463 tstates after interrupt, at which point there is no delay for 96 tstates while the border and horizontal refresh are drawn. The pattern starts again at 14559 tstates and continues for all 192 lines of screen data. After this, there is no delay until the end of the frame as the bottom border and vertical refresh happen, and no delay until 14335 tstates after the start of the next frame as the top border is drawn.

Spectrum 128 and +2

On the Spectrum 128 and Spectrum +2, memory pages 1, 3, 5 and 7 are contended. This means that RAM from 0x4000 to 0x7fff is always contended (as memory page 5 is always mapped in there) and RAM from 0xc000 to 0xffff can be contended if page 1, 3, 5 or 7 is paged in there. The 128 and +2 also have a different timing pattern from the 16K and 48K models due to their different line and frame lengths: the 6,5,4,3,2,1,0,0 pattern starts 14361 tstates after interrupt, and repeats every 228 tstates rather than 224.

Spectrum +2A, +3, +2B, and +3B

The gate array in the +2A, +3, +2B, and +3B differs more significantly in that it applies less contention than the ULAs in the earlier models. Specifically, it applies memory contention only if the $\overline{\text{MREQ}}$ line is active, whereas the 16K/48K ULA applies it under all circumstances. Unlike the 128 or +2 the Amstrad gate array contends pages 4, 5, 6, and 7. In the instruction breakdown table, contention patterns which differ on the gate array models are shown in **bold** in the 'Amstrad gate array' column, with sections of contention specific to the Ferranti ULAs shown in **bold** in the 'ULA' column. T-state counts associated with the ULA contention still apply to the gate array pattern, but the contention itself does not. With ULA contention excluded, the timings in both columns are identical.

The timing pattern also differs significantly:

Tstates	Delay
14361	1 (until 14362)
14362	No delay
14363	7 (until 14370)
14364	6 (until 14370)
14365	5 (until 14370)
14366	4 (until 14370)
14367	3 (until 14370)
14368	2 (until 14370)
14369	1 (until 14370)
14370	No delay
14371	7 (until 14378)
14372	6 (until 14378)
14373	5 (until 14378)
14374	4 (until 14378)
14375	3 (until 14378)
14376	2 (until 14378)

The pattern repeats until 14490 tstates, when the first scanline has been finished, after which no delays are inserted until 14589 tstates when the pattern begins again.

NTSC Spectrum

The NTSC Spectrum has the same 6,5,4,3,2,1,0,0 contention pattern as the 16K/48K models, but starting at tstate 8959 rather than 14335.

Examples

Both these refer to a 48K Spectrum.

Example 1: if PC = 25000, HL = 26000, the instruction at address 25000 is LD (HL),A and we're at tstate 14335:

- Delay for 6 tstates (the contention delay for tstate 14335); now at tstate 14341.
- 4 tstates fetching the opcode; now at tstate 14345.
- Delay for 4 tstates (delay for tstate 14345); now at tstate 14349.
- 3 tstates storing the byte; now at tstate 14352.

The next opcode will then be read at tstate 14352.

Example 2: the same setup as example 1, except with PC=40000 (not contended):

- No delay because PC is not contended.
- 4 tstates fetching the opcode; now at cycle 14339.
- Delay for 2 tstates (for tstate 14339); now at tstate 14341;
- 3 tstates storing the byte; now at tstate 14344.

Timing differences

It has been observed that on ULA based machines, the timings may be one tstate later than normal. All timings given in this document are for "early timing"; for "late timing", simply add one to add T-state counts given. Machines based on the Amstrad gate array do not exhibit this behaviour.

The physical reason for this difference is that as the ULA heats up, it drifts from "early timing" to "late timing" due to increased thermal resistance. A machine that has been left off for some time and just switched on will always exhibit "early timing". Some emulators have a "late timing" option to switch the ULA to a "hot" state.

Instruction breakdown

In this below:

- dd is any of the registers BC,DE,HL,SP
- qq is any of the registers BC,DE,HL,AF
- ss is any of the registers BC,DE,HL
- ii is either of the index registers IX or IY.
- ir is the IR (Interrupt and Refresh) register pair
- cc is any (applicable) condition NZ,Z,NC,C,PO,PE,P,M
- nn is a 16-bit number
- n is an 8-bit number
- b is a number from 0 to 7 (BIT/SET/RES instructions)
- r and r' are any of the registers A,B,C,D,E,H,L
- alo is an arithmetic or logical operation: ADD/ADC/SUB/SBC/AND/XOR/OR and CP
- sro is a shift/rotate operation: RLC/RRC/RL/RR/SLA/SRA/SRL and SLL (undocumented)

Further notes:

- The values for the registers listed in the table below are relative to the starting value of the register when the instruction is about to be executed.
- For conditional instructions, entries in [square brackets] are applied only if the condition is met. If the instruction is not conditional (e.g. CALL nn) the entries in [] always apply.
- The replacement of HL by either IX or IY does not affect the timings, except for the addition of an initial pc:4 for the DD or FD prefix; similarly, a DD or FD prefix on an instruction which does not involve HL just adds an initial pc:4.
- The undocumented variants of the doubly shifted DDCB and FDCB opcodes have the same timings as the documented versions.
- In some read-modify-write operations (like INC (HL)), the write operation is always the last one. That may be important to know the exact point in which video is updated, for example. In such instructions that point is annotated for clarity as "(write)" after the address.
- Access to I/O ports is treated differently to access to memory; full details are given Contended I/O. The delays specified there should be applied when an I/O port is accessed; this is designated by "I/O" in the table below.

In the table below, contention patterns which differ on gate array models are shown in **bold** in the 'Amstrad gate array' column, with sections of contention specific to the Ferranti ULAs shown in **bold** in the 'ULA' column. T-state counts associated with the ULA contention still apply to the gate array pattern, but the contention itself does not. With ULA contention excluded, the timings in both columns are identical.

Opcode	ULA	Amstrad gate array
NOP	pc:4	pc:4
LD r,r'		
alo A,r		
INC/DEC r		
EXX		
EX AF,AF'		
EX DE,HL		
DAA		
CPL		
CCF		
SCF		
DI		
EI		
RLA		
RRA		
RLCA		
RRCA		
JP (HL)		
NOPD	pc:4,pc+1:4	pc:4,pc+1:4
sro r		
BIT b,r		
SET b,r		
RES b,r		
NEG		
IM 0/1/2		
LD A,I	pc:4,pc+1:4,**ir**:1	**pc:4,pc+1:5**
LD A,R		
LD I,A		
LD R,A		
INC/DEC dd	pc:4,**ir**:1 ×2	**pc:6**
LD SP,HL		
ADD HL,dd	pc:4,**ir**:1 ×7	**pc:11**
ADC HL,dd	pc:4,pc+1:4,**ir**:1 ×7	**pc:4,pc+1:11**
SBC HL,dd		
LD r,n	pc:4,pc+1:3	pc:4,pc+1:3
alo A,n		
LD r,(ss)	pc:4,ss:3	pc:4,ss:3
LD (ss),r		
alo A,(HL)	pc:4,hl:3	pc:4,hl:3
LD r,(ii+n)	pc:4,pc+1:4,pc+2:3,**pc+2**:1 ×5,ii+n:3	**pc:4,pc+1:4,pc+2:8,ii+n:3**
LD (ii+n),r		
alo A,(ii+n)		
BIT b,(HL)	pc:4,pc+1:4,hl:3,**hl**:1	**pc:4,pc+1:4,hl:4**

BIT b,(ii+n)	pc:4,pc+1:4,pc+2:3,pc+3:3,**pc+3**:1 ×2,ii+n:3,**ii+n**:1	**pc:4,pc+1:4,pc+2:3,pc+3:5,ii+n:4**
LD dd,nn	pc:4,pc+1:3,pc+2:3	pc:4,pc+1:3,pc+2:3
JP nn		
JP cc,nn		
LD (HL),n	pc:4,pc+1:3,hl:3	pc:4,pc+1:3,hl:3
LD (ii+n),n	pc:4,pc+1:4,pc+2:3,pc+3:3,**pc+3**:1 ×2,ii+n:3	**pc:4,pc+1:4,pc+2:3,pc+3:5,ii+n:3**
LD A,(nn)	pc:4,pc+1:3,pc+2:3,nn:3	pc:4,pc+1:3,pc+2:3,nn:3
LD (nn),A		
LD HL,(nn)	pc:4,pc+1:3,pc+2:3,nn:3,nn+1:3[3]	pc:4,pc+1:3,pc+2:3,nn:3,nn+1:3[3]
LD (nn),HL		
LD dd,(nn)	pc:4,pc+1:4,pc+2:3,pc+3:3,nn:3,nn+1:3[4]	pc:4,pc+1:4,pc+2:3,pc+3:3,nn:3,nn+1:3[4]
LD (nn),dd		
INC/DEC (HL)	pc:4,hl:3,**hl**:1,hl(write):3	**pc:4,hl:4,hl(write):3**
SET b,(HL)	pc:4,pc+1:4,hl:3,**hl**:1,hl(write):3	**pc:4,pc+1:4,hl:4,hl(write):3**
RES b,(HL)		
sro (HL)		
INC/DEC (ii+n)	pc:4,pc+1:4,pc+2:3,**pc+2**:1 ×5,ii+n:3,**ii+n**:1,ii+n(write):3	**pc:4,pc+1:4,pc+2:8,ii+n:4,ii+n(write):3**
SET b,(ii+n)	pc:4,pc+1:4,pc+2:3,pc+3:3,**pc+3**:1 x 2,ii+n:3,**ii+n**:1,ii+n(write):3	**pc:4,pc+1:4,pc+2:3,pc+3:5,ii+n:4,ii+n(write):3**
RES b,(ii+n)		
sro (ii+n)		
POP dd	pc:4,sp:3,sp+1:3	pc:4,sp:3,sp+1:3
RET		
RETI	pc:4,pc+1:4,sp:3,sp+1:3	pc:4,pc+1:4,sp:3,sp+1:3
RETN		
RET cc	pc:4,**ir**:1,[sp:3,sp+1:3]	**pc:5,[sp:3,sp+1:3]**
PUSH dd	pc:4,**ir**:1,sp-1:3,sp-2:3	pc:5,sp-1:3,sp-2:3
RST n		
CALL nn	pc:4,pc+1:3,pc+2:3,[**pc+2**:1,sp-1:3,sp-2:3]	**pc:4,pc+1:3,pc+2:3,[1,sp-1:3,sp-2:3]**
CALL cc,nn		
JR n	pc:4,pc+1:3,[**pc+1**:1 ×5]	**pc:4,pc+1:3,[5]**
JR cc,n		
DJNZ n	pc:4,**ir**:1,pc+1:3,[**pc+1**:1 ×5]	**pc:5,pc+1:3,[5]**
RLD	pc:4,pc+1:4,hl:3,**hl**:1 ×4,hl(write):3	**pc:4,pc+1:4,hl:7,hl(write):3**
RRD		
IN A,(n)	pc:4,pc+1:3,I/O	pc:4,pc+1:3,I/O
OUT (n),A		
IN r,(C)	pc:4,pc+1:4,I/O	pc:4,pc+1:4,I/O
OUT (C),r		
EX (SP),HL	pc:4,sp:3,sp+1:3,**sp+1**:1,sp+1(write):3,sp(write):3,**sp(write)**:1 ×2	**pc:4,sp:3,sp+1:4,sp+1(write):3,sp(write):5**
LDI/LDIR	pc:4,pc+1:4,hl:3,de:3,**de**:1 ×2,[**de**:1 ×5]	**pc:4,pc+1:4,hl:3,de:5,[5]**
LDD/LDDR		
CPI/CPIR	pc:4,pc+1:4,hl:3,**hl**:1 ×5,[**hl**:1 ×5]	**pc:4,pc+1:4,hl:8,[5]**
CPD/CPDR		

INI/INIR	pc:4,pc+1:4,**ir**:1,I/O,hl:3,[**hl**:1 ×5]	**pc:4,pc+1:5,I/O,hl:3,[5]**
IND/INDR		
OUTI/OTIR	pc:4,pc+1:4,**ir**:1,hl:3,I/O,[**bc**:1 ×5]	**pc:4,pc+1:5,hl:3,I/O,[5]**
OUTD/OTDR		

Notes

1. http://www.zxdesign.info/memContRevision.shtml
2. In this document, we label the first tstate which *begins* with $\overline{\text{INT}}$ low as tstate 0; some other resources label this tstate as tstate 1, which means that all tstate counts will be one greater. Note that this is purely a notational difference, and is *not* the same as the effect observed in the timing differences section, which is a actual difference in behaviour between different machines; when using the notation which labels the first $\overline{\text{INT}}$ low tstate as tstate 1, the first contended memory cycle is at either 14336 or 14337 tstates.
3. Applies to the unprefixed version of these opcodes (22 and 2A)
4. Applies to the prefixed version of these opcodes (ED43, ED4B, ED53, ED5B, ED63, ED6B, ED73 and ED7B)

External links

- Wikipedia article section on bus arbitration from a more general point of view

Article license information

Retrieved from ‘https://sinclair.wiki.zxnet.co.uk/w/index.php?title=Contended_memory&oldid=1840’

This page was last modified on 1 January 2018, at 22:06.