1. General Descriptions

KL5C8400 is a fast 8 bit CPU, which is compatible with the Z80 microprocessor at binary level. With an internal 16 bit RISC-like architecture, the performance of the CPU is equivalent to the Z80 microprocessor at 44 MHz. With this high performance, the CPU is faster than typical 16 bit microcontrollers and CPUs. The advanced CMOS technology provides high performance at low power consumption.

The CPU has two operation modes controlled by CNFG input (mode pin). The CPU provides 1.2 times higher performance at the same clock rate in Z80 mode. In KC80 mode, the CPU provides 1.3 times higher performance at the same clock rate. There is also an advantage in memory access time in KC80 mode, because the CPU doesn't have M1 cycle.

In Z80 mode, the CPU operates in Z80 compatible bus cycles. Then, Z80 peripherals can be used with the CPU in mode 2 interrupt.

In KC80 mode, the CPU fetches its opecode using the same bus cycle as memory read. Then, slower memory can be used with the CPU in KC80 mode. There is also a performance advantage because it fetches its opcode in shorter opcode fetch cycle. In KC80 mode, only mode 1 interrupt can be used.

Features

- Instruction set: Fully compatible with Z80 MPU at binary level
- High speed operation: 33 MHz (equivalent performance of Z80 at 44 MHz)
- Minimum execution time: 90 ns (3 clocks)
- Low power consumption:
- Address space: 64 Kbyte
- Operational clock frequency: 0 ~ 33 MHz
- Interrupt: Maskable interrupt 1, Non-maskable interrupt 1
- Package: 44 pin QFP package
- Two operation modes: Z80 mode (CNFG = "H"), KC80 mode (CNFG = "L")

Instruction execution time comparison

instructions	KL5C8400 (Z80 mode)	KL5C8400 (KC80 mode)	Z80
LD B, C	4 clocks	3 clocks	4 clocks
ADD HL, BC	4 clocks	3 clocks	11 clocks
DEC DE	4 clocks	3 clocks	6 clocks
POP AF	10 clocks	9 clocks	10 clocks
JR Z, +20h	10 clocks	9 clocks	12 clocks

2. Block Diagram

Figure 1a shows the block diagram of KL5C8400.

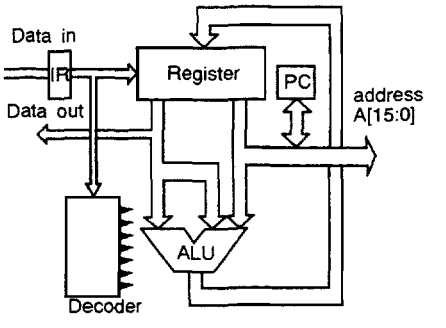
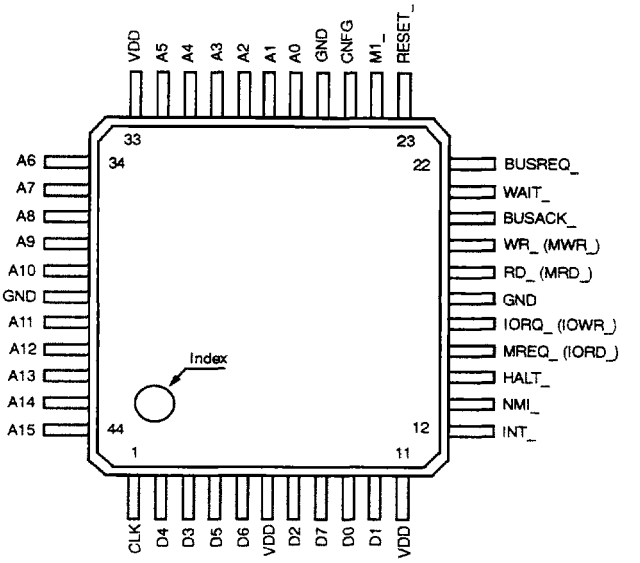


Fig.1a Block Diagram

3. Pin Description



( ) indicates signal names in KC80 Mode.

Fig.1b Pin assignment

pin name	I/O	description
GND	Power supply	Connect 0V.
VDD	Power supply	Connect 5V.
CNFG	I	Input for setting mode. When CNFG = "H", the setting is Z80 mode. When CNFG = "L", the setting is KC80 mode. Fix this pin to "H" or "L" on the target board.
RD_	O	Read (active "L", 3-state output, Z80 mode signal) This signal goes active while reading data from memory or an I/O device.
WR_	O	Write (active "L", 3-state output, Z80 mode signal) This signal goes active while writing data to memory or an I/O device.

pin name	I/O	description
IORQ_	O	I/O request (active "L", 3-state output, Z80 mode signal) This signal goes active while accessing I/O.
MREQ_	O	Memory request (active "L", 3-state output, Z80 mode signal) This signal goes active while accessing memory.
CLK	I	System clock
WAIT_	I	Wait (active "L") Wait states are inserted when this signal is "L".
A[15:0]	O	Address bus (3-state output)
D[7:0]	I/O	Data bus
RESET_	I	Reset (active "L") The CPU is initialized when this signal is "L".
NMI_	I	Non-Maskable interrupt (falling edge trigger) This input accepts the non-maskable interrupt. The priority of this input is higher than INT_ but lower than BREQ_. As soon as KL5C8400 completes a current instruction, it jumps to 0066H regardless of the status of interrupt enable flip-flop (IFF).
HALT_	O	Halt (active "L") This signal indicates that the CPU executes HALT instruction and is in the HALT status. The CPU resumes from the HALT status by receiving NMI, INT or reset. During HALT status, the CPU continuously executes NOP instructions.
INT_	I	Maskable interrupt (active "L") This input accepts the interrupt from I/O devices. If the interrupt enable flip-flop (IFF) of the CPU is set, and BUSREQ_ is inactive, the CPU completes the execution of the current instrution, and starts interrupt service.

pin name	I/O	description
M1_	O	Machine Cycle One (active "L") This signal indicates that the current machine cycle is the opecode fetch cycle of a executing instruction.
BUSREQ_	I	Bus request (active "L") The adress bus and the data bus go to high-impedance state after completion of the current instruction execution when this signal goes active. This signal has higher priority than NMI_ and INT_.
BUSACK_	O	Bus acknowledge (active "L") This signal indicates that the CPU released the bus for an external bus master.
MRD_	O	Memory read (active "L", 3-state output, KC80 mode) This signal goes active while reading data from a memory.
MWR_	O	Memory write (active "L", 3-state output, KC80 mode) This signal goes active while writing data to a memory.
IORD_	O	I/O read (active "L", 3-state output, KC80 mode) This signal goes active while reading data from an I/O device.
IOWR_	O	I/O write (active "L", 3-state output, KC80 mode) This signal goes active while writing data to an I/O device.

INT_, NMI_, WAIT_, BUSREQ_, RESET_ are provided with internal pull-up resistors.
About those characteristics (the pull-up current value), see chapter 8.

6. Functional description and timing

In this section, functional description and timing of Z80 mode are described. KC80 mode is described only the difference between Z80 mode in section 7.

6.1 Basic operation (Instruction cycle)

Basic operation of KL5C8400 can be divided into the following 5 cycles.

1)Opcode fetch cycle - Fetches opcode of an instruction from memory.
2)Memory read cycle - reads data from memory
3)Memory write cycle - writes data to memory
4)I/O read cycle - reads data from I/O device
5)I/O write cycle - writes data to I/O device

Follows are the description of the each machine cycles.

Opcode fetch cycle

As shown Figure 4, the CPU fetches one byte of an instruction in 4 clock cycles unless there is a wait. During this cycle, MREQ_, RD_ and M1_ go active. This cycle appears at the end of the instruction execution sequence. the CPU outputs the refresh address to the address bus in the latter half of the opcode fetch cycle. Note that the CPU doesn't have the refresh signal. When WAIT_ goes active ("L") at the rising edge of T_2, the wait cycle T_W is entered.

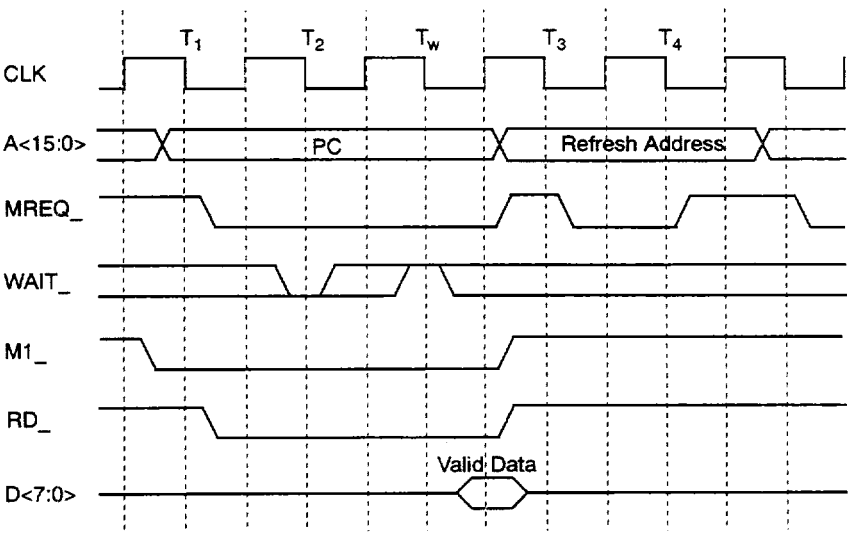


Fig.4 Opcode fetch cycle (Z80 mode)

Memory read/write cycle

The difference between the opcode fetch cycle is whether M1_ goes active or not, and is one bus cycle in three -clock cycles. When WAIT_ goes active ("L") at the rising edge of T_2, the wait cycle T_W is inserted.

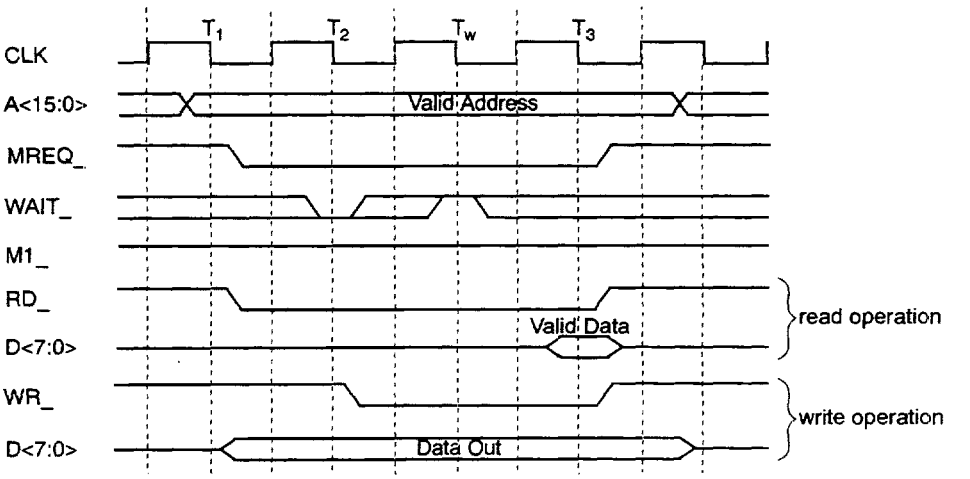


Fig.5 Memory read/write cycle (Z80 mode)

I/O read/write cycle

This cycle is minimum of four-clock cycles. When WAIT_ goes active ("L") at the rising edge of T_3, the wait cycle T_W is inserted.

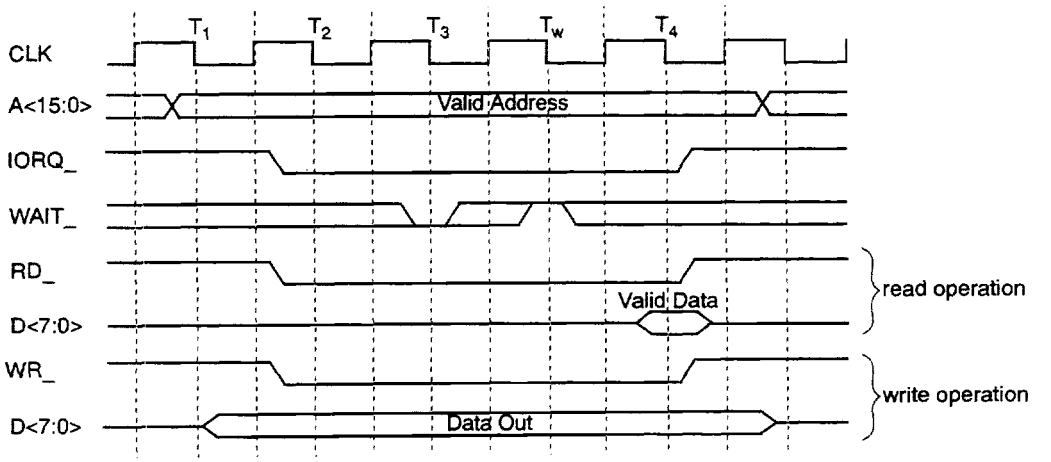


Fig.6 I/O read/write cycle (Z80 mode)

Instruction Prefetch cycle

The CPU has the prefetch cycle at the end of instruction execution cycle all the time. Figure 8 shows that the CPU executes the instruction sequence indicated in Figure 7 as an example.

 (referred as "arrow" hereinafter) in the Figure 8 denotes the prefetch cycle. The arrow ① part prefetches the instruction at n+3 (ADD A,D), the arrow ② part is for prefetching instruction at n+4 (opcode of JP instruction, C3H), the arrow ③ part is for prefetching 77H at 1000H (LD [HL], A instruction). As shown, these cycles appear at the end of execution of current instruction.

address	mnemonic	code
n	LD A, [1234H]	3A 34 12
n+3	ADD A, D	82
n+4	JP 1000H	C3 00 10
.	.	
.	.	
1000H	LD [HL], A	77
.	.	
1234H		5A

Fig.7 Source code for Fig. 8

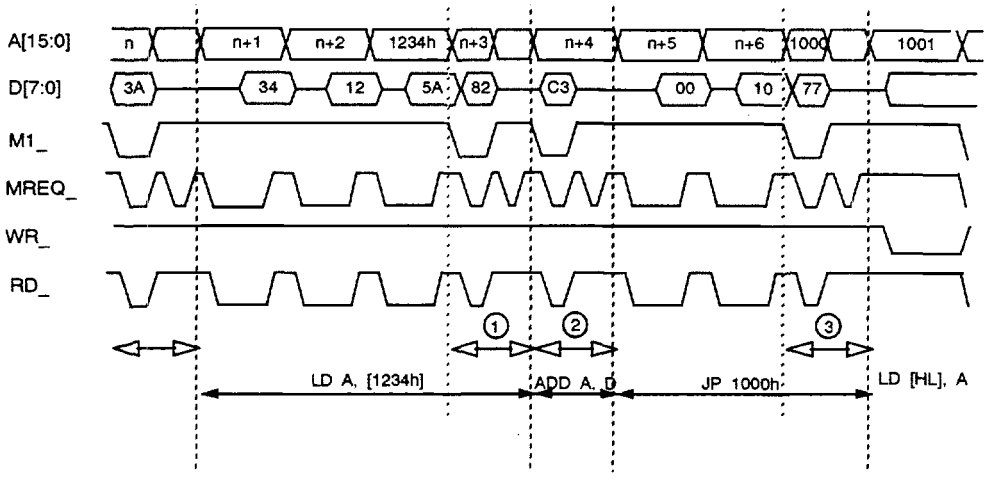
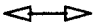


Fig.8 Prefetch cycle

( denotes the prefetch cycle. Assumes memory location 1000H has 77H, memory location 1234H has 5AH, and D register has 81H.)

Special case for prefetch.

(Discard of the prefetched instruction)

For example, under normal condition, the prefetched instruction, at address 1000H, data 77H at ③ in Figure 8 on execution of the instruction, JP 1000H is held and execute correctly. However, on interrupt the prefetched instruction will be discarded and refetch on return from an interrupt service routine. On interrupt recognition, PC is decremented by one before saved onto stack. On RETI or RETN instruction the PC will be loaded with the address of the discarded instruction. Figure 13 has the timing diagram for the case of receiving interrupt in mode 1 when executing LDIR instruction. In the figure, the data "EDH" prefetched at ⑥, is discarded and then go to interrupt service. On return, the CPU resume from the prefetch of this "EDH". For the bus request cycle, the data "EDH" prefetched at ⑥ prior to the bus request will be kept, and on bus release the CPU starts to prefetch from the next address to the "EDH".

6.2 Bus release (Bus request/acknowledge cycle)

Under normal operation, the CPU holds the control of address bus, control bus (RD_, WR_, MREQ_ and IORQ_). However, if there is an external bus request (BUSREQ_= "L"), address bus goes to high-impedance state, all the interrupts are disabled, BUSACK_ turns to "L", and the CPU releases the bus control to the external device. By using this feature, data transfer without CPU intervention (DMA transfer) can be done. Figure 9 shows basic timing for the bus request cycle. This cycle continues while BUSREQ_ stays "L". Note that bus request is accepted at the end of the bus cycle.

6.3 Interrupt and timing

The CPU can handle the following two kinds of interrupts.

1)maskable interrupt on INT_

2)Non-maskable interrupt on NMI_

2) has the higher priority over 1). If both request made simultaneously, 2) will be accepted over 1).

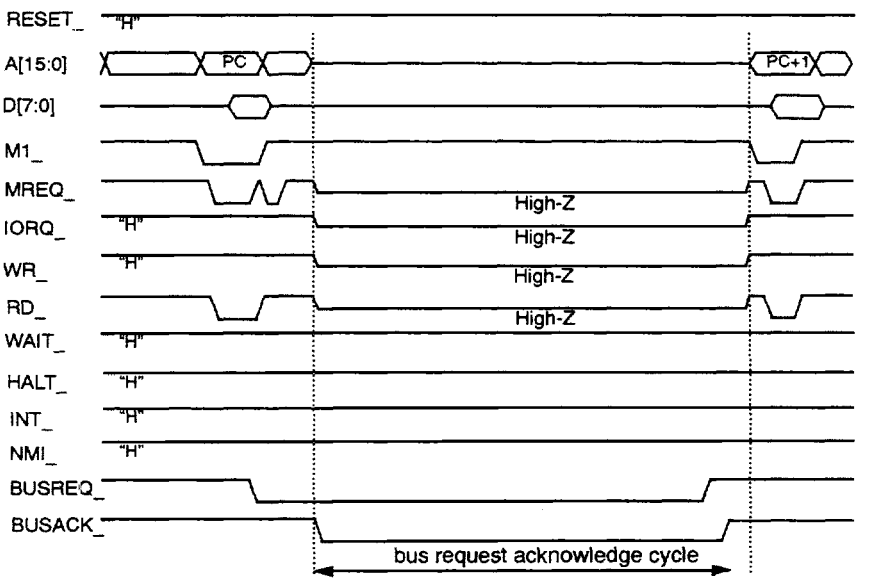


Fig.9 Bus request timing

Maskable interrupt

The EI instruction enables interrupts, and the DI instruction disables interrupts. The control of interrupt is implemented using two flip-flops (IFF1 and IFF2). Figure 10 has the state table for these flip-flops.

The maskable interrupt will be accepted if all of the following conditions are met:

1)Both IFF1 and IFF2 are set. (The EI instruction and RETN instructions change the status of these flags after the execution of the following instruction. So if there are EI instruction followed by DI instruction, the interrupt request will not be accepted.)

2)BREQ_ is inactive ("H").

3)No NMI_ falling edge has been detected.

As shows in the Figure 11, the CPU reads the interrupt vector when M1_ and IORQ_ go active ("L"). When WAIT_ goes active ("L") at the rising edge of T_4, the wait cycle T_W is entered.

events	IFF1	IFF2	
reset	0	0	
DI instruction	0	0	
EI instruction	1	1	
INT accepted	0	0	
NMI accepted	0	-	
RETN instruction accepted	IFF2	-	IFF2 is copied into IFF1
LD A, I instruction	-	-	IFF2 is copied into P/V
LD A, R instruction	-	-	IFF2 is copied into P/V

Fig.10 State diagram for IFF1 and IFF2

("-" denotes remain unchanged.)

Maskable interrupt modes

The CPU has the following three different interrupt handling modes, and interrupt sequence is different by each modes. The following descriptions are the operations in each mode.

1)Mode 0

This is the default mode and set to this on reset automatically. Also, executing "IM 0" instruction sets the CPU to this mode. In this mode, the CPU executes the instruction read during interrupt acknowledge cycle. Usually the instruction to be used in this mode is RST instructions (RST 00H ~ RST 38H). Figure 12 show each timing for RST instructions. For the all interrupt modes, if INT_ goes active during execution of a mode change instruction (IM 0 ~ 2), the new mode become effective right after these instructions.

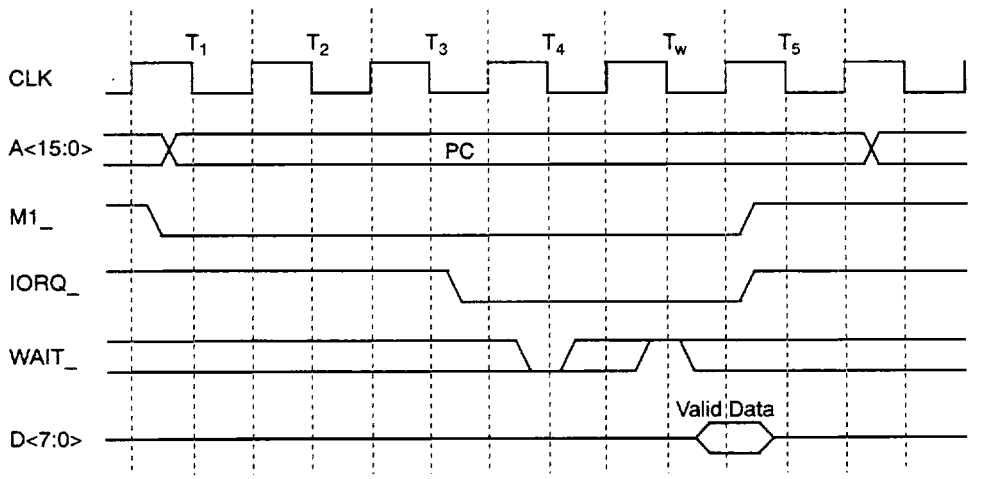


Fig.11 Z80 special M1 cycle (Z80 mode)

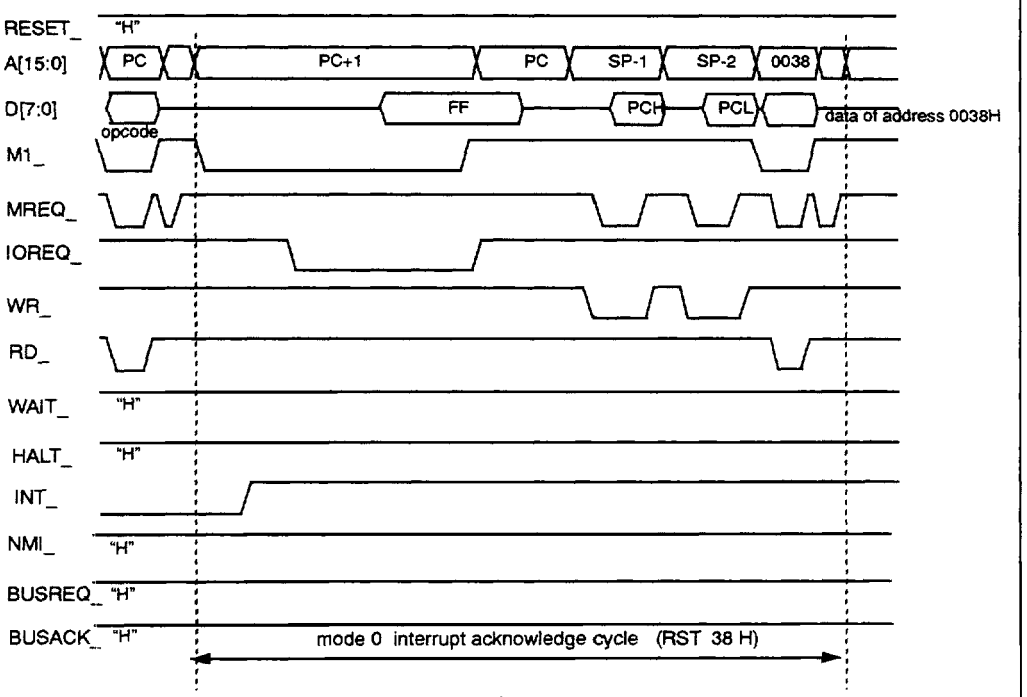


Fig.12 Mode 0 timing

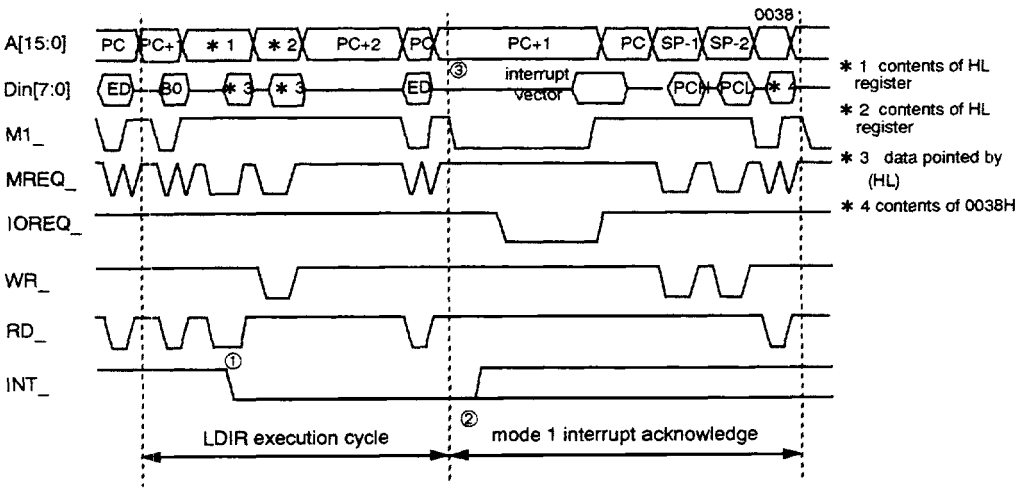


Fig.13 Interrupt acknowledge cycle (Mode 1 timing)

(The figure shows the case when the interrupt occurred at ①,accepting it at the instruction boundary ②. This case restarts from the fetch of ED at ③ on the return from an interrupt.)

2)Mode 1

The "IM 1 instruction" sets the CPU in this mode. In this mode, the CPU saves the contents the PC (Program Counter) onto the stack, ignores the data read during the interrupt acknowledge cycle, and executes "RST 38H" instruction internally. Figure 13 has the timing for Mode 1 interrupt.

3)Mode 2

The "IM 2 instruction sets" the CPU to this mode. In this mode, as shown in the Figure 14, the 8-bit wide vector with 0 in LSB read during the interrupt acknowledge cycle as a lower byte, and the content of the I register as a upper byte form up a pointer which points an entry in a table of address for the interrupt service routines. After forming up the 16-bit address, the CPU fetches the table location and gets the start address, saves the return address onto the stack, then jumps to the service routine. Figure 15 has the timing diagram for Mode 2 interrupt acknowledge cycle.

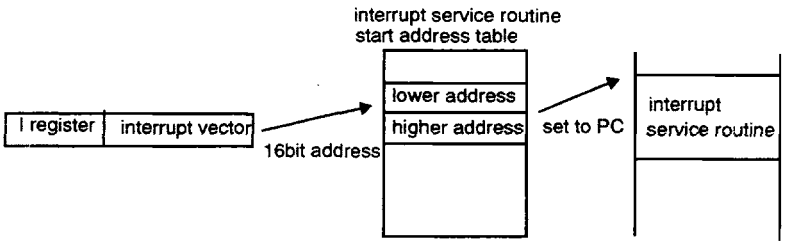


Fig.14 Mode 2 interrupt

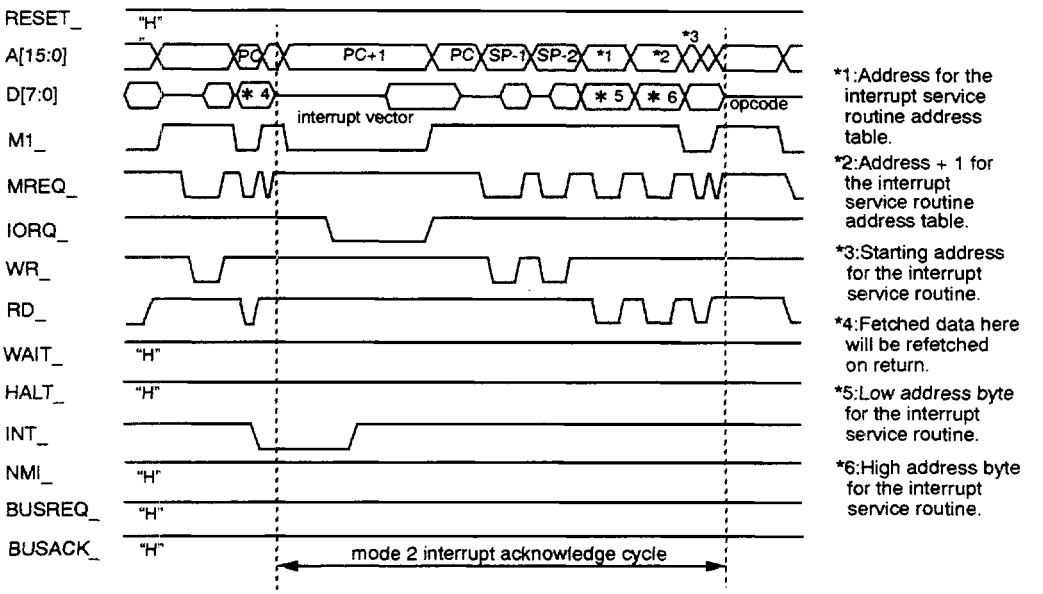


Fig.15 Mode 2 timing

Timing on interrupt acceptance

The interrupt is accepted at the end of instruction.In case of block move instruction, the interrupt accepted at the end of repeat movement. The Figure 13 show the interrupt acknowledge timing. In this figure, an interrupt is accepted during block move instruction. ④ shows that when an interrupt request occurs, the interrupt will not be accepted until the prefetch cycle of the last instruction, ⑤. In this case, the data "EDH" fetched at ⑥ will not be kept by the CPU, and it will be refetched on the return from an interrupt.

Non-maskable interrupt

Non-maskable interrupt is an interrupt which can not be masked by software. The falling edge of the signal NMI_ is latched internally, and checked on the clock falling edge of the last cycle of every instruction. Non-maskable interrupt takes place if BREQ_ is inactive.

When a non-maskable interrupt is accepted, the CPU saves the contents of the PC onto the stack, and jumps to 0066H. Return from non-maskable service is done by "RETN" instruction. Figure 16 shows the timing for NMI_ acknowledge cycle.

Enable/disable of interrupts

Enable/disable of an interrupt is controlled by two flip-flops - IFF1 and IFF2. Refer to Figure 10 on the state diagram for these flip-flops. As shown, maskable interrupt will be accepted when both IFF1 and IFF2 set to 1. After reset and execution of DI instruction maskable interrupt is disabled, because both of these flip-flops are reset. An interrupt can be accepted while both IFF1 and IFF2 are set to 1 by execution of an EI instruction. The reason there are two flip-flops is to memorize the status of EI/DI condition with IFF2 on acceptance a Non-maskable interrupt. For example, if both IFF1 and IFF2 are '1' (maskable interrupt is enabled) on acceptance a non-maskable interrupt, in a non-maskable interrupt acknowledge cycle IFF1 is cleared to 0 and maskable interrupts are disabled. At this stage, IFF2 still holds "1" and the content of IFF2 is copied back into IFF1 on RETN instruction so that maskable interrupts could be enabled automatically again.

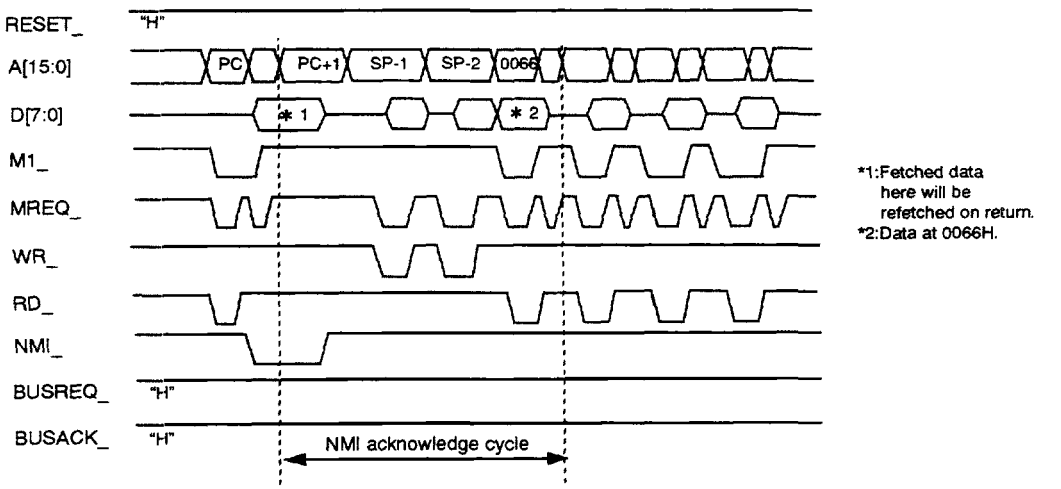


Fig.16 Non-maskable interrupt timing

6.4 Halt enter and exit

By executing the HALT instruction, the CPU enters into the HALT cycle. In this cycle, the CPU continues to execute NOP instruction internally. The CPU exits from this cycle by either reset, or interrupts (Non-maskable interrupt or interrupt with IFF set). Figure 17 shows timing for the case of acceptance an interrupt in mode 2 during the HALT cycle. Then the CPU exits from the HALT cycle. The HALT cycle made up with two separate bus cycles; the first cycle is idle cycle and the second cycle is opcode fetch cycle. The address lines holds the next address to the HALT instruction during this cycle. The instruction fetched during the second opcode fetch cycle will not be read by the CPU. The INT_ input is sampled on the falling CLK edge of the second cycle, and forces HALT_ back to "H".

6.5 Reset timing

By keeping RESET_ input "L" for the minimum three-clock cycles, the CPU is reset. During reset cycle, the address bus goes to high-impedance state. When RESET_ goes inactive ("H"), opcode fetch cycle will be initiated, and starts from 0000H. The interrupt mode is set to mode 0. IFF1, IFF2 flags and I, R registers are reset.

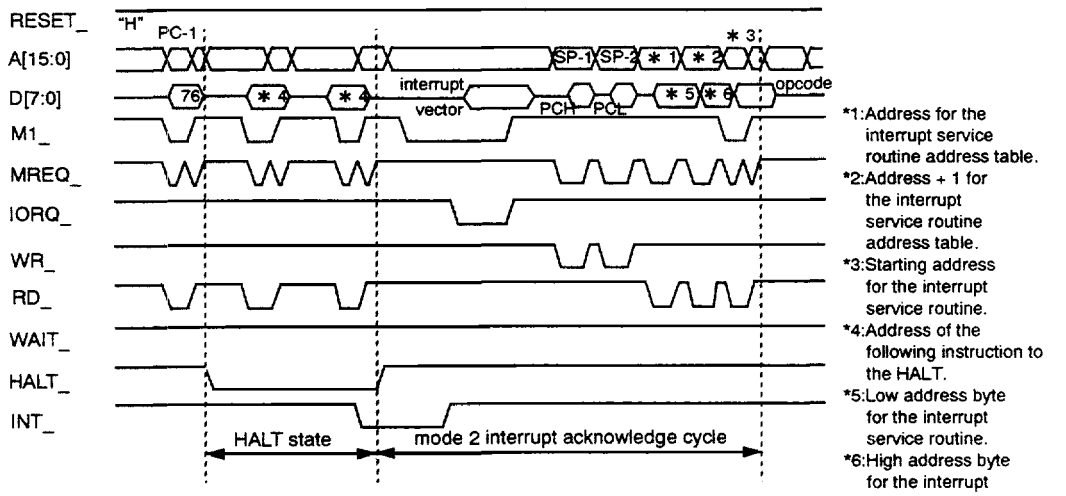


Fig.17 HALT Exit (Mode 2)

7. KC80 Mode

KL5C8400 operates in KC80 mode when the CNFG_ input is "L". In KC80 mode, the CPU fetches its opecode using the same bus cycle as the memory read. Then, slower memories can be used with the CPU in KC80 mode. There is also a performance advantage because it fetches its opcodes in shorter opcode fetch cycles. In KC80 mode, only mode 1 interrupt can be used.

In KC80 mode, the CPU outputs MRD_,MWR_, IORD_ and IOWR_ instead of MREQ_, IORQ_, RD_ and WR_ in Z80 mode. Therefore each cycle of the memory read/write cycle and the I/O read/write cycle changes as shown in Figure 18 and 19.

Figure 18 shows the memory read/write cycle. The memory read/write cycle and the opcode fetch cycle are three-clock bus cycles. When WAIT_ goes active ("L") at the rising edge between T_2 and T_3, the wait cycle T_W is inserted.

In KC80 mode, the opcode fetch cycle is the same as the memory read cycle except M1_ signal. Note that M1_ is inactive ("H") in the opcode fetch cycle but goes active ("L") in the memory read/write cycle.

Figure 19 shows the I/O read/write cycle. The I/O read/write cycle is four-clock bus cycles. When WAIT_ goes active ("L") at the rising edge between T_3 and T_4, the wait cycle T_W is inserted. Note that M1_ goes active ("L") also in the I/O read/write cycle.

Interrupt acknowledge cycle in KC80 mode

In KC80 mode, only mode 1 interrupt can be used as maskable interrupt. Figure 19 shows maskable interrupt (mode 1) timing. When WAIT_ goes active ("L") at the rising edge between T_4 and T_5, the wait cycle T_W is inserted as well as in Z80 mode.

Non-maskable interrupt (NMI) can be naturally used in KC80 mode.

Other timings

The operations in KC80 mode is the same as ones in Z80 mode except the basic bus cycles (memory read/write cycle, I/O read/write cycle, interrupt acknowledge cycle) shown Figure 18, 19 and 20.

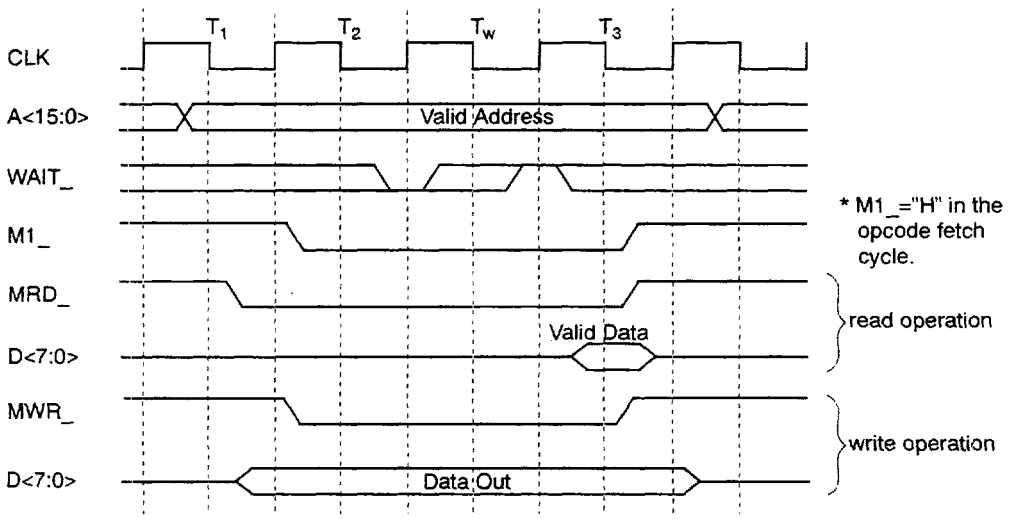


Fig.18 Memory read/write cycle (KC80 mode)

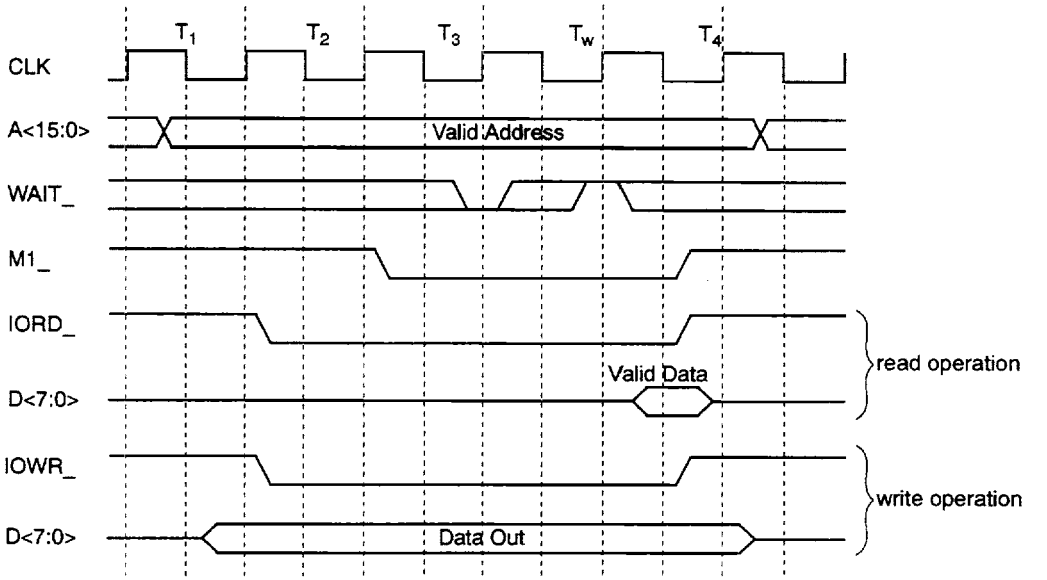


Fig.19 I/O read/write cycle (KC80 mode)

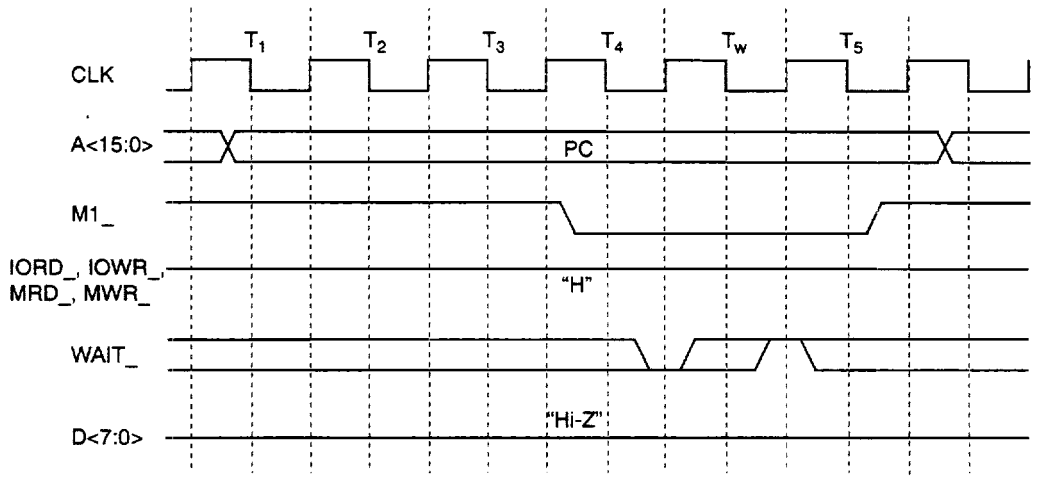


Fig. 20 Mode 1 interrupt accepting timing (KC80 mode)