APPENDIX C Q & A

C.1 Internal CPU Function

Q.1.1

In memory space

(1) If a word access is made to the offset address FFFFH in a segment, what does the real address become?

(2) When the V25/V35 family is reset, the internal data area is located at FFE00H to FFFFFH. Does it overlap the special function register area and bootstrap address of the internal data area?

A.1.1

(1) If a word access is made across the segment boundary (offset address = address next to FFFFH) in the V25/V35 family, one byte in the segment (offset address = FFFFH) and one byte outside the segment (starting address of the next segment) are accessed. On the V20, V30, V40, and V50, address 0000H in the same segment is accessed, unlike in the V25/V35 family.

(2) The addresses overlap each other, but program fetch is not executed for the internal data area. When memory is accessed to fetch a program, bus control signals, such as $\overline{MREQ}$ R/$\overline{W}$, and $\overline{MSTB}$ are always output to the external data area, thus the external memory is accessed. However, the internal ROM version devices (μPD70P322) do not output control signals.

Only a data access is made to the special function register area.

Q.1.2

What is the concept of the memory access time when the V25 family is interfaced with memory?

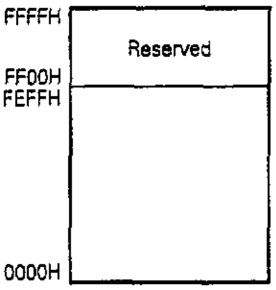
A.1.2

To interface the V25 family with memory, relate V25 family specification t_{DADR} to the address access time of the memory and t_{DMRD} to the $\overline{OE}$ access time of the memory.

Q.1.3

In I/O space

(1) Addresses FF00H to FFFFH are a reserved area. When is this area used?



(2) If an I/O space address is not decoded with respect to the high-order eight bits (A8 to A15), what problem arises for its use?

A.1.3

(1) The reserved area is an area which can be used for future product expansion, and is not used at present.

(2) No problems arise for its use.

Q.1.4

When using memory-mapped I/O

(1) When the CPU internal memory is accessed, is the IDB register rewritten by the CPU rather than by the user?

(2) When using a register bank for hardware interrupts, is a write made directly into the register bank area?

A.1.4

(1) The user needs to set the IDB register. A special function register access is programmed as a normal memory access is programmed.

(2) The register bank is relocated to memory according to the IDB register. Write directly into the register bank area.

C.2 Interrupt Function

Q.2.1

Can multiple interrupts be made within the same group?

A.2.1

Multiple interrupts cannot be made within the same group. If interrupt flags are set within the same group, the interrupt having the highest priority level is acknowledged, and other interrupts within the same group are not acknowledged until a new FINT instruction is executed.

Therefore, for example, even if an INTP0 interrupt occurs while an external interrupt INTP1 is being executed, multiple interrupt servicing of INTP0 and INTP1 cannot be performed.

Q.2.2

For priority levels of external interrupts

(1) If an external interrupt request of INTP0 or INTP1 input is made while an INT interrupt is being executed, is the external interrupt acknowledged?

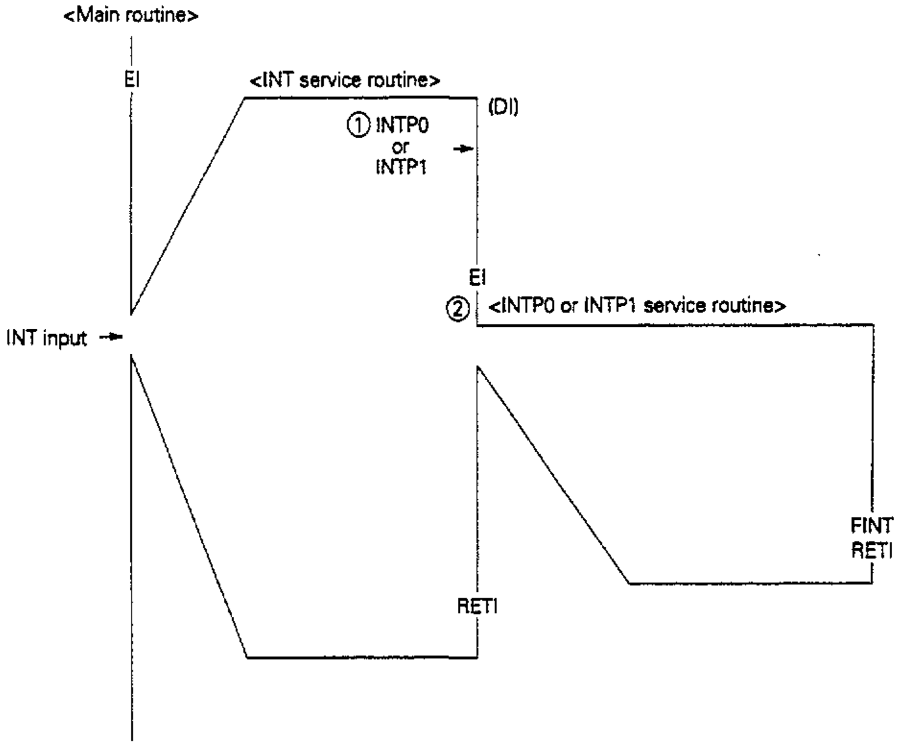
(2) If interrupt requests are input to INTP0 and INTP1 at the same time that an INT interrupt is being executed, how is the operation performed?

(3) If INT is input while an INTP0 or INTP1 interrupt is being executed, is the INT interrupt acknowledged?

A.2.2

(1) If INTP0 or INTP1 is input during execution of the INT interrupt service routine, it is acknowledged if the interrupt flag is set to the EI state.

An example of the INT and INTP0 or INTP1 service routine is given below.

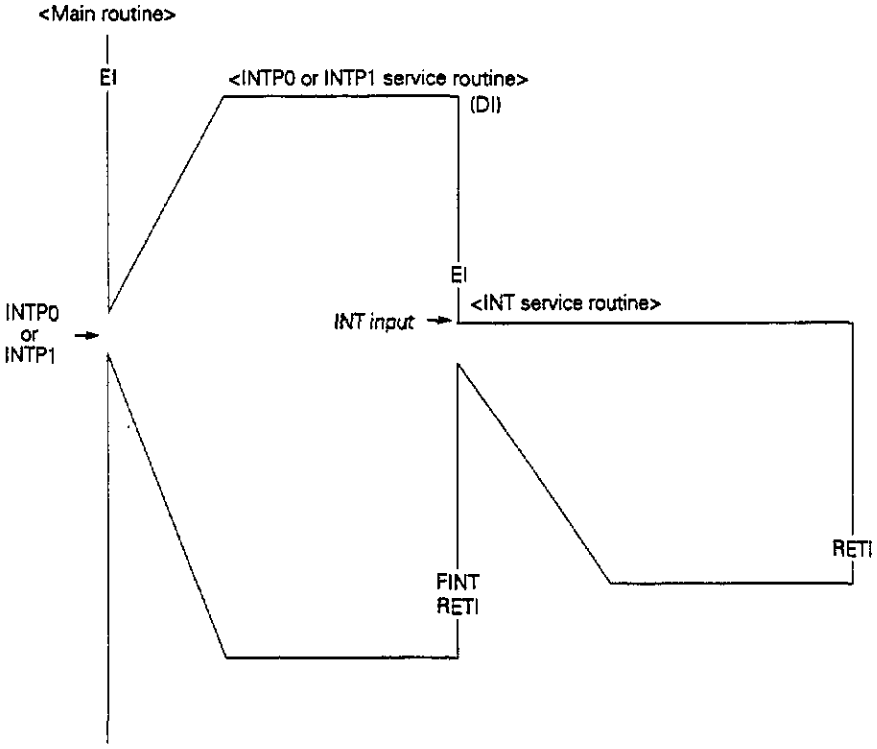


① Because INT is acknowledged and the interrupt disable state (DI) is set, even if an INTP0 or INTP1 interrupt request occurs, it is not acknowledged.

② When the interrupt flag is set to EI (enable interrupt), execution of the service routine for the pending INTP0 or INTP1 interrupt is started.

(2) If INTP0 and INTP1 are input at the same time during execution of the INT interrupt service routine, INTP0 is acknowledged and INTP1 is held pending because multiple interrupts within the same group cannot be executed.

(3) The INT interrupt is not subject to multiple interrupt servicing control. Therefore, the INT interrupt is always acknowledged if the interrupt flag is set to the EI state. Shown below is an example of an INTP0 or INTP1 and INT interrupt service routine.



Q.2.3

Register bank switching function

(1) Is the same register bank used for multiple interrupts within the same group? If so, can multiple interrupt servicing be performed?

(2) What types of interrupts can the register bank switching function not be used for?

(3) Does the CPU automatically switch register banks? If so, can the register bank contents before the register banks are switched be saved and restored?

(4) Interrupt request registers PR0 to PR2 specify a new register bank. Within the same group, the value is fixed to 7 (PR0 to PR2 = 1, 1, 1) except for the highest priority. Is it fixed to register bank 7 except for the highest priority?

A.2.3

(1) If a response by register bank switching is set for multiple interrupts within the same group, the interrupts respond by using the same register bank.

However, the V25/V35 cannot examine by occurrence of what interrupt a branch is taken to the service routine using the register bank, and cannot make two or more interrupts within the same group respond by register bank switching, but can perform the same interrupt servicing.

On the other hand, the V25+/V35+ can examine what interrupt is acknowledged by using the interrupt source register (IRQS) after register bank switching response.

(2) The register bank switching function cannot be used for NMI, INT, INTTB, or software interrupts (except for the BRKCS instruction).

(3) The CPU automatically switches register banks. At that time, PC and PSW are saved for the bank after switching, but other registers remain unchanged and are retained in the register bank before switching. Only PC and PSW are restored.

(4) It is not fixed to bank 7.

The bank number at register bank switching of interrupt sources making up the same group is determined by the interrupt request control register (PR0 to PR2) of the interrupt source having the highest priority level within the same priority level group.

Q.2.4

The user's manual says "FINT instruction is executed just before termination of interrupt servicing except for NMI, INT, or software interrupts". However:

(1) Is the FINT instruction required for input/output instruction interrupts and FPO instruction interrupts?

(2) May the FINT instruction be executed in any case other than just before termination of interrupt servicing?

A.2.4

(1) The FINT instruction is not required. The FINT instruction is an instruction used as a signal of the interrupt servicing termination for the internal interrupt controller, and therefore it needs to be executed in returning from interrupt servicing subject to priority level control from the interrupt controller.

(2) Be sure to execute the FINT instruction just before the specified RETI (or RETRBI) instruction.
The V25/V35 family contains a special function register for managing the interrupt priority levels and the FINT instruction resets the least significant bit (highest priority level) of the set bits of the interrupt priority register (ISPR) in the interrupt controller. Therefore, when the FINT instruction is executed, an interrupt having the same or lower level as the interrupt currently being serviced is acknowledged. (However, no interrupt is acknowledged between the FINT instruction and the next instruction.)
If the FINT instruction is not executed, priority level control cannot be performed accurately.

Q.2.5

When bit 7 (IF) of the interrupt request control register is set to 1 by software, how does the V25/V35 family operate (EI state)?

A.2.5

An interrupt (or macro service) occurs as an interrupt request is generated by the hardware.

Q.2.6

The macro service function transfers data between the special function register area and memory space according to an interrupt request. However:

(1) Can block transfer be executed by the macro service function?

(2) Can data be transferred between I/O and memory without using any special function register?

(3) When data is transferred between a special function register and memory, are the address and bus control signals externally output?

(4) What are application examples of the macro service function?

(5) What is the efficient use of macro service channels when a number of tasks are processed?

(6) When the MSC value (transfer count by macro service) is 0, what vector does an interrupt occurrence go to?

A.2.6

(1) When one interrupt request occurs, one data transfer (containing operation processing) is executed by the macro service function and block transfer cannot be executed by the function. However, block data processing can be performed by making a number of requests.

(2) DMA is used for transfer between I/O and memory. Macro service is used for data transfer between a special function register and memory.

(3) The address and bus control signals for memory are output when external memory is accessed. However, address and bus control signals are not output for special function registers.

(4) Application examples are given below.

- At given time intervals, data from a port is read and a search is made for the data.
- At given time intervals, data is output to a port.
- The interval time is changed according to an external request.

(5) Because macro service need not be held pending by priority level control, it is generally efficient to start using the macro service channels at 0 and the register banks at 7.

(6) If MSC is 0, the MS/$\overline{\text{INT}}$ bit of the interrupt request control register is set to 0. If the bit is 0, a vectored interrupt or an interrupt due to register bank switching occurs, depending on how the ENCS bit is set. When the ENCS bit is 0, a vectored interrupt is used, and when set to 1 the register bank switching function is used.

Q.2.7

Can wait states be inserted into an $\overline{\text{INTAK}}$ cycle?

A.2.7

Wait states cannot be inserted into the $\overline{\text{INTAK}}$ cycle.

In the V25/V35 family, the external interrupt controller is positioned as an additional function (for expansion) and the on-chip wait controller does not contain a wait state insertion function for interrupt acknowledge cycles.

Q.2.8

To read or write one word in the V25 family, the word is divided into the low-order byte and high-order byte. When an interrupt request occurs between reading the low-order byte and reading the high-order byte, is the interrupt acknowledged?

A.2.8

The interrupt request is not acknowledged until execution of one instruction terminates.

C.3 Bus Control

Q.3.1

Memory bank configuration of V35 family

(1) Why are the $\overline{\text{UBE}}$ and $\overline{\text{MSTB}}$ pins used to select a chip in memory bank selection? Why is the $\overline{\text{UBE}}$ pin required for the $\overline{\text{MSTB}}$ pin to read the low-order address?

(2) Are the A0 signal and $\overline{\text{UBE}}$ signal required in a read cycle?

(3) Why is the physical address of the A18 bit output in the second bus cycle of address time division output in the memory cycle?

(4) Which of the high-order and low-order bits of an address are output first?

(5) Can a program be fetched in byte units?

A.3.1

(1) The $\overline{\text{UBE}}$ pin is multiplexed with the A18 pin, and is activated at the timing when the $\overline{\text{MSTB}}$ pin is activated. Therefore, for high-order memory bank chip selection, the $\overline{\text{UBE}}$ pin and $\overline{\text{MSTB}}$ pin are ANDed with each other. The $\overline{\text{UBE}}$ pin is required to access only memory's high-order bank or memory's low-order bank during write operation into odd addresses in word units (read operation in byte units is performed twice) and write operation in byte units.

(2) The A0 signal and $\overline{\text{UBE}}$ signal are not required in read cycles.

(3) The A18 bit is output in order to output the address as early as possible in limitation of the number of bits.

(4) The A9 to A19 bits and A0 bit are output first.

(5) A program is always fetched in word units by signals. When an odd address is accessed, only the high-order byte of the address is read (the low-order byte becomes invalid).

Q.3.2

Access to on-chip RAM area or on-chip ROM area

(1) When the on-chip RAM area is accessed, can wait states be inserted into a bus cycle by setting the wait control register (WTC)?

(2) If a memory access is made to the V25/V35 on-chip ROM area, are bus address control signals externally output?

A.3.2

(1) The on-chip RAM area is always accessed with no wait, independently of how programmable wait is set. This also applies to the V25/V35's on-chip ROM area.

(2) If a program is fetched from an on-chip ROM area or a data access is made to an on-chip ROM area, no bus address control signals are externally output. When a program is fetched from the on-chip ROM area, the internal dedicated bus is used, and therefore the previous access cycle address is held for the external bus address.

Q.3.3

READY signal input

(1) When the number of wait states is fixed to 0, 1, or 2, if the READY signal goes low, does the CPU malfunction?

(2) With the V25, when two TAW states are inserted, the READY pin is set low, and one TW state is inserted, what is the timing specification?

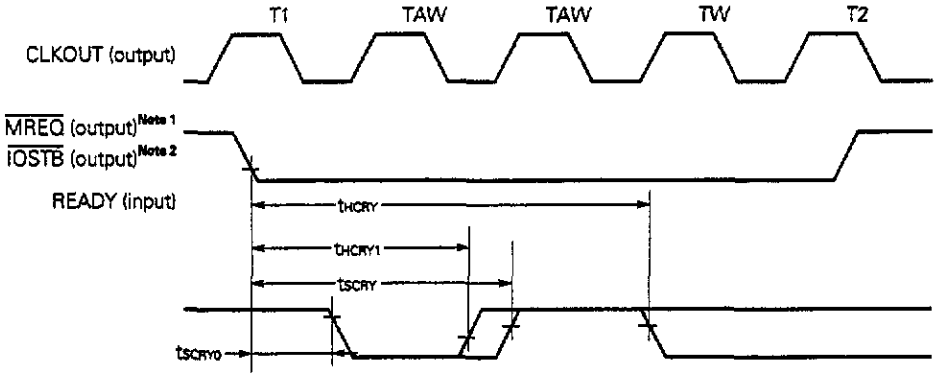
(3) If READY signal input is made asynchronous (when the t_{SCRY}, t_{HCRY} condition is not satisfied), does the CPU malfunction?

(4) *Is it possible to first deactivate the READY signal (low) and activate (high) only when a bus cycle is escaped?*

A.3.3

(1) When the number of wait states is fixed to 0, 1, or 2, READY signal input is ignored. Therefore, the CPU operates normally.

(2) For V25 READY signal input, satisfy the t_{SCRY} and t_{HCRY} specifications.



Notes 1. In memory cycle
2. In I/O cycle

(3) If READY signal input is made asynchronous, the state in which a wait state is entered or the state in which no wait is entered is set, and the CPU does not malfunction.

(4) It is possible. Even if the READY signal starts as inactive (low) and is active (high) only when a bus cycle is skipped, no problems will arise.

Q.3.4

In memory cycles

(1) What are the differences between the $\overline{\text{MREQ}}$ signal and $\overline{\text{MSTB}}$ signal in the V25 family?

(2) When the CPU accesses internal memory, are the $\overline{\text{MREQ}}$ and $\overline{\text{MSTB}}$ signals deactivated?

(3) In the V35 family, the $\overline{\text{MSTB}}$ signal is prolonged by wait insertion at the memory read timing. Why isn't the $\overline{\text{MSTB}}$ signal prolonged by wait insertion in a memory write cycle?

A.3.4

(1) To interface the V25 family with memory, an address is decoded to generate a memory chip select signal. At that time, the $\overline{\text{MREQ}}$ signal controls memory $\overline{\text{OE}}$.
The $\overline{\text{MSTB}}$ signal is used for an interface with DRAM. The $\overline{\text{MSTB}}$ signal controls the DRAM $\overline{\text{CAS}}$ signal. It is not necessary to use the $\overline{\text{MSTB}}$ signal for other than the DRAM interface.

(2) When the CPU accesses internal memory, the $\overline{\text{MREQ}}$ signal and $\overline{\text{MSTB}}$ signal are deactivated (high).

(3) The width of the $\overline{\text{MSTB}}$ signal is fixed regardless of whether or not a wait is inserted in the write cycle. With the V35 family, to interface DRAM, the $\overline{\text{CAS}}$ signal (memory input signal) is generated by the $\overline{\text{MSTB}}$ signal. To execute DMA transfer from I/O (slow output) to memory, the $\overline{\text{MSTB}}$ signal is delayed to write data into DRAM on the $\overline{\text{CAS}}$ signal's falling edge ($\overline{\text{MSTB}}$ signal's falling edge).

Q.3.5

When moving from a DMA cycle to a CPU bus cycle (fetch or data access), is an idle cycle entered?

A.3.5

No idle cycle is entered.

If a program is already prefetched at the DMA cycle termination, the next bus cycle (prefetch) is executed without entering an idle cycle.

Q.3.6

In the V25 family user's manual, data is read or written on the CLKOUT signal's falling edge in the I/O read/write cycle drawing. When designing, is it necessary to make the same drawing as in the user's manual?

A.3.6

For the I/O read/write cycle timing, refer to the AC characteristics described on the Data Sheet. When designing, the CLKOUT signal need not be used. The I/O read/write cycle timing is generated by the $\overline{\text{IOSTB}}$ signal.

Q.3.7

In the V25 family memory read timing

(1) Is it necessary to satisfy t_{DADR}, t_{DMRD}, and t_{DMSD}?

(2) Is speed within the data delay time t_{DMRD} required from the $\overline{\text{MREQ}}$ signal's falling edge?

(3) Is data read on the $\overline{\text{MREQ}}$ signal's rising edge (t_{WMRL})?

A.3.7

(1) t_{DADR}, t_{DMRD}, and t_{DMSD} must be satisfied. (See section **Q.1.2**)

(2) Speed within t_{DMRD} is required.

(3) Data is read on the $\overline{\text{MREQ}}$ signal's rising edge.

Q.3.8

Are t_{DAIS} and t_{DAMR} prolonged by wait insertion at the V25 family memory and I/O read/write cycle timing?

A.3.8

t_{DAIS} and t_{DAMR} are not prolonged, even if a wait is inserted.

Remark If the AC characteristics change by wait insertion, the specification is represented by a function of the number (n) of wait states.

C.4 DMA Controller

Q.4.1

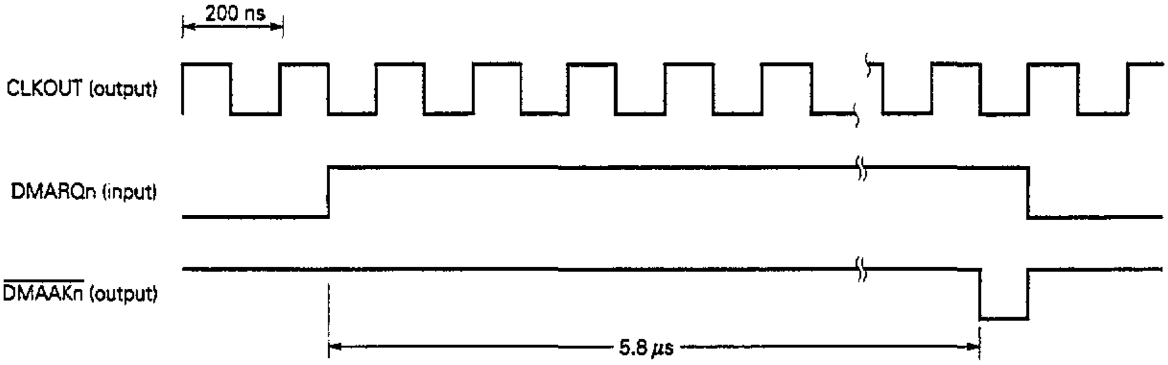
What is the MIN. value of the required time from DMARQ signal input to $\overline{\text{DMAAK}}$ signal output at the V25/ V25S DMA transfer timing?

A.4.1

The MIN. value of the required time from DMARQ signal input to $\overline{\text{DMAAK}}$ signal output on the v25/V25S is: N denotes the number of required clock cycles for the instruction being executed.

- One transfer mode: 29 + N clock cycles
- Other modes: 27 + N clock cycles

At the following timing, 29 × 200 ns = 5.8 μs is required until the $\overline{\text{DMAAK}}$ signal is output when one clock cycle is 200 ns in the one transfer mode (with no wait).



(when bus hold function, refresh function is not used)

Q.4.2

DMA mode register

(1) If the EDMA bit is 0 when the DMARQ signal is input, is a DMA request acknowledged?

(2) Is there a specification for DMARQ signal input for register setting?

(3) Can temporary stop of DMA transfer be controlled by setting the EDMA bit?

A.4.2

(1) If the EDMA bit of the DMA mode register is 0, DMA requests (DMARQ signal input) are ignored.

(2) The input delay time of the DMARQ signal for register setting is not specified.

(3) It can be controlled. If the EDMA bit is set to 0 to temporarily stop DMA transfer and then is set to 1, DMA transfer can be restarted.

Q.4.3

In the V25 family, how is an access to I/O determined in DMA transfer from memory to I/O?

A.4.3

I/O is accessed by the $\overline{\text{DMAAK}}$ signal.

In DMA transfer between I/O and memory, the I/O which becomes the destination or source must be fixed by the hardware. In response to the $\overline{\text{DMAAK}}$ signal, the I/O recognizes that a DMA response follows.

Q.4.4

Differences between V25/V35 and V25+/V35+

(1) Do they differ in their DMA transfer addressing method?

(2) Is there a direct source of transfer rate improvement in DMA transfer on the V25+/V35+?

A.4.4

(1) The V25/V35 uses the segment specification method; the V25+/V35+ uses the linear addressing method.

(2) A direct source of transfer rate improvement is as to whether or not a dummy cycle exists on the timing chart.

The V25/V35 starts a DMA transfer cycle by the microprogram (basic software in the CPU) and performs request acknowledgment and transfer address control by software. Therefore, a dummy cycle is inserted before and after the DMA bus cycle.

On the other hand, the V25+/V35+ performs transfer processing by dedicated hardware, and therefore performs processing instantaneously as compared with the V25/V35; for this reason, no dummy cycle is inserted.

Because of these differences, the V25+/V35+ enables a drastic improvement in DMA response and transfer rates.

Q.4.5

Can the number of transfer bytes of one DMA be specified?

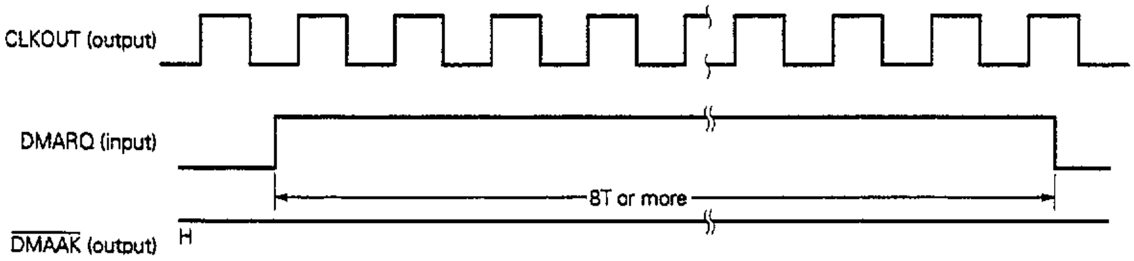
A.4.5

The number of transfer bytes and words for DMA transfer can be specified by setting the terminal counter (TC). In the byte transfer mode, 64 Kbytes can be transferred consecutively; in the word transfer mode, 128 Kbytes can be transferred consecutively.

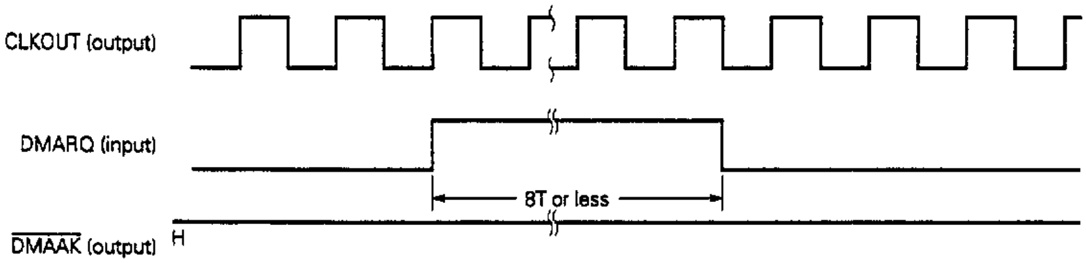
Q.4.6

When a DMA request is canceled, how does the CPU operate at the following timing (except in the demand release mode)?

① When the DMARQ signal is active for 8T or more and the DMA request is canceled before the $\overline{\text{DMAAK}}$ signal is output



② When the DMARQ signal is active within 8T (containing a short pulse) and the DMA request is canceled before the $\overline{\text{DMAAK}}$ signal is output



A.4.6

① Because the DMARQ signal is received on the DMARQ rising edge (when in single step, burst, or one transfer mode), the DMA request is acknowledged.

② The CPU does not operate abnormally, but may not acknowledge the DMA request.

Q.4.7

On the V25, how much time is taken from the DMA transfer termination to a DMA interrupt occurrence?

A.4.7

After setting DMA interrupt request control register DF0, DF1, the V25 executes the first instruction of the interrupt service routine in 76 clock cycles (MIN.) for a vectored interrupt or in 45 clock cycles (MIN.) for a register bank switching interrupt.

However, depending on the bus state at the last DMA transfer with TC = 0, program instruction fetch may be inserted before the interrupt acknowledge cycle starts just after the DMA transfer. At that time, interrupt servicing is delayed as long as the instruction fetch clock cycles.

Q.4.8

In DMA transfer in the burst mode, if a refresh request is made during V25+/V35+ DMA transfer, what does the DMA timing become?

A.4.8

If a refresh request is made during DMA transfer in the burst mode, a refresh cycle is inserted. A refresh cycle is inserted between 1-byte transfer cycles. At that time, no idle cycle is entered before or after it.

Q.4.9

In the single step mode and the burst mode, can asynchronous DMA transfer processing be performed such that with the DMARQ signal always set high, DMA processing is controlled by setting the EDMA and TDMA bits of the DMA mode register?

A.4.9

In the single step mode, it can be controlled (stopped) by setting the EDMA bit.

In the burst mode, once DMA transfer starts, instruction processing containing manipulation of the EDMA and TDMA bits is not performed, and therefore the DMA transfer cannot be stopped.

These modes are started on the rising edge of DMARQ signal input or by manipulating the TDMA bit. At that time, the EDMA bit must be set to the enable state.

For asynchronous operation, DMA transfer is also started by fixing the DMARQ signal high and manipulating the PMC2 register from the port mode to the control mode. However, TDMA bit control is recommended.

Q.4.10

In the one transfer mode

(1) When a memory address is output to the address bus, where is an I/O address specified?

(2) If the DMARQ signal is active while the EDMA bit of the DMA mode register is changed from 0 to 1, is a DMA request acknowledged?

(3) In the user's manual, transfer from memory to I/O is the only type of transfer described in the one transfer mode. Does transfer from I/O to memory differ from transfer from memory to I/O only in the R/$\overline{\mathrm{W}}$ signal level? For transfer from I/O to memory, is it necessary to address the DMA service channels?

(4) What is the MIN. value of the DMARQ signal pulse width to acknowledge a DMA request with the V25 family?

A.4.10

(1) In transfer from memory to I/O in the one transfer mode, no I/O address is output.
When DMA transfer is executed, the I/O requesting the DMA transfer is determined by the hardware. Therefore, if the $\overline{\mathrm{DMAAK}}$ signal is used as an I/O chip select signal, I/O can be selected.

(2) The DMA request is not acknowledged. In the one transfer mode, when the EDMA bit is 0, DMA requests are ignored.

(3) In the one transfer mode, transfer from I/O to memory differs from transfer from memory to I/O only in the R/$\overline{\mathrm{W}}$ signal output level.
In transfer between I/O and memory, I/O cannot be addressed by software, therefore any DMA service channel setting is insignificant.

(4) It is specified by AC characteristic t_{WIQH}.

Q.4.11

V25+/V35+ demand release mode

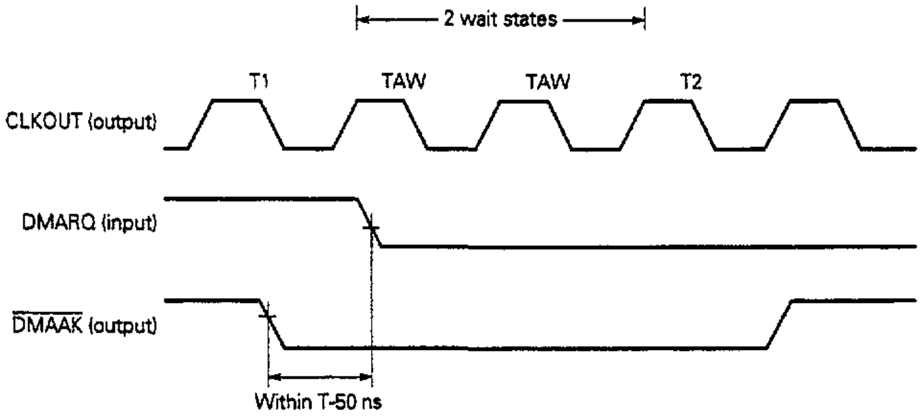
(1) When consecutive transfer is executed one byte at a time under the DMARQ signal control, does the CPU bus cycle operate in any period other than DMA transfer cycle?

(2) What control is performed for DMA transfer by using DMARQ signal control?

A.4.11

(1) The V25+/V35+ processes DMA transfer using dedicated hardware, and therefore the CPU bus cycle operates in any period other than the DMA transfer cycle.

(2) To execute DMA transfer in the demand release mode, stop control must be performed by the DMARQ signal. In the stop control, the number of DMA transfer cycle wait states must be two or more and the DMARQ signal must be turned off within (T-50) ns (t_{SDADQ}) after the $\overline{\text{DMAAK}}$ signal's falling edge in the last cycle before the transfer stops. (See below.)
Therefore, to ensure that the transfer is stopped in the last DMA transfer cycle, the DMARQ signal exceeding (T-50) ns is masked after the $\overline{\text{DMAAK}}$ signal's falling edge.
To ensure that DMA is started once or more, hold the DMARQ signal high until the $\overline{\text{DMAAK}}$ signal's falling edge.



C.5 Clock Generator

Q.5.1

Processor control register (PRC)

(1) If the oscillation frequency dividing ratio is changed by setting the PCK0 and PCK1 bits, is CPU operation affected?

(2) If an internal RAM access is disabled by setting the PRC RAMEN bit, can the external memory of the same address be accessed? At that time, can the register bank be used?

A.5.1

(1) Even if the oscillation frequency dividing ratio is changed by setting the PCK0 and PCK1 bits, no problems arise in CPU operation. The frequency can be changed by software.

(2) If the RAMEN bit is set to 0 to disable accessing the internal RAM, external memory is accessed regardless of the addresses.

The register bank can be used. A data access cannot be made as memory, but the register bank can be referenced as registers.

C.6 Timer Unit

Q.6.1

When the timer unit is set to the interval timer mode

(1) Are timers 0 and 1 fixed to the 16-bit full count mode?

(2) Is it possible to make the period set in the TM0 register of timer 0 a square wave and output the square wave to the TOUT pin?

A.6.1

(1) Timers 0 and 1 are not fixed to the 16-bit full count mode. Any desired count operation can be performed by setting count values in the MD0 and MD1 registers.

(2) In timer 0 in the interval timer mode, the TM0 register serves as a timer (counter) and a period is set in the MD0 bit. The square wave inverted in the period can be output to the TOUT pin. Square waves that are not 50% duty square waves cannot be output.

Q.6.2

If the system clock frequency (f_{CLK}) is 8 MHz in the interval timer mode on the V25, when TCLKn bit = 0 of the timer control register is used, what value is set in the MDn register to set the 20-ms interval timer value?

A.6.2

To set the 20-ms interval timer value when f_{CLK} = 8 MHz and TCLKn bit = 0, set the MDn register value to 682BH (26667).

Calculation example

TCLKn = 0: Count clock $f_{CLK}/6$

For use with 8 MHz, 8 MHz/6 ≒ 1.33 MHz (clock period: 0.75 μs)

The 20-ms interval timer counts as follows.

$$\frac{20\ \text{ms}}{0.75\ \mu\text{s}} \fallingdotseq 26667$$

Set MDn = 682BH (26667).

C.7 Serial Interface

Q.7.1

Does insertion of an idle time of one clock cycle or more between data units in data transmission/reception at a baud rate of 750 kbps or more mean insertion of one clock cycle between bits?

A.7.1

Insertion of the idle time of one clock cycle or more between data units means that the idle time of one clock cycle or more (shift clock) is required for each one-byte data transfer. In the asynchronous mode, an idle cycle with high level is required for one clock cycle or more. In the asynchronous mode, set one stop bit for V25/V35 family reception and two stop bits for the serial controller at the transmitting party for interfacing.

Q.7.2

To make a serial interface of the V25/V35 family with another microcomputer, data shifts one bit and is not restored due to mixing noise in the serial line, etc. What is a countermeasure for this symptom?

A.7.2

If data bit shift occurs due to noise, the serial register must be cleared and transfer must be executed again from the beginning.

To clear the serial register, perform either of the following.

① Write into the SCC0 register

② Change the MD0 bit of the SCM0 register (I/O interface mode → asynchronous mode → I/O interface mode)

Q.7.3

When using the asynchronous mode for transmission on the V25 family serial interface

(1) How long does it take for a start bit to be actually output after a macro service is started by a program?

(2) Assuming a baud rate of 4800 bps, how long does it take for a start bit to be actually output after data is set in a transmit buffer by a program?

A.7.3

(1) Described below is how to find the approximate time when on-chip RAM is enabled and byte transfer is executed.

① First, the IF flag is set at the execution start timing of the next instruction to the IF flag setting instruction.

② To execute a macro service (in normal mode)
The macro service is executed 11 clock cycles (MIN.) after the IF flag is set. Furthermore, it takes (24+W) clock cycles to execute data transfer from memory to a transmit buffer (TxBn). (W is the number of wait states.)

③ To execute a macro service (in character search mode)
The macro service is executed 11 clock cycles after the IF flag is set. Furthermore, it takes (27+W) clock cycles to execute data transfer from memory to a transmit buffer (TxBn), as in ②above.

④ When the macro service terminates and execution of the next instruction starts, data transfer to the transmit buffer (TxBn) executed in ②or ③ is actually executed.

⑤ A start bit is output on the rising edge of the next shift clock cycle after step ④terminates. The approximate time is found from the time of ②+⑤ or ③+⑤ if ① and ④can be ignored.

(2) When a write instruction into a transmit buffer (TxBn) is executed, data is transferred to the transmit buffer at the start timing of the next instruction. The serial interface operates independently of the instruction execution (system clock) and a start bit is output 1/f_{CLK} (seconds) to 1/4800 (seconds) after.

Q.7.4

For transmission in the I/O interface mode, can a transmit clock be input from the outside?

A.7.4

External transmit clocks cannot be input.

Q.7.5

In the V35 family, what is the macro service response time (maximum transfer rate) in the I/O interface mode?

A.7.5

The macro service response time used for the serial interface (request occurrence → transfer execution start) is as follows. (N is the remaining number of clock cycles of the instruction being executed when a request is acknowledged.)

For transmission and on-chip RAM enable

- Normal mode: 36+N to 52+N clock cycles
- Character search mode: 39+N to 55+N clock cycles

Therefore, the data exchange speed with a transmit buffer is limited by the response time.

This also applies to reception (but the number of clock cycles differs).

However, if bus hold, refresh cycle, etc., are contained, it is beyond the range.

Q.7.6

In serial interface interrupt requests

(1) What is the condition for clearing (to 0) bit 7 of the interrupt request control register (SEFn, SRFn, STFn)?

(2) What is the timing at which the SRFn bit is set to ON?

(3) Is register bank switching response enabled for serial interface interrupts?

(4) If a reception error interrupt occurs during reception completion interrupt servicing by a macro service, is the interrupt servicing executed so far then held pending?

A.7.6

(1) The SEFn, SRFn, and STFn bits are automatically cleared to 0 in their respective interrupt acknowledge cycles.

(2) The SRFn bit is set to ON when data is set in a receive buffer (RxBn) from the shift register.

(3) Register bank switching response is enabled for serial interface interrupts.

(4) The reception error interrupt has a higher priority level than the reception completion interrupt or transmission completion interrupt. The reception completion interrupt is held pending while the reception error interrupt is being serviced.

Therefore, for reception interrupt servicing by a macro service, the pending state is released by executing a FINT instruction, after which macro service is executed and data in the reception bank is transferred.

C.8 Standby Function

Q.8.1

How long is the oscillation stabilization time if the STOP mode is released by NMI?

A.8.1

The oscillation stabilization time is 30 ms.

Q.8.2

How much is the V_{DD} consumption current until the CPU operates, if the STOP mode is released by NMI?

A.8.2

The V_{DD} consumption current becomes the same as that in the HALT mode.

C.9 Reset Function

Q.9.1

How much is the V_{DD} consumption current if system reset is applied?

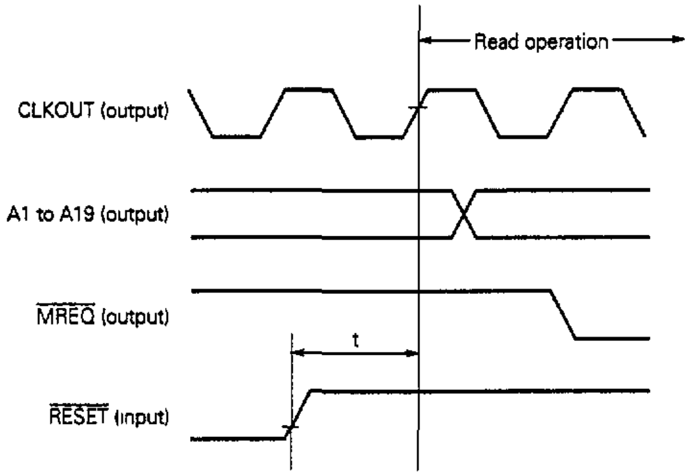
A.9.1

The V_{DD} consumption current when reset is the same as that in the normal operation mode.

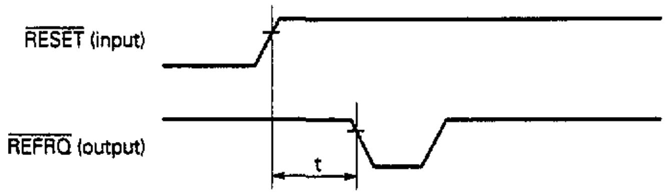
Q.9.2

In the V35 family, when the low-to-high transition of the $\overline{\text{RESET}}$ signal is made to release system reset

(1) How long does it take until the CPU operates?


Read operation
CLKOUT (output)
A1 to A19 (output)
$\overline{\text{MREQ}}$ (output)
t
$\overline{\text{RESET}}$ (input)

(2) How long does it take until the $\overline{\text{REFRQ}}$ signal is output?


$\overline{\text{RESET}}$ (input)
$\overline{\text{REFRQ}}$ (output)
t

A.9.2

(1) A branch is taken to the start address (FFFF0H) 21 clock cycles after the $\overline{\text{RESET}}$ signal goes high.

(2) When the refresh mode register (RFM) is in the state in which the $\overline{\text{RESET}}$ signal is input consecutively, the first $\overline{\text{REFRQ}}$ signal is output $2^4/f_{CLK}$ (seconds) after the system reset is released.

C.10 Other

Q.10.1

Can a program distinguish between the V25 and V25+ and between the V35 and V35+?

A.10.1

They can be distinguished from each other by reading the contents of address xxF6BH or xxF7BH of the special function register area just after reset, as listed below.

	F6BH	F7BH
V25, V35	00H (SCE0)	00H (SCE1)
V25+, V35+	60H (SCS0)	60H (SCS1)

However, because for start bit (bit 7), the RxD0 and RxD1 pins' state is monitored intact, the actually read value depends on the state of the RxD0 and RxD1 pins.

For functional differences between the V25 and V25+ and between the V35 and V35+, see the V25+/V35+ user's manual.

Q.10.2

What are the differences between the V25 and V25S and between the V35 and V35S?

A.10.2

The V25S/V35S contains two operation modes of N mode and S mode for the opcode conversion function.

The N mode is a mode for executing V25/V35 opcodes. The S mode is a mode for executing user-defined opcodes.

For other differences, see the V25S/V35S user's manual.

Q.10.3

Why is the IC pin fixed high with an external pull-up resistor?

A.10.3

The IC pin is connected internally. It is recommended to connect a pull-up resistor for protection of the pin against external input such as a surge.

Q.10.4

What do EA and T of "EA + 6 + T" represent in the instruction execution time (number of clock cycles)?

A.10.4

EA denotes the number of clock cycles which varies depending on the memory addressing mode. T denotes the number of wait states.

Q.10.5

For transfer instruction MOV, why are no wait states inserted with the number of clock cycles for on-chip RAM access disable being "EA + 2" for "MOV mem reg"?

A.10.5

The V25/V35 family performs three-stage pipeline processing as follows.

PAU	→ EXU	→ BCU
(Effective address calculation)	(Operation execution)	(Bus driver)

The number of "MOV mem reg" instruction execution clock cycles is only the number of clock cycles required for the PAU and EXU, and when operation processing terminates, data is sent to the BCU for execution of the next instruction.