"Interrupts"—A Powerful Tool of the Biphase Communications Processor

National Semiconductor
Application Note 499
Mark Koether
May 1989

When you have only 5.5 μs to respond you have to act fast. This is the amount of time specified in the IBM 3270 Product Attachment Information document as the maximum time allowed to respond to a message in a 3270 environment. This 5.5 μs is why the DP8344 interrupts are specifically tailored for the task of managing a communications line and feature very short latency times. This article contains information that will help the user to take better advantage of the extensive interrupt capability found in the DP8344.

The DP8344 has two external and four internal interrupt sources. The external interrupt sources are the Non-Maskable Interrupt pin, ($\overline{NMI}$), and the Bi-directional Interrupt ReQuest pin ($\overline{BIRQ}$). A Non-Maskable Interrupt is detected by the CPU when $\overline{NMI}$ receives a falling edge. The falling edge is captured internally and the interrupt is processed when it is detected by the CPU as described later. $\overline{BIRQ}$ can function as both an interrupt into the DP8344 and as an output which can be used to interrupt other devices. When $\overline{BIRQ}$ is configured as an input an interrupt will occur if the pin is held low. Note that $\overline{BIRQ}$ is not edge sensitive and if the pin is taken back high before the interrupt is processed by the CPU then no interrupt will occur.

The internal interrupts consist of the Transmitter FIFO Empty (TFE) interrupt, the Line Turn Around (LTA) interrupt, the Time Out (TO) interrupt, and a user selectable receiver interrupt source.

The receiver interrupt source is selected from either the Receiver FIFO Full (RFF) interrupt, the Data Available (DA) interrupt, or the Receiver Active (RA) interrupt. The RFF interrupt occurs when the receive FIFO is full or if the receiver detects an error condition. This interrupt enables the user to handle packets of data as opposed to handling every data word individually. It also allows the program to spend additional time performing other tasks. However, since the RFF interrupt is only asserted when the receive FIFO is full, the LTA interrupt should be used in conjunction with RFF to allow the program to check the FIFO for additional words at the end of a message. The DA interrupt indicates valid data is present in the receive FIFO and also occurs if the receiver detects an error condition. It should be used when it is desirable to handle each data word individually. The DA interrupt also allows the program to utilize the time between receiving each data word for performing other tasks. The RA interrupt is asserted when the receiver detects a valid start sequence. It provides the user with an early indication of data coming into the receiver. This allows the program time to perform any necessary overhead activity before handling the receiver data. The RA interrupt is asserted approximately 90 transceiver clock cycles prior to data becoming available in the receive FIFO when using 3270 mode. Consequently, if the transceiver and CPU are operating at the same clock frequency, approximately 90 clock cycles (T-states) are available for interrupt latency and taking care of overhead prior to handling the received data.

A TFE interrupt occurs when the last word in the transmit FIFO is loaded into the encoder. This interrupt allows a program to continue working on another task while the transmitter is sending data. It is especially useful when sending a long message. When the transmit FIFO becomes empty the program is alerted by the TFE interrupt and may continue the message by loading additional words into the FIFO. This approach frees up a significant amount of processing time. For example, after the transmit FIFO is loaded it takes the transmitter approximately 264 transceiver clock cycles to send the starting sequence and two data words in 3270 mode. With the CPU operating at the transceiver clock frequency, the program has approximately 264 T-states available before the TFE interrupt will occur.

Once the TFE interrupt occurs the CPU has approximately 80 transceiver clock cycles to load the transmit FIFO in order to continue a multiframe message in 3270 mode. If the CPU is operating at the transceiver clock frequency, the program has approximately 80 T-states to accomplish the load operation. Since the load to the Receive/Transmit Register, {RTR}, only takes 2 T-states, 78 T-states are available for interrupt latency and processing overhead after the interrupt occurs.

The LTA interrupt provides an easy means for determining the end of a message. This allows a program to quickly begin transmitting after the end of a reception. The LTA interrupt indicates that the receiver detected a valid end sequence in 3270 mode of operation. In 5250 operating mode, the LTA interrupt occurs when the last fill bit has been received and no further input transitions are detected by the receiver. However, a LTA interrupt does not occur in 5250 or 8-bit non-promiscuous modes of operation unless an address match was decoded by the receiver.

The TO interrupt occurs when the CPU timer counts down to zero. The timer provides a flexible means for timing events. It is a sixteen bit counter which can be loaded by accessing CPU registers {TRH} and {TRL} and is controlled by the [TCS], [TLD] and [TST] bits in the Auxiliary Control Register, {ACR}.

After an interrupt occurs the event that generated it must be handled in order to clear the interrupt. The exception to this is $\overline{NMI}$. Since it is falling edge triggered, it is cleared internally when the CPU processes the interrupt. The actions necessary to clear the interrupts are listed in Table I.

In the case where $\overline{BIRQ}$ is asserted, the response will be dependent on the system design. Ordinarily, this response would involve some hardware handshaking such as reading or writing a specific data memory location. When internal interrupts become asserted there are specific actions which must be taken by a program to clear these interrupts. The RFF interrupt is cleared when the receive FIFO is no longer full and any errors detected by the receiver are cleared. Data is read from the receive FIFO by reading {RTR}. Reading the Error Code Register, {ECR}, clears any errors detected by the receiver. The DA interrupt is cleared when the receive FIFO is empty and any errors detected by the receiver are cleared. The RA interrupt is cleared by reading {RTR} or {ECR}. All three receiver interrupts are cleared when the transceiver is reset. In many cases, resetting the transceiver is the preferable response to an error detected

RRD-B30M105/Printed in U. S. A.

TABLE I. Clearing Interrupts

Interrupt	How to Clear Interrupt
$\overline{\text{NMI}}$	Internally Cleared When Recognized by the CPU.
RFF	Read {RTR} When Receive FIFO is Full. Read {ECR} When an Error Occurs. Read {ECR} and {RTR} When an Error Occurs and Receive FIFO is Full. Reset the Transceiver. Reset the DP8344.
DA	Read {RTR} When Receive FIFO is Not Empty. Read {ECR} When an Error Occurs. Read {ECR} and {RTR} When an Error Occurs and Receive FIFO is Not Empty. Reset the Transceiver. Reset the DP8344.
RA	Read {RTR} or {ECR}. Reset the Transceiver. Reset the DP8344.
TFE	Write to {RTR}.
LTA	Write to {RTR}. Reset the Transceiver. Reset the DP8344. Write a One to {NCF} Bit 4.
$\overline{\text{BIRQ}}$	System Dependent.
TO	Write a One to {CCR} Bit 7. Stop the Timer. Reset the DP8344.

by the receiver. The TFE interrupt is cleared by writing to {RTR}. Unlike the receiver interrupts, the TFE interrupt is asserted when the transceiver is reset. The LTA interrupt is also cleared by writing to {RTR} or resetting the transceiver. In addition, it may be cleared by writing a one to bit 4 of the Network Command Flags register, {NCF}. The last internal interrupt is TO. It is cleared by writing a one to bit 7 in the Condition-Code Register, {CCR} or by stopping the timer. Note that the timer reloads itself and continues to count after the interrupt has been generated regardless of whether a one is written to bit 7 in {CCR}.

With the exception of $\overline{\text{NMI}}$, all of the interrupts are disabled when the DP8344 is reset. In order to make use of the interrupts they must be enabled in software. Software enabling and disabling of the interrupts is performed by changing the state of the Global Interrupt Enable, [GIE], bit in {ACR} and the state of the individual interrupt mask bits in the Interrupt Control Register, {ICR}.

[GIE] is a read/write register bit and so may be changed by using any instruction that can write to {ACR}. In addition, the RET, RETF, and EXX instructions have option fields which can be used to alter the state of [GIE]. RET and RETF are the return instructions in the DP8344 and EXX is used to exchange register banks. The EXX instruction can set or clear [GIE] as well as leaving it unchanged. The RET and RETF instructions can restore [GIE] to the value that was saved on the address stack at the time the interrupt was recognized. They also provide the options of clearing or setting [GIE] or leaving it unchanged. [GIE] is cleared when an interrupt is recognized by the CPU in order to prevent other interrupts from occurring during an interrupt service routine. The [GIE] options described above facilitate enabling and disabling interrupts when returning from an interrupt service routine. The restore option is especially useful with the $\overline{\text{NMI}}$. Since a Non-Maskable Interrupt can occur whether [GIE] is set or cleared, the restore [GIE] option can be used in the return instruction to put [GIE] back to its state prior to the interrupt occurring.

As the name implies, [GIE] affects all the maskable interrupts. However, in order to use any of these interrupts they must be unmasked by changing the state of their associated mask bit in {ICR}. When set high, bits [IM0], [IM1], [IM2], [IM3], and [IM4] in {ICR} mask the receiver interrupt, TFE interrupt, LTA interrupt, $\overline{\text{BIRQ}}$ interrupt, and TO interrupt respectively. To enable an interrupt, its mask bit must be set low. The interrupts and associated mask bits are shown in Table II. These bits are set high when the DP8344 is reset. Bits [RIS1] and [RIS0] in {ICR} are used to select the source of the receiver interrupt as shown in Table III. Note that only one of these interrupts can be active as the source of the receiver interrupt.

TABLE II. {ICR} Interrupt Mask Bits and Interrupt Priority

Interrupt	Mask Bit	Priority
$\overline{\text{NMI}}$	—	Highest
RFF, DA, RA	IM0	
TFE	IM1	
LTA	IM2	
$\overline{\text{BIRQ}}$	IM3	
TO	IM4	Lowest

TABLE III. {ICR} Receiver Interrupt Select Bits

RIS1	RIS0	Receiver Interrupt Source
0	0	RFF
0	1	DA
1	0	Reserved
1	1	RA

As stated earlier, [GIE] is cleared when an interrupt is recognized by the CPU. This prevents other interrupts from occurring in the interrupt service routine. In cases where it is desirable to allow nesting of interrupts, [GIE] should be set high within the interrupt routine. An example of nesting interrupts is using the RA interrupt in the main program and switching to the RFF or DA interrupt in the RA interrupt routine. Note that the internal address stack is twelve words deep and there is no recovery from a stack overflow. Therefore, care should be taken when nesting interrupts.

When more than one interrupt is unmasked and asserted, the CPU processes the interrupt with the highest priority first. $\overline{\text{NMI}}$ has the highest priority followed by the receiver interrupt, TFE, LTA, $\overline{\text{BIRQ}}$, and TO. Therefore, if DA and $\overline{\text{BIRQ}}$ were both active, DA would be processed first followed by $\overline{\text{BIRQ}}$. However, if a higher priority interrupt occurred while the DA interrupt was being handled then it would be processed before $\overline{\text{BIRQ}}$. Each time the interrupts are sampled, the highest priority interrupt is processed first, regardless of how long a lower priority interrupt has been active. Interrupt priority is summarized in Table II.

A call to the interrupt address is generated when an interrupt is detected by the CPU. The address for each interrupt is constructed by concatenating the Interrupt Base Register, {IBR}, contents with the individual interrupt code as shown in Table IV. There is room between the interrupt addresses for a maximum of four instruction words. Normally, at each interrupt address there would be a jump instruction to an interrupt service routine. The return instruction at the end of the interrupt service routine would then return to the address at which the interrupt occurred. By changing {IBR} it is possible to locate the interrupt jump table in memory wherever it is convenient or for one program to use more than one interrupt jump table.

TABLE IV. Interrupt Vector Generation

Interrupt	Code
NMI	111
RFF, DA, RA	001
TFE	010
LTA	011
BIRQ	100
TO	101

Interrupt Vector

{IBR} Contents	0 0 0	Code	0 0
15 8		4	2 0

As mentioned previously, the interrupts are sampled in the CPU prior to the start of each instruction. To be precise, they are sampled by each falling edge of the CPU clock with the last falling edge prior to the start of the next instruction determining whether an interrupt will be processed. The timing of a typical interrupt event is shown in *Figure 1*. The interrupt occurs during the current instruction and is sampled by the falling edge of the CPU clock. The next instruction is not operated on and its address is stored in the internal address stack. In addition, the current state of [GIE] and the states of the ALU flags and bank positions are stored in the internal address stack. A 2 T-state call is now executed in place of the non-executed instruction. This call will cause a branch to the interrupt address that is generated in the first half of T-state T1. [GIE] is then cleared during the first half of T-state T2. From this description it is evident that the shortest interrupt latency is 2.5 T-states. This assumes that an interrupt occurs during the first half of T2 and is sampled by the next falling edge of the CPU clock. However, a number of factors can increase the interrupt latency. If the interrupt misses the setup time to the falling edge of the last CPU clock the response time will increase by a minimum of 2 T-states. This increase is caused by the execution of one additional instruction. Of course, if the additional instruction takes more than 2 T-states to execute the interrupt latency will be greater.

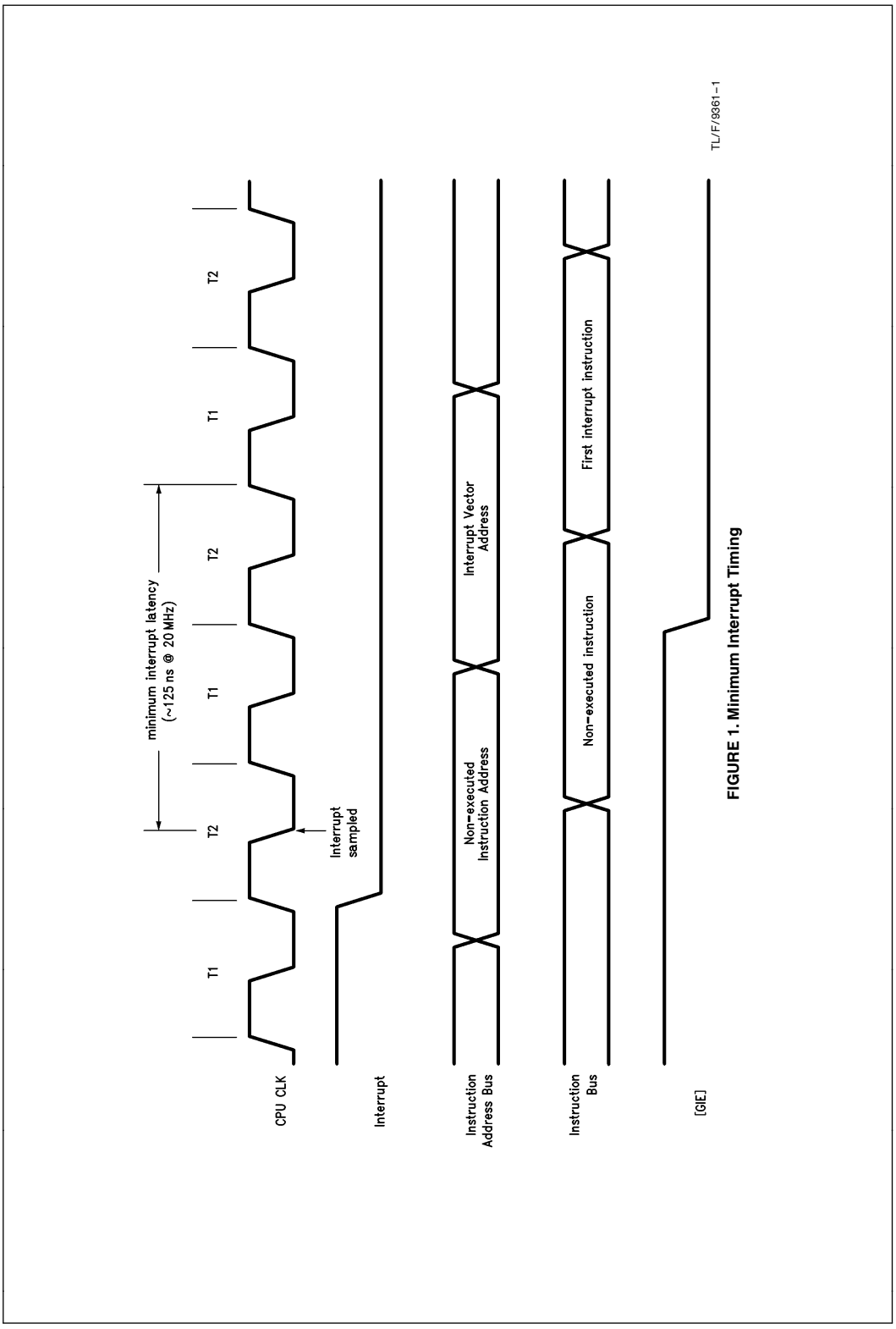

FIGURE 1. Minimum Interrupt Timing

Running the DP8344 with wait states will also increase interrupt latency. Instruction memory wait states increase latency by increasing the length of each instruction, including the call to the interrupt service routine. Data memory wait states will increase interrupt latency if an interrupt must wait for an instruction which accesses data memory to execute before it can be processed. A less obvious factor that can increase interrupt latency is data memory accesses by the remote system. If the DP8344 is attempting a data memory access and the remote system already has control of the data memory bus, the CPU will be waited. If an interrupt occurs at this time it will not be processed until the DP8344 is able to complete the instruction which is accessing data memory. This implies that a system with a lot of data memory arbitration occurring between the DP8344 and the remote system may have a longer average interrupt latency. The worst case interrupt latency will occur when the external $\overline{\text{LOCK}}$ or $\overline{\text{WAIT}}$ pins are asserted. Clearly, if the CPU is stopped by the assertion of the $\overline{\text{WAIT}}$ pin any interrupts ocurring will not be processed until the CPU is released from the wait state. Asserting the $\overline{\text{LOCK}}$ pin would have the same affect if the DP8344 attempts to make a data memory access. Note that interrupts are not disabled or cleared when the CPU is stopped by the remote system deasserting [STRT] in the Remote Interface Configuration, {RIC}, register. When the CPU is restarted any asserted interrupts will be processed. From the above discussion it is evident that calculating the interrupt latency is not trivial and will be dependent on the program and the system.

The interrupts on the DP8344 are powerful tools for controlling events in a time critical environment. They are one of the many reasons why the DP8344 Bi-phase Communications Processor provides a superior solution to managing communications interfaces.

Lit. # 100499

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
1111 West Bardin Road
Arlington, TX 76017
Tel: 1(800) 272-9959
Fax: 1(800) 737-7018

National Semiconductor Europe
Fax: (+49) 0-180-530 85 86
Email: cnjwge@tevm2.nsc.com
Deutsch Tel: (+49) 0-180-530 85 85
English Tel: (+49) 0-180-532 78 32
Français Tel: (+49) 0-180-532 93 58
Italiano Tel: (+49) 0-180-534 16 80

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductor Japan Ltd.
Tel: 81-043-299-2309
Fax: 81-043-299-2408