EE 3232 DIGITAL SYSTEM III
6.5 MULTI-LEVEL INTERRUPT HANDLING

March 2000 CPD

OBJECTIVE

To implement a multi-level interrupt generation system that responds to interrupts generated by: an internal 80C188XL timer, an external 8254A timer and an external real time clock. The concepts developed here form the basis of all multi-tasking systems. Upon completion of this experiment you will have developed a template for more elaborate interrupt driven tasks such as printing, serial IO and analog IO.

REFERENCES

1. "Embedded Microprocessor System Design", K.L. Short, Chapter 14.
2. "Microprocessors and Interfacing", 2nd Edition, R.V. Hall, pp. 255-259 and 232-242.
3. "The 80C188XL Microprocessor Users Manual", available in the lab, also at http://developer.intel.com/design/intarch/manuals/
4. "The 82C206 Integrated Peripheral Controller", Section 8.1, also at http://www.chips.com/prodframe.htm (Select mature products and scroll to the 82C206)
5. Section 7.1 - 7.2, SBC188 Schematics Sheet 1/11 and 2/11.
6. Section 7.10, SBC188 Schematics Sheet 10/11.
7. "The SBC188 Users Manual", available in the lab.
1. "Paradigm Debug/RT - 186 Users Guide", available in the lab.
9. "Paradigm Locate Reference Manual", available in the lab.

EQUIPMENT

1. SBC188 board.
2. Host PC.
3. IO interface board.
4. One 26-pin ribbon cable.
5. One 10-pin ribbon cable.

BACKGROUND

In this lab you will investigate the hardware and software that allows a system to respond to a variety of periodic interrupts. The SBC188 has three cascaded interrupt controllers as shown in Figure 6.5.1, that permit a large number of external interrupts. Each interrupt request is processed by one or more of the cascaded interrupt controllers. Software will be developed and the interrupt controllers will be initialized such that each interrupt causes a branch of control to an interrupt handler where an LED is toggled.

The concepts developed in this experiment may be used to implement interrupt driven handshaking IO interfaces with the printer and serial IO devices of Experiments 6.2 and 6.3. It is

also possible to synchronize the analog IO operations in Experiment 6.4 using timer interrupts.

The source of one of the interrupts used is the internal 80C188XL timer associated with the 80C188XL control port, T1CON. The second interrupt source is an external 8254A timer that generates an interrupt request at the input of an 8259A interrupt controller configured as a ***master***. The third interrupt source is an external real time clock that generates an interrupt at the input of an 8259A interrupt controller configured as a ***slave***. Each of the interrupt sources will be programmed to generate periodic interrupt requests. A block diagram of the SBC188 interrupt sources and interrupt controllers used in this experiment appears in Figure 6.5.1. You should be able to verify the block diagram connections in Figure 6.5.1 by consulting the SBC188 schematics sheets 10/11, 1/11 and 2/11 and identifying INTSER1, INT0 and /INTA.

The Internal 80C188XL Timers

The 80C188XL microprocessor has three internal timers, Timer 0, Timer 1 and Timer 2. In this experiment Timer 2 will be used as a pre-scaler (with T2CMPA initialized to N) to divide the timer clock = (CPUCLOCK ÷ 4) MHz by N. The periodic output from Timer 2 will then be used to clock Timer 1 which is programmed (with T1CMPA initialized to M) to generate a periodic interrupt request of frequency = (CPUCLOCK ÷ 4) ÷ N ÷ M. CPUCLK = 16 MHz for the 80C188XL.

The timers operate by counting up from the present count value by 1 with each clock cycle. When the count value equals the maximum compare value, TxCMPA, the timer output goes low for one clock cycle causing an interrupt. On the next clock the count value is reset to 0, the timer output is set to '1' and the process repeats.

To operate Timer x, (x = 0, 1 or 2) under program control,

- First write the initial count value (usually 0) to the 16-bit port TxCNT using outport(TxCNT, 0). NOTE: the **starting or base address** of all internal ports in the 80C188XL is configured as 0xFF00 by default. The offsets are all listed on page 4-3 of the 80C188XL/80C186XL Microprocessor Users Manual.
- Next write the maximum compare value to the 16-bit port TxCMPA using outport(TxCMPA, value).
- Enable Timer x by writing the 16-bit control word to port TxCON. The bit formats for TxCON are described on pages 9-7 to 9-9 of the 80C188XL/80C186XL Microprocessor Users Manual.

80C188XL Interrupts and the 80C188XL Interrupt Controller

The 80C188XL microprocessor has four external interrupts: INT0, INT1, INT2 and INT3, and five internal interrupts generated by on-chip peripherals: Timer 0, Timer 1, Timer 2, DMA 0 and DMA 1. A detailed description of the 80C188XL interrupts is available in Chapter 8 of the 80C188XL/80C186XL Microprocessor Users Manual.

Internal Timer x interrupts (for x = 0, 1 or 2) are unmasked inside the 80C188XL by writing 0x0000 to the 16-bit port TCUCON and masked by writing 0x0008 to TCUCON. Refer to page 8-13 of the 80C188XL/80C186XL Microprocessor Users Manual.

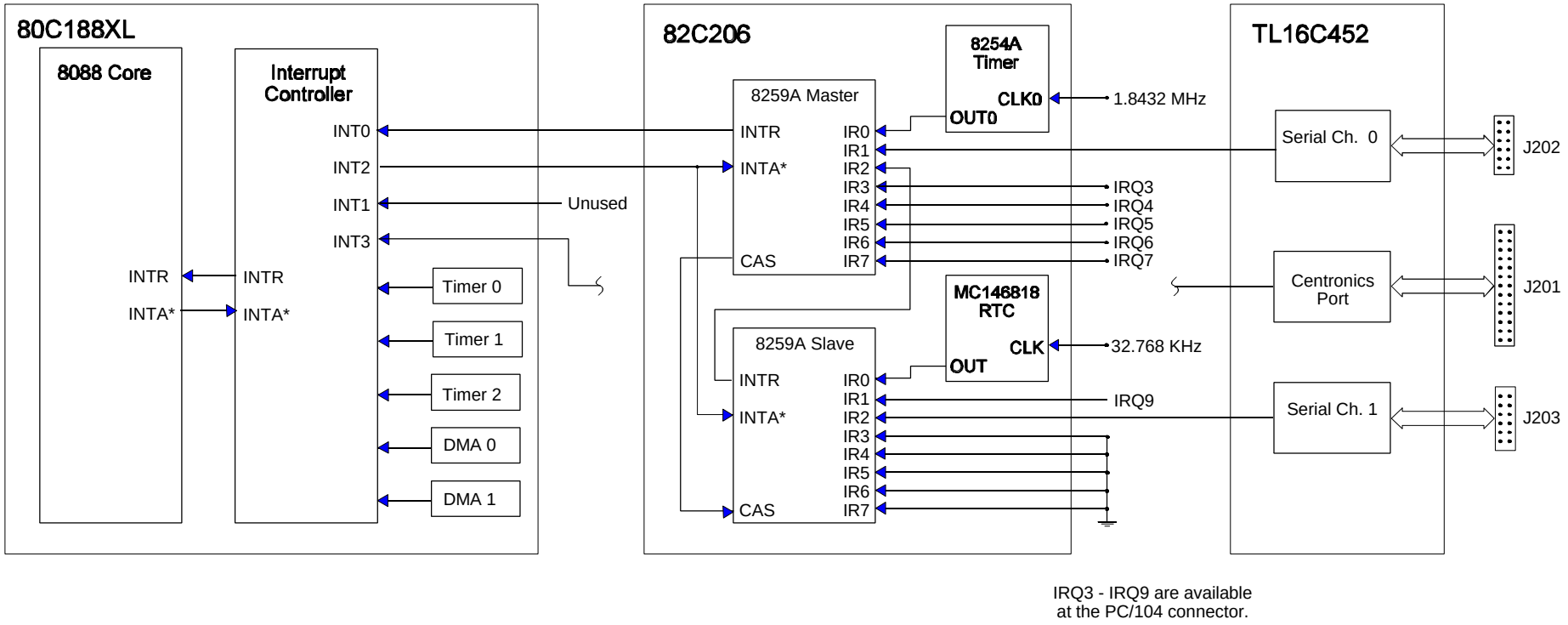

Figure 6.5.1. SBC188 Interrupt Hardware.

The SBC188 will be configured in this experiment so that an external (cascaded) 8259A interrupt controller is interfaced to the 80C188XL interrupt pin INT0. To unmask interrupt requests at pin INT0 (which is connected to a ***master*** 8259A interrupt controller), write 0x0030 to I0CON (address 0xFF38) to program hi-level trigger, cascade mode, not fully nested with priority 000B, (highest). Refer to page 8-15 of the 80C188XL/80C186XL Microprocessor Users Manual. To mask interrupts at pin INT0 set bit-3 by writing 0x0038 to I0CON.

Upon completion of any interrupt handler (i.e., interrupt service routine) it is necessary to send an appropriate ***end of interrupt command*** to one or more of the interrupt controllers (this clears the proper bit of an associated in-service-register) to ensure proper priority resolution. If the interrupt handler was initiated by TIMER x, (x = 0, 1 or 2) then 0x0008 must be written to the 16-bit port EOI (address 0xFF22) as described on pages 8-21 and 8-22 of the 80C188XL/80C186XL Microprocessor Users Manual. If the interrupt handler was initiated by external pin INT0 then 0x000C must be written to the 16-bit port EOI (address 0xFF22) as described on pages 8-9 and 8-22 of the 80C188XL/80C186XL Microprocessor Users Manual.

The 82C206 Integrated Peripheral

The 82C206 Integrated Peripheral incorporates two 8259A interrupt controllers, an 8254A timer, an MC146818 real time clock and two 8237 DMA controllers as shown in Figure 6.5.1. In this experiment you will use both 8259A interrupt controllers, counter 0 of the 8254A and the real time clock. Data sheets for the 82C206 appear in Section 8.1.

The 8259A manages interrupts by accepting interrupt requests (at pins IR0-IR7) from peripherals, resolving priority on pending interrupt requests **and** interrupts in service, issuing interrupts to the CPU and supplying the interrupt type number that is used by the CPU to determine which interrupt handler is executed. The 82C206 contains two 8259A's that are cascaded internally, one configured as a master (with address 0x0020 - 0x0021) and the other as a slave (with address (0x00A0 - 0x00A1). The two devices must be programmed to operate in cascade mode for proper operation of all 16 interrupt requests. A connection exists inside the 82C206 between IR0 of the master 8259A and OUT0 of an 8254A timer to permit timer interrupts. A connection also exists inside the 82C206 between IR0 of the slave 8259A and the output of an on-chip MC146818 real time clock to permit real time clock interrupts as shown in Figure 6.5.1.

The initialization sequence for the interrupt generation system of the SBC188 is rather involved. A summary of the steps for cascaded, non-nested, ***normal end of interrupt*** mode follows. Refer to the 80C206 Data Sheets.

Initialization of the Slave 8259A (pages 22-29 of the 82C206 Data Sheets) :

- Write 0x19 to the 8-bit port ICW1 (mapped at address 0x00A0) to program level triggered cascade operation.
- Write the desired 8-bit interrupt type (base number) to the 8-bit port ICW2 (address 0x00A1) to program the base type number for interrupt requests at the slave interrupt controller.
- Write 0x02 to the 8-bit port ICW3 (mapped at address 0x00A1) indicating the slave controller address.
- Write 0x01 to the 8-bit port ICW4 (mapped at address 0x00A1) to program single interrupt nesting and normal end of interrupts.

Initialization of the Master 8259A (pages 22-29 of the 82C206 Data Sheets) :

- Write 0x19 to the 8-bit port ICW1 (mapped at address 0x0020) to program level triggered cascade operation.
- Write the desired 8-bit interrupt type (base number) to the 8-bit port ICW2 (mapped at address 0x0021) to program the base type number for interrupt requests at the master interrupt controller.
- Write 0x04 to the 8-bit port ICW3 (mapped at address 0x0021) indicating a slave controller connected to IR2.
- Write 0x01 to the 8-bit port ICW4 (mapped at address 0x0021) to program single interrupt nesting and normal end of interrupts.

- **N O T E:** After the 8259A slave and master interrupt controllers are programmed, the 80C188XL interrupt controller associated with I0CON must be initialized by writing 0x0030 to the 16-bit port I0CON (mapped at address 0xFF38) to program hi-level trigger, cascade mode, not fully nested with priority 000, (highest).

Unmasking the Master 8259A IR2 and Slave 8259A IR2:

- Write 0xFA (= 1111 1010 binary) to the 8-bit port OCW1 of the Master (mapped at address 0x0021) to unmask IR0 and IR2 in the master. Unmasking IR0 turns on 8254A timer interrupts. Unmasking IR2 permits the slave 8259A to interrupt the master 8259A.
- Write 0xFE (= 1111 1110 binary) to the 8-bit port OCW1 of the Slave (mapped at address 0x00A1) to unmask IR0. This permits real time clock interrupts.

Within the interrupt handler (i.e., interrupt service routine) there is a need to send an ***end of interrupt command*** to each of the interrupt controllers as follows (in order for further interrupts to be processed).

- Write 0x60 to the 8-bit port OCW2 in the Slave (mapped at address 0x00A0) to signify that the interrupt service routine for (slave) IR0 has finished execution, i.e., the real time clock interrupt routine.
- Write 0x62 to the 8-bit port OCW2 in the Master (mapped at address 0x00A0) to signify that the interrupt service routine for (master) IR2 has finished execution.
- **N O T E :** Since the Master 8259A drives INT0 of the 80C188XL it is necessary to write 0x000C to the 16-bit port EOI (16-bit port mapped at address 0xFF22) to signify that the interrupt service routine for hardware interrupt INT0 has finished execution.

The 8254A Timer (pages 29-35 of the 82C206 Data Sheets)

The 8254A timer consists of three independent counting elements called Counter 0, 1 and 2. We will only use Counter 0 which is clocked at input CLK0 by an external 1.8432 MHz oscillator on the SBC188. The counter output, OUT0, is connected directly to IR0 of the Master 8259A interrupt controller. This allows you to configure the 8254A for generating accurately timed interrupt requests.

By consulting the I/O Port Map (for the SBC188) you should find that the 8254A timer

(within the 82C206) has addresses of 0x0040 to 0x0043. To use Counter 0 for generating interrupts the mode of operation must first be programmed by writing an appropriate control word to the control register, (8-bit port mapped at address 0x0043), and then the Counter needs to be loaded with an initial count value to start counter operation.

For example to program Counter 0 for binary counting in Mode 0 - ***Interrupt on Terminal Count***, write 0x30 to the 8-bit port with address 0x0043. Next, write in sequence the least significant byte (or LSB) and then the most significant byte (or MSB) of the 16-bit count value, say N, into Counter 0 (mapped at the 8-bit port address 0x0040). The counter begins counting **down** from N with each clock pulse and when 0 is reached an interrupt is automatically generated at IR0 of the master 8259A. To generate another interrupt request the counter needs to be reloaded by re-writing the LSB and MSB of N in sequence to the 8-bit port 0x0040.

The Real Time Clock (pages 35-40 of the 82C206 Data Sheets)

The MC146818 real time clock has a built in time of day clock with alarms, a 100 year calendar and a (programmable) periodic interrupt. We will only use the (programmable) periodic interrupt function. To program interrupt generation by the real time clock, you must first set the INDEX register (8-bit port mapped at 0x0070) so it points to register A by writing 0x0A (8-bit port mapped at 0x0070). Then you need to specify the frequency that drives the RTC clock, input, (via bits DV2-DV0) and the bits that specify the periodic interrupt rate, RS3-RS0. Refer to pages 36-37 of the 82C206 Data Sheets for further detail.

By examining the SBC188 schematic sheet 10/11 you should verify that the RTC clock input is 32.768 KHz, therefore DV2-DV0 = 010B. To generate a periodic interrupt with a period of 500 ms for example (with an input clock of 32.768 KHz.) choose RS2-RS0 = 1111B. Therefore to complete the initialization write 0x2F to the RTC data register, (8-bit port mapped at 0x0071).

PREPARATION

1. Examine programs L351.C and LA351.ASM that toggles the LED connected to bit-5 of the 82C55A output port KEY_PC. The program initializes Timer 1 and Timer 2 (of the 80C188XL) for periodic interrupt generation. Each interrupt causes a branch of control to the interrupt service routine, **timer_80188_int_handler()** where LED-5 connected to the 82C55A output port, KEY_PC, is toggled. Carefully identify the program details for:

   - Initializing the 82C55A parallel port, PC.
   - Initializing the interrupt vector table.
   - Initializing the 80C188XL timers.
   - Initializing the timer and interrupt control registers, TCUCON and I0CON, within the 80C188XL.
   - Unmasking timer interrupts.
   - Issuing end of interrupt commands inside the interrupt service routine.
   - Masking timer interrupts (inside the interrupt service routine).

2. Examine programs L352.C and LA352.ASM that toggles the LED-6 connected to the 82C55A output port KEY_PC. The program initializes the 8254A timer and the master 8259A interrupt controller (within the 82C206) for periodic interrupt generation. Each interrupt causes a branch of control to the interrupt service routine, **timer_8254_int_handler()**. Carefully identify the program details for:

   - Initializing the interrupt vector table.
   - Initializing the 82C55A parallel port, PC.
   - Initializing the 8254A timer (within the 82C206).
   - Initializing the master 8259A interrupt controller, (within the 82C206).
   - Initializing the 80C188XL interrupt control register, I0CON.
   - Unmasking interrupts at IR0 in the master 8259A.
   - Issuing end of interrupt commands within the interrupt service routine.
   - Masking timer interrupts (inside the interrupt service routine).
   - Resetting the 8254A timer for counting in Mode-0 (inside the interrupt service routine).

3. Examine programs L353.C and LA353.ASM that toggles the LED-7 connected to the 82C55A output port, KEY_PC. The program initializes the the MC146818 real time clock and the master and slave 8259A interrupt controllers (within the 82C206) for periodic interrupt generation. Each interrupt causes a branch of control to the interrupt service routine, **rtc_int_handler()**. Carefully identify the program details for:

   - Initializing the interrupt vector table.
   - Initializing the 82C55A parallel port, PC.
   - Initializing the real time clock (within the 82C206).
   - Initializing the master and slave 8259A interrupt controllers, (within the 82C206).
   - Initializing the 80C188XL interrupt control register, I0CON.
   - Unmasking interrupts at IR2 in the master 8259A and interrupts at IR0 in the slave 8259A.
   - Issuing end of interrupt commands inside the interrupt service routine.
   - Masking real time clock interrupts (inside the interrupt service routine).

4. Create a program that combines the operation of the programs, L351.C, L352.C and L353.C such that the three timer interrupts are allowed to occur simultaneously - each toggling a different LED.

EXPERIMENT

1. Compile, link and locate the source programs L351.C, L352.C and L353.C. Now load each of the programs into the SBC188 and verify program operation.

2. Load, execute and debug the multi-level interrupt system developed in Part 4 of the Preparation.

SUMMARY

1. What is the function of an interrupt controller?

2. How does the (8259A type) interrupt controller resolve priority using the IRR and ISR?

3. How are the interrupt type numbers generated by the interrupt controllers?

4. What is the purpose of the ***end of interrupt command***?

EXTENSIONS: Converting Previous Experiments to Interrupt Driven IO

1. Using the results of this experiment modify the program PRN_PUT_STRING() in Lab 6.2 such that the elements of the string are printed one at a time from within an interrupt handler. Refer to the Section EXTENSIONS in Experiment 6.2.

2. Using the results of this experiment modify the program MOUSE_GET_PACKET() in Lab 6.3 such that when the serial receiver is ready (with a character to be read by the CPU) an interrupt is generated which branches control to an interrupt handler where the the next character is read and assembled into a packet. Refer to the Section EXTENSIONS in Experiment 6.3.

3. Using the results of this experiment modify the programs ANALOG_IO, FILTER_IO and/or PEAK_IO in Experiment 6.4 such that the sampling of analog input and output is synchronized with a periodic interrupt.