

COPYRIGHT

Copyright © 1983 by MULTITECH INDUSTRIAL CORP. All rights reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of MULTITECH INDUSTRIAL CORP.

DISCLAIMER

MULTITECH INDUSTRIAL CORP. makes no representations or warranties, either express or implied, with respect to the contents hereof and specifically disclaims any warranties or merchantability or fitness for any particular purpose. MULTITECH INDUSTRIAL CORP. software described in this manual is sold or licensed "as is". Should the programs prove defective following their purchase, the buyer (and not MULTITECH INDUSTRIAL CORP., its distributor, or its dealer) assumes the entire cost of all necessary servicing, repair, and any incidental or consequential damages resulting from any defect in the software. Further, MULTITECH INDUSTRIAL CORP. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of MULTITECH INDUSTRIAL CORP. to notify any person of such revision or changes.



Multitech
INDUSTRIAL CORP.

OFFICE
15FL, 135 CHIEN KUO N. ROAD, SEC. 2, TAIPEI
10479, TAIWAN, R.O.C.
TEL: (02) 505-5533
TELEX: 19162 MULTIC FAX: (02) 505-4451
FACTORY/
1 INDUSTRYE. ROAD, III.
HSINCHU SCIENCE-BASED INDUSTRIAL PARK
HSINCHU, TAIWAN 300, R.O.C.

TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION TO IOM-MPF-IP	1
1.	Installation	3
2.	Hardware Specification	4
CHAPTER 2	PIO APPLICATION EXAMPLE	7
1.	Introduction	9
2.	Operating Procedure	9
3.	Flow Chart	10
4.	Program Description	13
CHAPTER 3	CTC APPLICATION EXAMPLE	15
1.	Introduction	17
2.	Operating Procedure	23
3.	Flow Chart	27
4.	Program Description	35
CHAPTER 4	8251 APPLICATION EXAMPLE	37
1.	Introduction	39
2.	Operating Procedure	41
3.	Flow Chart	43
4.	Program Description	47
APPENDIX A	Z80-PIO	49
APPENDIX B	Z80-CTC	79
APPENDIX C	8251	109
APPENDIX D	IOM-MPF-IP HARDWARE CIRCUIT	127

Introduction

IOM-MPF-IP is the interface of MPF-IP for input/output control and memory expansion. It uses PIO (Parallel I/O Control) to control the parallel input and output, 8251 to control the serial input and output, CTC (Counter/Timer Circuit Control) to control the counter and timer.

In addition, a 4K EPROM 2732 and three 2016 (2K RAM) are used to expand the memory capacity.

The EPROM 2732 chip on the IOM-MPF-IP contains four programs, including one program for demonstrating the use of the PIO, two programs for demonstrating the application of the CTC, and one for 8251 application.

The program for demonstrating the application of the PIO starts at memory location B000H. That for demonstrating the application of the 8251 starts at memory location B300H. The two programs for demonstrating the application of the CTC start at memory location B100H and B700H, respectively.

After you have connected the MPF-IP with IOM-MPF-IP, you can run these programs by typing:

G <starting address> 

Note that you don't have to type the left and right caret -- "<" and ">". After pressing the  key, simply type in the starting address of the program which you want to execute.

1. Installation

- 1) Connect the connector J1 of MPF-IP and the connector J2 of the IOM-MPF-IP with a flat cable.
- 2) Plug the power cord of the MPF-IP into the wall outlet.
- 3) Plug the power cord of the IOM-MPF-IP into the wall outlet.

2. Hardware Specification

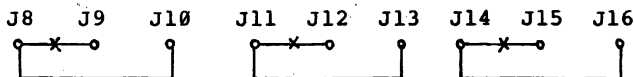
- 1) ROM: +5V 2732x1, total 4K bytes.

Basic ROM addresses: B000H-BFFFH

- 2) RAM: static RAM 2016x3, total 6K bytes.

Basic RAM addresses: D800H-EFFFH.

The basic RAM addresses may be changed to C000H-D7FFH by rewriting J8 through J16. when IOM-MPF-IP works together with EPB board. The rewriting is as follows:



Disconnect J8-J9, J11-J12, J14-J15 and connect J8-J10, J11-J13, J14-J16.

(Note: The EPB RAM address: D800H-EFFFH)

- 3) I/O port

- a. Programmable I/O port: 8251x1

I/O addresses: 60H-63H

- b. Programmable I/O port: PIOx1, which has two I/O ports, port A and B.

I/O addresses: 68H-6BH

- c. Programmable I/O port: CTCx1, which has 4 channels.

I/O addresses: 64H-67H

- d. Display: MPF-IP display.

- e. Keyboard: MPF-IP keyboard

- f. Audio tape interface: MPF-IP cassette interface.

- g. System power consumption: about 350mA.

- h. Power input: Power Adapter Input 110V/220V, output +9V/600mA.

- i. Interface connector/Cable: 40-pin flat cable and

male connector used to interface with MPF-IP.

- j. Extension connector: 40-pin flat cable connector provides CPU bus signals to other optional boards.
- k. PC board specifications: 157mm x 220mm x 1.6mm

CHAPTER 2

PIO APPLICATION EXAMPLE

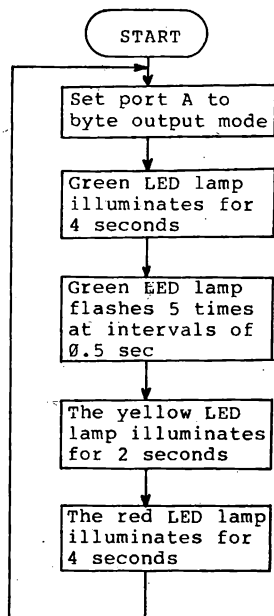
1. Introduction

This program uses Port A for the traffic light (red, yellow and green) control. First, set the PIO to byte output mode, then use subroutine SCAN1 to scan the display buffer. Because the execution of the SCAN1 subroutine takes 15.667 ms, You may set the value of the B register as required to control the duration of illumination of the display and the LED lamp.

2. Operating Procedure

- 1) Connect the socket TR1, TR2 and TR3 of J3 and the PA0, PA1, PA2 of J5 respectively with a cable.
- 2) Key in <G>=B000 
- 3) The screen displays "GREEN", and the green LED lamp which is on the IOM board illuminates for 4 seconds.
- 4) Then the word "GREEN" and the green LED lamp flashes five times at intervals of 0.5 sec.
- 5) Next, the word "YELLOW" appears, the yellow LED lamp illuminates for 2 seconds.
- 6) After that, the word "RED" appears with the red lamp illuminating for 4 seconds.
- 7) Then the word "GREEN" appears again with the green LED lamp.
- 8) The process cycles over and over again.
- 9) Pressing RESET key will stop the cycle.

3. Flow Chart



```

1  ;
2  ; *****
3  ; *
4  ; *      PIO DEMO PROGRAM      *
5  ; *
6  ; *****
7  ;
8  ; COPYRIGHT, MULTITECH INDUSTRIAL CORP. 1983
9  ; WRITTEN BY BING-CHUANG CHANG OF R&D DEPARTMENT
10 ; THE ADDRESS OF ROUTINE IS B000H.
11 ; THIS PROGRAM USES Z80-PIO AS TRAFFIC LIGHT CONTROLLER.
12 ; THE I/O ADDRESS OF PIO IS FROM 68H TO 6BH.
13 SCAN EQU 0246H ; UTILITY SUBROUTINE OF MPF_IP
14 SCAN1 EQU 029BH ; UTILITY SUBROUTINE OF MPF_IP
15 DEC_SP EQU 399H ; UTILITY SUBROUTINE OF MPF_IP
16 CLEAR EQU 989H ; UTILITY SUBROUTINE OF MPF_IP
17 DISP EQU 0FF84H ; THE BUFFER OF DISPLAY BUFFER
18 ; POINTER
19 MSG EQU 09CAH ; UTILITY SUBROUTINE OF MPF_IP
20 DISPBF EQU 0FF2CH ; DISPLAY BUFFER
21 CONVER EQU 0821H ; UTILITY SUBROUTINE OF MPF_IP
22 PIODA EQU 68H ; DATA PORT OF PIO PORT A
23 PIOCA EQU 6AH ; CONTROL PORT OF PIO PORT A
24 ;
B000 ; ORG 0B000H
B000 3E0F 25 LD A, 0FH
B002 D36A 26 OUT (PIODCA), A ; PIO PORT A WITH OUTPUT MODE
B004 CDB7B0 27 START CALL DISLED ; ALL LEDS DARKEN
B007 3EFE 28 LD A, 0FEH
B009 D368 29 OUT (PIODA), A ; GREEN LED LIGHT
B00B CDB909 30 CALL CLEAR ; CLEAR DISPLAY BUFFER,
; MAKE DISP INITIAL
; POSITION
31 ;
B00E 218EB0 33 LD HL, GREEN
B011 CDCA09 34 CALL MSG ; CONVERT ASCII CODE TO
; DISPLAY FORMAT
B014 CD9903 36 CALL DEC_SP ; DELETE CURSOR
B017 06FF 37 LD B, 0FFH ; SCAN & LIGHT 4 SEC
B019 DD212CFF 38 LD IX, DISPBF
B01D CD9B02 39 DISP1 CALL SCAN1 ; THE TIME OF SCAN1 IS
; ABOUT 15.667 MSEC
40 ;
B020 10FB 41 DJNZ DISP1
B022 0605 42 LD B, 5H ; GREEN LED FLASHES 5 TIMES
B024 C5 43 FLASH PUSH BC
B025 3EFF 44 LD A, 0FFH
B027 D368 45 OUT (PIODA), A ; GREEN LED DARKENS
B029 CDB7B0 46 CALL DELAY1 ; DELAY 0.5 SEC
B02C 3EFE 47 LD A, 0FEH
B02E D368 48 OUT (PIODA), A ; GREEN LED LIGHT
B030 CDB909 49 CALL CLEAR
B033 218EB0 50 LD HL, GREEN
B036 CDCA09 51 CALL MSG
B039 CD9903 52 CALL DEC_SP
B03C 0620 53 LD B, 20H ; GREEN LED LIGHT ABOUT
; 0.5 SEC
54 ;
B03E DD212CFF 55 LD IX, DISPBF
B042 CD9B02 56 DISP4 CALL SCAN1
B045 10FB 57 DJNZ DISP4
B047 C1 58 POP BC

```

LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	ASM 5.9
B04B	100A	59		DJNZ	FLASH	
B04A	C0B9B0	60		CALL	DISLED	
B04D	3EFD	61		LD	A, 0FDH	YELLOW LED LIGHT
B04F	D368	62		OUT	(PIODA), A	
B051	C0B909	63		CALL	CLEAR	
B054	219CB0	64		LD	HL, YELLOW	
B057	CDCA09	65		CALL	MSG	
B05A	CD9903	66		CALL	DEC_SP	
B05D	0680	67		LD	B, 80H	SCAN & LIGHT 2 SEC
B05F	DD212CFF	68		LD	IX, DISPB	
B063	CD9B02	69	DISP2	CALL	SCAN1	
B066	10FB	70		DJNZ	DISP2	
B068	C0B9B0	71		CALL	DISLED	
B06B	3EFD	72		LD	A, 0FBH	RED LED LIGHT
B06D	D368	73		OUT	(PIODA), A	
B06F	C0B909	74		CALL	CLEAR	
B072	21A0B0	75		LD	HL, RED	
B075	CDCA09	76		CALL	MSG	
B078	CD9903	77		CALL	DEC_SP	
B07B	06FF	78		LD	B, 0FFH	SCAN & LIGHT 4 SEC
B07D	DD212CFF	79		LD	IX, DISPB	
B081	CD9B02	80	DISP3	CALL	SCAN1	
B084	10FB	81		DJNZ	DISP3	
B086	C304B0	82		JP	START	
B089	3EFF	83	DISLED	LD	A, 0FFH	
B08B	D368	84		OUT	(PIODA), A	
B08D	C9	85		RET		
B08E	20202020	86	GREEN	DEFB	'	GREEN'
B09B	0D	87		DEFB	0DH	
B09C	20202020	88	YELLOW	DEFB	'	YELLOW'
B0AA	0D	89		DEFB	0DH	
B0AB	20202020	90	RED	DEFB	'	RED'
B0B6	0D	91		DEFB	0DH	
B0B7	01404A	92	DELAY1	LD	BC, 4A40H	
B0BA	EDAI	93	DELA1	CPI		
B0BC	00	94		NOP		
B0BD	00	95		NOP		
B0BE	E0	96		RET	P0	
B0BF	1BF9	97		JR	DELA1	
		98				
		99				

4. Program Description

- 1) Statements 25-26 define the port A to byte output mode.
- 2) Statements 27-41 use the bit PA0 to activate the green LED lamp and convert the ASCII code to the display format, and then execute subroutine SCAN1 for 255 times (It takes 15.667 msec for SCAN1 to execute once), so the green LED can illuminate for 4 seconds ($15.667\text{msec} \times 255 = 4\text{sec}$, $0FFH = 255$).
- 3) Statements 42-59 cause the green LED lamp to flash 5 times at an interval of 0.5 sec. ($15.667\text{ msec} \times 32 = 0.5\text{ sec}$, $20H = 32$).
- 4) Statements 60-70 use the bit PA1 to activate the yellow LED lamp to illuminate for 2 seconds.
- 5) Statements 71-81 use bit PA2 to activate the red LED lamp to illuminate for 4 seconds.

CHAPTER 3

CTC APPLICATION

EXAMPLE

1. Introduction

This program uses CTC as a clock which is in the Timer mode, and set 256 as Prescaler's value. In this program, you will notice that when the main program is doing looping (statements 125-127 whose function is scanning), the CTC is counting at the same time.

The value of Prescaler is 256, and the value of Time Constant Register is 0FFH, so the number of system clock is $256 \times 255 = 65280$. The frequency of system clock is $3579545/2 = 1789772$, and $1789772/65280 = 27$, so CTC must interrupt CPU 27 times to cost 1 second. As a result, it takes roughly one second for the CTC to interrupt the CPU 27 times. The time for the CTC to interrupt CPU once is about $1/27$ sec, that is, it will take 1 second to interrupt the CPU 27 times, so the program must define counter value as 1BH (1BH=27).

Time calculation: $(256 \times 255 \times 27) / 1789772 = 1$ (sec)

Deviation calculation: $1789772 - (256 \times 255 \times 27) = 27212$

Deviation per second: $(1/1789772) \times 27212 = 15.2$ msec

So the deviation is 1 second per 66 seconds, 545 seconds per hour.

1.1 Introduction to the Z80 CPU Interrupt

Before proceeding to the experimentation of the CTC, it is necessary for the reader to get familiar with the principles regarding the Z80 CPU interrupt.

The Z80 CPU can suspend the current program execution by using an external interrupt request. The CPU then starts executing the interrupt service routine. Once the service routine is completed, the CPU then returns to the main program from which it was interrupted.

The Z80 CPU has two interrupt inputs: a non-maskable interrupt and a software maskable interrupt.

1) NMI Request (Non-maskable Interrupt):

The non-maskable interrupt (NMI) line cannot be disabled by the programmer and will be activated whenever an external device inputs an interrupt request to it.

The NMI signal is sampled by the CPU at the rising edge of the last clock at the end of any instruction. The NMI request line will be at logic "0" if there is a non-maskable interrupt request. The CPU automatically saves the program counter (PC) in the stack area and jumps to location 0066H (a fixed memory address assigned by the Z80 CPU). The CPU will not respond to any further NMI requests. The CPU then executes the service routine until a RETN instruction appears and then it fetches the PC of the main program from the stack to continue the execution of the main program. At this time, the CPU will accept another NMI request.

In MPF-IP, memory addresses 0000H through 1FFFFH are reserved for the monitor program. Once a non-maskable interrupt is accepted, the CPU automatically jumps to location 0066H. The non-maskable interrupt request line has a higher priority than any other interrupt. It is very useful in the event of a power failure, which obviously takes precedence over all other activities. For instance, if the voltage level of the power supply battery of the microcomputer drops to a certain level, then a voltage comparator circuit will activate a non-maskable interrupt request signal. The

CPU then suspends its current program execution and starts battery-recharging. The recharging process is controlled by a software program. The starting address of this control sequence must be at 0066H.

2) INT Request (Maskable interrupt):

The maskable interrupt (INT) line can be disabled by resetting an internal interrupt Enable Flip Flop (IFF). The Enable Flip Flop can be set or disabled by the programmer using Enable Interrupt (EI) and Disable Interrupt (DI) instructions.

The interrupt request at the INT line can be masked. For instance, after the battery-recharging process has started, the CPU can return to its main program execution. When the battery is charged to a certain level, another voltage comparator circuit will generate an INT interrupt request signal. If the CPU is not executing a very important program, then it may acknowledge the interrupt requests and jumps to a service routine designed to stop the recharging process. Usually, stopping the recharging process is not an emergency task, thus, the CPU may continue to execute an important program and ignore this kind of interrupt request. For instance, when the CPU is reading data from a tape, interrupt will cause the data in the tape to be missed. Thus, if a DI instruction is included at the beginning of the "Read Tape" routine, then the INT interrupt request will be masked. An EI (Enable Interrupt) is usually included at the end of the "Read Tape" routine in order to enable the INT interrupt request line.

The Z80 CPU can be programmed to respond to the maskable interrupt in three possible modes, namely- 0=IM0, 1=IM1 and 2=IM2, by the IM (Interrupt Mode) instruction.

With the IM0 mode, whenever the CPU receives an instruction (usually, it is a "RESTART" operation) on the data bus from a peripheral device, then the CPU will jump to one of the 8 fixed memory addresses (0000, 0008, 0010, 0018, 0020, 0028, 0030 & 0038) and execute the program. In MPF-IP, mode 0 cannot be used because the addresses specified for the instructions are already reserved for the monitor program.

If the IM1 is selected by the programmer, the CPU will respond to an interrupt by executing a restart instruction to location 0038H.

The last mode is the IM2 mode which is the most powerful interrupt response mode. With this mode, the programmer maintains a table of 16-bit starting addresses for interrupt service routines. The low-order 8-bits of the pointer must be supplied by the interrupting device. The high-order 8 bits of the pointer is formed from the contents of the internal I register (Interrupt Vector Register). When an interrupt is accepted, the 16-bit pointer must be formed to obtain the starting address of the desired interrupt service routine from the table.

If the Z80 input/output interface devices (PIO, CTC, SIO) are used in the microcomputer system, then the IM2 mode will also be the most useful interrupt request response.

Example Experiments:

1. Testing the $\overline{\text{NMI}}$ interrupt response:

An interrupt request may be generated by touching a wire from the NMI input line of the CPU to the ground. After touching the NMI input line of the CPU, then the CPU will execute the program with starting address at 0066H.

2. Testing the $\overline{\text{INT}}$ interrupt response :

After a reset, the Z80 CPU will be automatically in the IM0 interrupt response mode and will disable the interrupt enable flip flop. Thus, before the CPU responds to the interrupt request, the following program must be executed.

```
IM 2           ; Select interrupt mode 2
LD A,0F8H
LD I,A         ; Assign F8H as the high-order byte
               ; of the interrupt vector address.
EI            ; Enable the interrupt request
               ; line INT.
```

In case the Z80 peripheral devices are not used in the system, the interrupt request signal is sent directly to the CPU. When the CPU acknowledges an interrupt request, the 8-bit data must be read in as the low-order byte of the vector address. If there is no electronic circuit for supplying this 8-bit vector address, then the data bus will be pulled up to "high" voltage state (logic "1") and read as FFH. That is, the CPU will form F8FFH as the 16-bit vector address. This 16-bit vector address is used as a pointer to obtain the starting address of the desired interrupt service routine from the table.

Suppose the starting address of the interrupt service routine is arranged at 0F920H, then the number 0F920H must be stored in memory addresses 0F8FFH and 0F900H. Load the following program into the MPF-IP for later testing.

LOC	OBJ	CODE	M	STMT	SOURCE	STATEMENT
F800				1	ORG	0F800H
F800	3EF8			2	LD	A,0F8H
F802	ED47			3	LD	I,A ; Define high-order ; vector address.
F804	2120F9			4	LD	HL,0F920H
F807	22FFF8			5	LD	(0F8FFH),HL ; store ; interrupt vector.
F80A	ED5E			6	IM	2
F80C	FB			7	EI	
F80D	F7			8	RST	30H ; Return to monitor ; program.
F920				9	ORG	0F920H ; Interrupt ; service routine.
F920	211234			10	LD	HL,3412H
F923	2240F9			11	LD	(0F940H),HL; store 3412H ; to RAM (0F940H).
F926	FB			12	EI	; Enable another ; interrupt.
F927	ED4D			13	RETI	; Return from ; interrupt.

- (1) Execute the above program by connecting a copper wire from the INT input line of the CPU to the ground. After the program is executed, the monitor will resume control of the microcomputer. The interrupt request line INT is also enabled. Then, key in some arbitrary numbers into RAM addresses 0F940H and 0F941H, and depress the INTR key in the keyboard. That is, an interrupt request signal is being input to the CPU INT line. Depress the AD key in the keyboard to reset the display buffer in the monitor program. Check if the interrupt service routine with starting address 0F920H is executed so that the designated numbers have been stored in RAM addresses 0F940H - 0F941H. Repeat the testing several times (change the contents of RAM before each test).

Results of test :

- (2) Instruction RETI is used as the end of an interrupt service routine. It is a routine to signal the I/O device that the interrupt routine has been completed. It facilitates the nesting of routines by allowing higher priority devices to suspend service of lower priority service routines. The standard Z80 I/O devices are not used in this experiment, thus, the RETI is not a necessary instruction. Replace instruction RETI in the above program by RST 30H and then repeat the test in (1). Record the result shown in the display after the interrupt request signal is input to the CPU. Discuss the results of the test.
- (3) Instruction EI (Enable Interrupt) must be included in every interrupt service routine, otherwise the INT line will be disabled after the CPU acknowledges an INT interrupt request. Instruction EI must be used to enable the maskable interrupt. The function of the EI instruction cannot be replaced by that of the RETI instruction.

Replace instruction with "Repeat" to test for interrupt request and to show that only the first interrupt is acknowledged and all other interrupts are ignored.

Results of test :

- (4) Write a program that will cause the PA0 line of the Z80 PIO to output "1" after the CPU receives an INT interrupt request and clear this line to "0" after 3 seconds have elapsed.

2. Operating Procedure

- 1) Input <M>=F800: HOUR MINUTE SECOND AMPFLG.
(AMPFLG=0 stands for AM, 1 stands for PM.)

AMPFLG=0 for AM
AMPFLG=1 for PM

Example: If the time now is AM, nine thirty and 20 seconds.

<M>= F800: 9 30 20 0

- 2) Press <G>=B100  , then CTC begins clicking.
3) The screen will display "AM 9 30 20" and go on clicking.
4) Press the **RESET** key to stop the timing.

If no PRT-MPF-IP is connected to the MPF-IP, you can run the CTC demonstration program which is stored in the EPROM starting from B100H.

If a PRT-MPF-IP is connected to the MPF-IP but the PRT-MPF-IP is turned off, you can still run the CTC demonstration program starting at B100H.

Executing the program for CTC application (which starts at B100H) on the MPF-IP while both the IOM-MPF-IP and PRT-MPF-IP (If the printer is on) are connected to the MPF-IP, will result in the PRT-MPF-IP operating automatically and going out of control. Aside from the PRT-MPF-IP being affected, the original CTC Demo Program also cannot function normally, it has a tendency to 'Run off'.

When the MPF-IP is connected with both an IOM-MPF-IP and a PRT-MPF-IP (with the printer "on"), you have to run the CTC application program starting at B700H.

The following is a brief analysis of this problem:

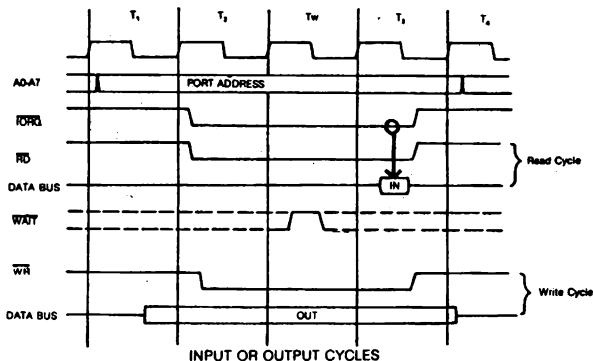


Fig. 1
Timing Diagram for Input & Output Cycles.

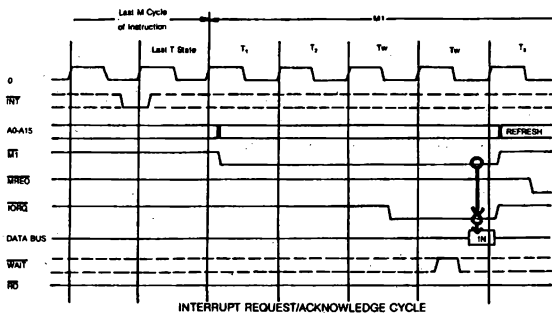


Fig. 2
I/O Timing Diagram for Interrupt
Request/Acknowledge Cycle.

From Fig. 1, we can see that the I/O Port is enabled when M1 = high and IORQ = low.

From Fig. 2, we can deduce that the I/O Port is enabled during an interrupt when M1 = IORQ = low. Since the CTC Demo Program starting at B100H employs Mode 2 Interrupt to implement an interrupt request, this may affect other I/O ports.

Mode 2 Interrupt processing expects an address from the interrupting device, this probably triggers on the printer and cause the program execution to be suspended.

This problem was not anticipated when designing the hardware for IOM-MPF-IP. So, this is where the second CTC Demo Program comes in, it not only resolves the problem regarding the printer but also provides the reader with a better understanding of the roles of the M1 and IORQ signals in an I/O cycle and their different functions in an Interrupt mode, plus better insight regarding the circuit design of the hardware.

To sum it up, we can now solve the problem affecting the PRT-MPF-IP by :

- I. Execute the new CTC Demo Program starting at memory address B700H instead of the original program which starts at memory location B100H, access to the new program can be done by:

Key in

Press

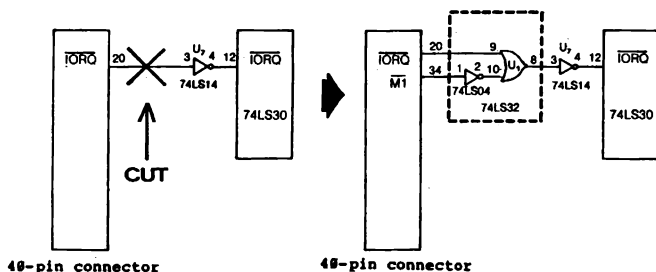
<M> = F800: HOUR MINUTE SECOND AMPFLG
(to store current time)

<G> = B700 
(to execute the program starting at B700H)

II. Should the reader prefers executing the original CTC Demo Program starting at B100H and yet wanted the problem regarding the printer eliminated, he can do so by using any of the following three methods:

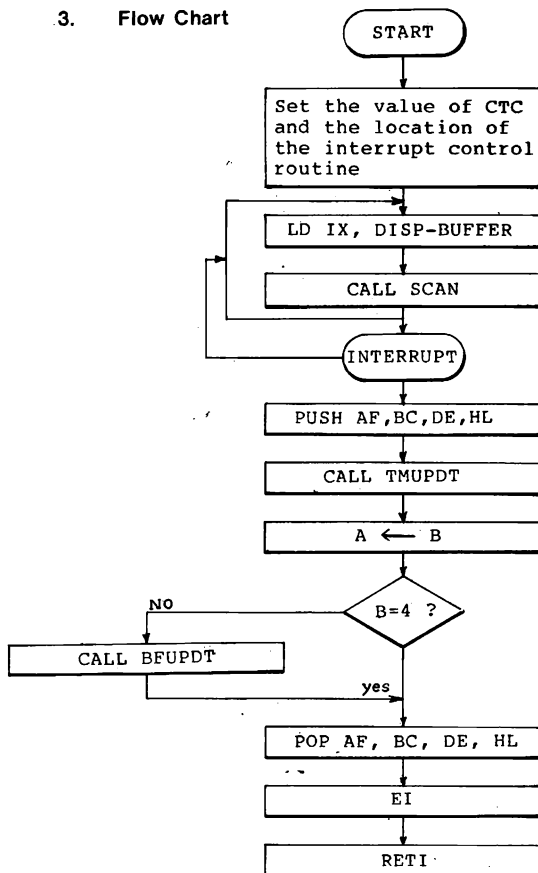
- 1) Disconnect the power line of the printer.
- 2) Modifying the hardware by changing the circuit of the PRT-MPF-IP as follows :

Change to



- a) Cut off the line between pin 20 of the 40-pin connector and pin 3 of U7 (74LS14).
- b) Connect pin 20 of the 40-pin connector to pin 9 of U1 (74LS32), and pin 34 of the 40-pin connector to pin 10 of U1 by passing through 74LS04.
- 3) Modifying the software. Rewrite the program by substituting instruction NOP into instructions already loaded in addresses xxCA, xxCB (CA & CB are PRT-MPF-IP's I/O Ports). The same procedure holds true for subroutines in the event that system subroutines are being used.

3. Flow Chart



```

100 ; *****
101 ; *
102 ; *          CTC DEMO_PROGRAM          *
103 ; *
104 ; *****
105 ;
106 ; THIS PROGRAM USES CTC TO DESIGN A CLOCK.
107 ; THE I/O ADDRESS OF CTC IS FROM 64H TO 67H.
108 CTC0 EQU 64H ; I/O OF CTC CHANNEL 0
B100 109 ORG 0B100H
B100 110 LD A,0B2H ; LOAD THE INTERRUPT REGISTER
B102 111 LD I,A
B104 112 LD A,10110101B ; LOAD THE CHANNEL CONTROL
B106 113 OUT (CTC0),A
B108 114 LD A,0FFH ; LOAD THE CONSTANT REGISTER
B10A 115 OUT (CTC0),A
B10C 116 LD A,0 ; LOAD THE INTERRUPT VECTOR
117 ; REGISTER
B10E 118 OUT (CTC0),A
B110 119 IM 2 ; SET INTERRUPT MODE 2
B112 120 EI
B113 121 CALL APMUP
B116 122 CALL HMSGUP
B119 123 LD A,0
B11B 124 LD (TMBF),A
B11E 125 LD IX,DISPBF
B122 126 CALL SCAN
B125 127 JR MAIN
B127 128 LD DE,TMBF
B12A 129 LD A,(DE)
B12B 130 INC A
B12C 131 LD (DE),A
B12D 132 CP 1CH ; INCREMENT SEC ONLY IF THE
133 ; NUMBER OF INTERRUPT REACHES
134 ; 28
B12F 135 LD B,4
B131 136 RET NZ
B132 137 XOR A
B133 138 DEC B
B134 139 LD (DE),A
B135 140 DEC DE
B136 141 DEC DE
B137 142 LD HL,MAXTAB
B13A 143 SCF
B13B 144 LD A,(DE)
B13C 145 ADC A,0
B13E 146 DAA
B13F 147 LD (DE),A
B140 148 SUB (HL) ; COMPARE WITH MAX_TABLE
B141 149 JR C,COMPL
B143 150 LD (DE),A
B144 151 CCF
B145 152 INC HL
B146 153 DEC DE
B147 154 DJNZ SMH
B149 155 LD A,(HOUR)
B14C 156 AND A
B14D 157 JR Z,SUCCESS ; IF REACH MAX, JUMP TO

```

```

158
B14F 3C 159 INC A ; ROUTINE SUCCESS
B150 FE13 160 CP 13H
B152 2015 161 JR NZ, HALF ; OVER TWELVE O'CLOCK
B154 3A03F8 162 LD A, (APMFLG)
B157 CB67 163 BIT 4, A
B159 201B 164 JR NZ, HOME
B15B EE01 165 XOR 01H ; CHANGE AM TO PM ; PM TO AM
B15D F610 166 OR 10H
B15F 3203F8 167 LD (APMFLG), A
B162 180F 168 JR CONTI
B164 3E01 169 SUCCES LD A, 1H
B166 3200F8 170 LD (HOUR), A
B169 3A03F8 171 HALF LD A, (APMFLG)
B16C E6EF 172 AND 0EFH
B16E 3203F8 173 LD (APMFLG), A
B171 1803 174 JR HOME
B173 CD20B2 175 CONTI CALL APMUP
B176 C9 176 HOME RET
B177 3A02F8 177 BFUPDT LD A, (SECOND)
B17A A7 178 AND A
B17B 2802 179 JR Z, JUDGE
B17D 1824 180 JR SUP
B17F 3A01F8 181 JUDGE LD A, (MINUTE)
B182 A7 182 AND A
B183 2802 183 JR Z, HMSUP
B185 180E 184 JR MSUP
B187 1138FF 185 HMSUP LD DE, DISPB+12 ; UPDATE HOUR, MINUTE, SECOND
186 ; DISPLAY BUFFER
B18A ED5384FF 187 LD (DISP), DE
B18E 0603 188 LD B, 3
B190 2100F8 189 LD HL, HOUR
B193 181A 190 JR LOOP
B195 113EFF 191 MSUP LD DE, DISPB+18 ; UPDATE MINUTE, SECOND
192 ; DISPLAY BUFFER
B198 ED5384FF 193 LD (DISP), DE
B19C 0602 194 LD B, 2
B19E 2101F8 195 LD HL, MINUTE
B1A1 180C 196 JR LOOP
B1A3 1144FF 197 SUP LD DE, DISPB+24 ; UPDATE SECOND DISPLAY BUFFER
B1A6 ED5384FF 198 LD (DISP), DE
B1AA 0601 199 LD B, 1
B1AC 2102F8 200 LD HL, SECOND
B1AF 3E30 201 LOOP LD A, 30H
B1B1 ED6F 202 RLD
B1B3 F5 203 PUSH AF
B1B4 CD2108 204 CALL CONVER ; CONVERT ASCII CODE TO
205 ; DISPLAY FORMAT
B1B7 F1 206 POP AF
B1B8 ED6F 207 RLD
B1BA F5 208 PUSH AF
B1BB CD2108 209 CALL CONVER
B1BE F1 210 POP AF
B1BF ED6F 211 RLD
B1C1 23 212 INC HL
B1C2 ED5B84FF 213 LD DE, (DISP)
B1C6 13 214 INC DE
B1C7 13 215 INC DE

```

LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	ASM 5.9
B1CB	ED5384FF		216	LD	(DISP), DE	
B1CC	10E1		217	DJNZ	LOOP	
B1CE	C9		218	RET		
B1CF	60		219	MAXTAB DEFB	60H	
B1D0	60		220	DEFB	60H	
B1D1	13		221	DEFB	13H	
B200			222	ORG	0B200H	
B200	02B2		223	DEFW	INTERRUPT	ENTRY POINT OF INTERRUPT
			224			SERVICE ROUTINE
			225	INTERRUPT:		
B202	F5		226	PUSH	AF	
B203	C5		227	PUSH	BC	
B204	D5		228	PUSH	DE	
B205	E5		229	PUSH	HL	
B206	CD27B1		230	CALL	TMUPDT	
B207	78		231	LD	A, B	
B20A	FE04		232	CP	4	
B20C	C477B1		233	CALL	NZ, BFUPDT	
B20F	E1		234	POP	HL	
B210	D1		235	POP	DE	
B211	C1		236	POP	BC	
B212	F1		237	POP	AF	
B213	F8		238	EI		
B214	ED4D		239	RETI		
B216	2020414D	AM	240	DEFM	' AM'	
B21A	0D		241	DEFB	0DH	
B21B	2020504D	PM	242	DEFM	' PM'	
B21F	0D		243	DEFB	0DH	
B220	CD8909	APHUP	244	CALL	CLEAR	
B223	3A03F8		245	LD	A, (APMFLG)	
B226	C847		246	BIT	0, A	
B228	2805		247	JR	Z, AMDECI	
B22A	211882		248	LD	HL, PM	
B22D	1803		249	JR	MIDWAY	
B22F	211682	AMDECI	250	LD	HL, AM	
B232	CDCA09	MIDWAY	251	CALL	MSG	
B235	CD9903		252	CALL	DEC_BP	
B238	C9		253	RET		
F800			390	ORG	0F800H	
F800			391	DEFB	1	
F801			392	DEFB	1	
F802			393	DEFB	1	
F803			394	DEFB	1	
F804			395	DEFB	1	
			254			

```

396 ;
397 ; *****
398 ; *
399 ; *          CTC DEMO_PROGRAM
400 ; *
401 ; *****
402 ;
403 ; THIS PROGRAM USES CTC TO DESIGN A CLOCK.
404 ; THE I/O ADDRESS OF CTC IS FROM 64H TO 67H.
405 DIG1 EQU 80H ; 8255 I PORT A
406 DIG2 EQU 81H ; 8255 I PORT B
407 DIG3 EQU 82H ; 8255 I PORT C
408 SEC1 EQU 90H ; 8255 II PORT A
409 SEC2 EQU 91H ; 8255 II PORT B
410 COLDEL EQU 80 ; COLUMN DELAY FOR ROUTINE SCANS.
411 CHRWR1 EQU 0724H
412 ORG 0B700H
413 LD A, 0B0H ; LOAD THE INTERRUPT REGISTER
414 LD I, A
415 LD A, 10110101B ; LOAD THE CHANNEL CONTROL
416 OUT (CTC0), A
417 LD A, 0FFH ; LOAD THE CONSTANT REGISTER
418 OUT (CTC0), A
419 LD A, 0 ; LOAD THE INTERRUPT VECTOR
420 ; REGISTER
421 OUT (CTC0), A
422 IM 2 ; SET INTERRUPT MODE 2
423 EI
424 CALL APMUP1
425 CALL HMBUP1
426 LD A, 0
427 LD (TMBF), A
428 MAIN1 LD IX, DISPF
429 CALL SCANS
430 JR MAIN1
431 TMUPD1 LD DE, TMBF
432 LD A, (DE)
433 INC A
434 LD (DE), A
435 CP 1CH ; INCREMENT SEC ONLY IF THE
436 ; NUMBER OF INTERRUPT REACHES
437 ; 28
438 LD B, 4
439 RET NZ
440 XOR A
441 DEC B
442 LD (DE), A
443 DEC DE
444 DEC DE
445 LD HL, MAXTA1
446 SCF
447 SMH1 LD A, (DE)
448 ADC A, 0
449 DAA
450 LD (DE), A
451 SUB (HL) ; COMPARE WITH MAX_TABLE
452 JR C, COMPL1
453 LD (DE), A

```

LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	ASM 5.9
B744			454	COMPL1	CCF	
B745	23		455		INC	HL
B746	1B		456		DEC	DE
B747	10F2		457		DJNZ	SMH1
B749	3A00F8		458		LD	A, (HOUR)
B74C	A7		459		AND	A
B74D	2B15		460		JR	Z, SUCCE1
			461			
B74F	3C		462		INC	A
B750	FE13		463		CP	13H
B752	2015		464		JR	NZ, HALF1
B754	3A03F8		465		LD	A, (APMFLC)
B757	C867		466		BIT	4, A
B759	201B		467		JR	NZ, HOME1
B75B	EE01		468		XOR	01H
B75D	F610		469		OR	10H
B75F	3203F8		470		LD	(APMFLC), A
B762	180F		471		JR	CONTI1
B764	3E01		472		LD	A, 1H
B766	3200F8		473		LD	(HOUR), A
B769	3A03F8		474	HALF1	LD	A, (APMFLC)
B76C	E6EF		475		AND	0EFH
B76E	3203F8		476		LD	(APMFLC), A
B771	1803		477		JR	HOME1
B773	CD20B8		478	CONTI1	CALL	APHUP1
B776	C9		479		RET	
B777	3A02F8		480	BFUPD1	LD	A, (SECOND)
B77A	A7		481		AND	A
B77B	2802		482		JR	Z, JUDGE1
B77D	1824		483		JR	SUP1
B77F	3A01F8		484	JUDGE1	LD	A, (MINUTE)
B782	A7		485		AND	A
B783	2802		486		JR	Z, HMSUP1
B785	180E		487		JR	MSUP1
B787	1138FF		488	HMSUP1	LD	DE, DISPB*12
			489			
B78A	ED5384FF		490		LD	(DISP), DE
B78E	0603		491		LD	B, 3
B790	2100F8		492		LD	HL, HOUR
B793	181A		493		JR	LOOP1
B795	113EFF		494	MSUP1	LD	DE, DISPB*18
			495			
B798	ED5384FF		496		LD	(DISP), DE
B79C	0602		497		LD	B, 2
B79E	2101F8		498		LD	HL, MINUTE
B7A1	180C		499		JR	LOOP1
B7A3	1144FF		500	SUP1	LD	DE, DISPB*24
B7A6	ED5384FF		501		LD	(DISP), DE
B7AA	0601		502		LD	B, 1
B7AC	2102F8		503		LD	HL, SECOND
B7AF	3E30		504	LOOP1	LD	A, 30H
B7B1	ED6F		505		RLO	
B7B3	F5		506	PUSH	AF	
B7B4	CD2108		507	CALL	CONVER	
			508			
B7B7	F1		509	POP	AF	
B7B8	ED6F		510	RLO		
B7BA	F5		511	PUSH	AF	
B7BB	CD2108		512	CALL	CONVER	
B7BE	F1		513	POP	AF	
B7BF	ED6F		514	RLO		
B7C1	23		515	INC	HL	

; IF REACH MAX, JUMP TO
; ROUTINE SUCCESS

; OVER TWELVE O'CLOCK

; CHANGE AM TO PM ; PM TO AM

; UPDATE HOUR, MINUTE, SECOND
; DISPLAY BUFFER

; UPDATE MINUTE, SECOND
; DISPLAY BUFFER

; UPDATE SECOND DISPLAY BUFFER

; CONVERT ASCII CODE TO
; DISPLAY FORMAT

IOM_HFF_IP					ASM 3.9	
LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	
87C2	ED5884FF		516		LD	DE, (DISP)
87C6	13		517		INC	DE
87C7	13		518		INC	DE
87C8	1802		519		JR	JUMP
87CA	00		520		NOP	
87CB	00		521		NOP	
87CC	ED5384FF		522	JUMP	LD	(DISP), DE
87D0	10D0		523		DJNZ	LOOP1
87D2	C9		524		RET	
87D3	60		525	MAXTA1	DEFB	60H
87D4	60		526		DEFB	60H
87D5	13		527		DEFB	13H
8800			528		ORG	08800H
8800	0288		529		DEFW	INT1
			530			ENTRY POINT OF INTERRUPT
			531			SERVICE ROUTINE
8802	F5		532	INT1:	PUSH	AF
8803	CS		533		PUSH	BC
8804	DS		534		PUSH	DE
8805	E5		535		PUSH	HL
8806	CD27B7		536		CALL	THUPD1
8807	78		537		LD	A, B
880A	FE84		538		CP	4
880C	C477B7		539		CALL	NZ, BFUPD1
880F	E1		540		POP	HL
8810	D1		541		POP	DE
8811	C1		542		POP	BC
8812	F1		543		POP	AF
8813	F8		544		EI	
8814	ED4D		545		RETI	
8816	2020414D		546	AM1	DEFM	' AM'
881A	0D		547		DEFB	0DH
881B	2020504D		548	PM1	DEFM	' PM'
881F	0D		549		DEFB	0DH
8820	C0B909		550	APMUP1	CALL	CLEAR
8823	3A03F8		551		LD	A, (APMFLC)
8826	C847		552		BIT	0, A
8828	2805		553		JR	Z, AMDEC1
882A	211808		554		LD	HL, PM1
882D	1803		555		JR	MIDWA1
882F	211608		556	AMDEC1	LD	HL, AM1
8832	CD7A88		557	MIDWA1	CALL	MSC1
8835	CD9903		558		CALL	DEC_SP
8838	C9		559		RET	
8839	11FEFF		560	SCAN3	LD	DE, 0FFFEH
883C	6A		561		LD	L, D
883D	2614		562		LD	H, 20
883F	D07E00		563	KCOL	LD	A, (IX)
8842	D390		564		OUT	(SEG1), A
8844	D023		565		INC	IX
8846	D07E00		566		LD	A, (IX)
8849	D391		567		OUT	(SEG2), A
884B	78		568		LD	A, E
884C	D380		569		OUT	(DIG1), A
884E	7A		570		LD	A, D
884F	D381		571		OUT	(DIG2), A
8851	7D		572		LD	A, L
8852	D382		573		OUT	(DIG3), A
8854	0650		574		LD	B, COLDEL
8856	10FE		575		DJNZ	S
8858	3EFF		576		LD	A, 0FFH
885A	D380		577		OUT	(DIG1), A
885C	D381		578		OUT	(DIG2), A
885E	D382		579		OUT	(DIG3), A
8860	D023		580		INC	IX

IOM_MPF_IP					ASM 5.9
LOC	OBJ CODE M	STMT	SOURCE	STATEMENT	
8862	A7	581		AND	A
8863	CB03	582		RLC	E
8865	3802	583		JR	C, RL1
8867	CB03	584		SET	0, E
8869	CB12	585	RL1	RL	D
8868	3802	586		JR	C, RL2
886D	CB02	587		SET	0, D
886F	CB15	588	RL2	RL	L
8871	25	589		DEC	H
8872	20C8	590		JR	NZ, KCOL
8874	11D8FF	591		LD	DE, -40
8877	DD19	592		ADD	IX, DE
8879	C9	593		RET	
887A	7E	594	MSC1	LD	A, (HL)
887B	23	595		INC	HL
887C	FE00	596		CP	0DH
887E	C8	597		RET	Z
887F	CD2409	598		CALL	CHRW1
8882	18F6	599		JR	MSG1
		600		END	

; GET ORIGINAL IX.

4. Program Description

- 1) Statements 109-119 set 256 as the CTC Prescaler's value, 0FFH as the TIME CONSTANT REGISTER's value, so the cycle for one CTC interrupt is $256 \times 255 = 65280$. That is to say, after 65280 clock pulses, the CTC will interrupt the CPU. Since the address of the interrupt service routine is B202H, the CPU will jump to address B202H (the program counter value) to execute the program when it receives the INT signal. This program uses CTC channel 0 as the timer.
- 2) CPU calls subroutine TMUPDT after jumping to the interrupt service routine (address B202H), and statements 128-139 will check whether the CPU will be interrupted 27 times. If yes, then $B \leftarrow B-1$, the number of seconds increased by 1; if no, the CPU will go back to the interrupt service routine.
- 3) Statements 140-154 will check if the number of seconds exceeds 60. If yes, then $B \leftarrow B-1$ again, and enter the calculation of minutes; if no, the CPU will continue the calculation of seconds. If the number of minutes exceeds 60, the value of B is reduced by 1 again, and enter the calculation of hours.
- 4) Statement 155-176 will check if the number of hours exceeds 12. If yes, the AM will be changed to PM or the PM will be changed into AM.
- 5) Statements 177-200 refresh the display. That is, if the number of seconds exceeds 60, the screen will display the renewed number of minutes (added by 1); if the number of minutes exceeds 60, the screen will display the renewed number of hours (added by 1); if the number of hours exceeds 12, the screen will change AM to PM or PM to AM.
- 6) Statements 201-218 will translate the values of seconds, minutes and hours which is stored in F800-F802 into the display format.

- 7) Statements 219-221 define the base number which is the upper limit number of seconds, minutes and hours.
- 8) Statements 222-239 is the interrupt service subroutine.
- 9) In order to reduce the deviation, you can design a program which will increase the number of seconds by 1 when 66 seconds has passed by, then the deviation will be decreased.
- 10) CTC can be designed for either Timer or Counter application such as the automation control circuit. Please refer to the Z80 HANDBOOK for detailed information.

CHAPTER 4

8251

APPLICATION

EXAMPLE

1. Introduction

- 1) This program is an experiment for serial data transfer, using 8251 in its Asynchronous Mode as an RS-232 compatible serial interface on MPF-IP, to connect to the RS-232 interface on CRT.

The CPU will first read the DIP switch, which is used to select the baud rate ranging from 50 to 9600, and then sent the data it has read to the time constant register of the CTC to determine the data transfer rate.

- 2) The CTC may be used either for timing or event counting. When it operates in the Timer Mode, it accepts system pulse ($3058 \text{ MHz}/2 = 1.79 \text{ MHz}$). When it works in the Counter Mode, CTC accepts and counts input pulses from CLK/TR2 ($1.79 \text{ MHz}/2 = 0.895 \text{ MHz}$).
- 3) The following is the CTC time constant calculation when it is in the Timer Mode.

$$M = \frac{\phi}{BR \times DF \times PR \times 2}$$

M = CTC time constant

ϕ = system clock pulse ($= 3.58 \text{ MHz}/2 = 1.79 \text{ MHz}$)

BR = baud rate

DF = 8251 Divider Factor (16 or 64, 16 is used in this program)

PR = CTC Prescaler (16 or 256, 16 is used in this program)

- 4) The following is the CTC time constant calculation when in the Counter Mode.

$$M = \frac{CLK}{BR \times DF \times 2}$$

M = CTC time constant
 CLK= 1.79MHz/2=0.895MHz
 BR = baud rate
 DF = 8251 divider factor

- 5) The table of time constant and their corresponding baud rates used in this program are as follows:

Time Constant		
Baud Rate	Timer Mode	Counter Mode
	DF=16, PR=16	DF=16
50	70	
75	47	
110	32	
150	23	
200	18	
300		93
600		47
1200		23
2400		12
4800		6
9600		3

- 6) The following is the table of DIP switch and baud rate.

Baud Rate	S4	S3	S2	S1	{ 0 = on 1 = off Timer Mode
50	0	0	0	0	
75	0	0	0	1	
110	0	0	1	0	
150	0	0	1	1	
200	0	1	0	0	Counter Mode
300	0	1	0	1	
600	0	1	1	0	
1200	0	1	1	1	
2400	1	0	0	0	
4800	1	0	0	1	
9600	1	0	1	0	

2. Operating Procedure.

- 1) Connect IOM-MPF-IP and CRT with a cable.
- 2) Use DIP switch to set the baud rate to the same as that of CRT's. (Please refer to the table above)
- 3) Key in <G>=B300 
- 4) The CRT terminal will display:

- 5) Then the MPF-IP screen will display "COMPLETE".
- 6) If you press , the screen will display an identical message.
- 7) If you press  key, the word on the MPF-IP screen will disappear and the CRT displays statements, and control will be transferred to the CRT monitor. At this moment, any key pressed on the CRT keyboard will be echoed on the CRT screen.

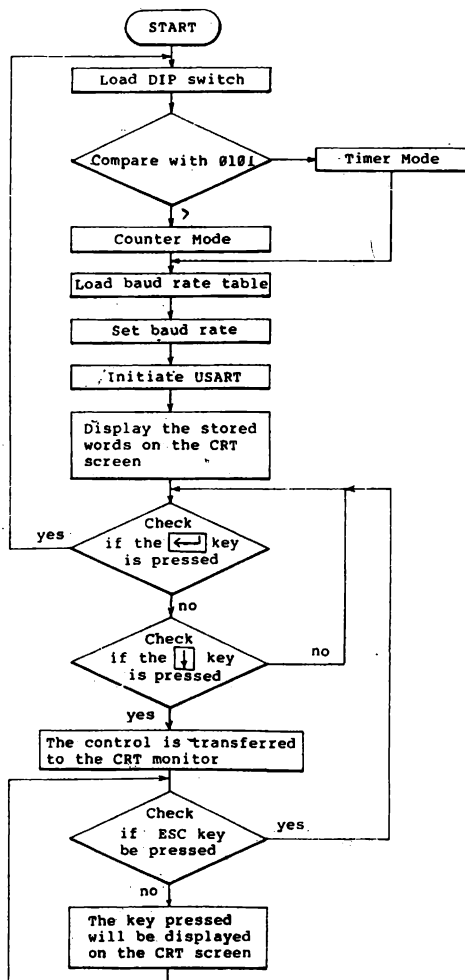
IOM-MPF-IP
 FEB. 13TH 1983 by Charles Chang
 IOM-MPF-IP is an I/O memory board.
 The address of ROM is from B000H to BFFFFH.
 The address of RAM is from D800H to EFFFFH.
 The I/O address of PIO is from 68H to 6FH.
 The I/O address of CTC is from 64H to 67H.
 The I/O address of 8251 is from 60H to 63H.
 It contains three demo. program.
 The first program uses PIO as a traffic light controller.
 The second program uses CTC to design a clock.
 The third program transfers data between IOM-MPF-IP and CRT through PS/2.

- 8) If you press the  key, the MPF-IP will regain control.
- 9) Press the  key to stop.

Note that it is necessary to run the application program for the 8251 with the IOM-MPF-IP connecting to a CRT. If the IOM-MPF-IP is not connected with a CRT, then nothing will happen after executing the program.

Please press the key on the CRT keyboard,
 and then it will display on the CRT screen.

3. Flow Chart



LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	ASM 5.9
			255		*****	
			256		*	
			257		*	
			258		8251 DEMO_PROGRAM	
			259		*	
			260		*****	
			261		THIS PROGRAM TRANSFERS DATA BETWEEN IOM_MPF_IP AND	
			262		CRT THROUGH RS_232 INTERFACE.	
			263		THE I/O ADDRESS OF 8251 IS FROM 60H TO 63H.	
8300			264		ORG 0B300H	
			265	URTD	EQU 60H	USART DATA
			266	URTCNT	EQU 61H	USART CONTROL
			267	CTC2	EQU 66H	BAUD RATE GENERATOR
			268	SPEED	EQU 6CH	BAUD RATE SWITCH
8300	CDB909		269	START3	CALL CLEAR	
8303	DB6C		270	IN	A, (SPEED)	READ BAUD RATE SWITCH
8305	E60F		271	AND	0FH	
8307	5F		272	LD	E, A	
8308	FE05		273	CP	0101B	ON-OFF-ON-OFF
830A	3E07		274	LD	A, 7	CHANNEL CONTROL WITH
			275			TIMMER MODE
830C	3802		276	JR	C, HIGSPD	
830E	3E47		277	LD	A, 47H	CHANNEL CONTROL WITH
			278			COUNTER MODE
8310	D366		279	HIGSPD	OUT (CTC2), A	
8312	217EB3		280	LD	HL, BDTAE	BAUD RATE TABLE
8315	1600		281	LD	D, 0	
8317	19		282	ADD	HL, DE	
8318	7E		283	LD	A, (HL)	TIMER(COUNTER) CONSTANT
8319	D366		284	OUT	(CTC2), A	
831B	2189B3		285	LD	HL, INIURT	INITIALIZE USART
831E	0606		286	LD	B, 6	
8320	4E		287	INIT	LD C, (HL)	PORT
8321	23		288	INC	HL	(HL)=DATA
8322	EDA3		289	OUTI		
8324	20FA		290	JR	NZ, INIT	
8326	21EAB3		291	LD	HL, TMSG	
8327	012002		292	LD	BC, ENDMSC-TMSG	
832C	7E		293	REPT	LD A, (HL)	
832D	CD73B3		294	CALL	CHRR	
8330	EDA1		295	CPI		
8332	EA2CB3		296	JP	PE, REPT	
8335	2109B6		297	LD	HL, COMPLE	
8338	CDCA09		298	CALL	MSG	
8339	CD9903		299	CALL	DEC_SP	
833E	DD212CFF		300	LD	IX, DISPBF	
8342	CD4602		301	REPT1	CALL SCAN	
8345	FE0D		302	CP	0DH	PRESS RETURN KEY?
8347	28B7		303	JR	Z, START3	
8349	FE69		304	CP	69H	PRESS DOWN ARROW KEY?
8348	2802		305	JR	Z, CRT	
834D	18F3		306	JR	REPT1	
834F	2195B3		307	CRT	LD HL, AMSC	
8352	015400		308	LD	BC, ZMSG-AMSC	
8355	7E		309	REP	LD A, (HL)	
8356	CD73B3		310	CALL	CHRR	
8359	EDA1		311	CPI		
835B	EA55B3		312	JP	PE, REP	

LOC	OBJ CODE	M	STMT	SOURCE	STATEMENT	ASM 5.9
835E	CD6AB3	313	REPT2	CALL	CHRRD	
8361	FE1B	314		CP	1BH	
8363	28DD	315		JR	Z, REPT1	:PRESS ESCAPE KEY?
8365	CD73B3	316		CALL	CHRRD	
8368	18F4	317		JR	REPT2	
836A	DB61	318	CHRRD	IN	A, (URCNT)	
836C	CB4F	319		BIT	1, A	
836E	28FA	320		JR	Z, CHRRD	
8370	DB60	321		IN	A, (URTD)	
8372	C9	322		RET		
8373	F5	323	CHRRD	PUSH	AF	
8374	DB61	324	WAIT1	IN	A, (URCNT)	
8376	CB4F	325		BIT	0, A	:CHECK BIT0 OF STATUS REGISTER
8378	28FA	326		JR	Z, WAIT1	
837A	F1	327		POP	AF	
837B	D360	328		OUT	(URTD), A	
837D	C9	329		RET		
837E	46	330	BDTAB	DEFB	70	:50 BAUD (TIMER MODE)
837F	2F	331		DEFB	47	:75 BAUD
8380	20	332		DEFB	32	:110 BAUD
8381	17	333		DEFB	23	:150 BAUD
8382	12	334		DEFB	18	:200 BAUD
8383	5D	335		DEFB	93	:300 BAUD (COUNTER MODE)
8384	2F	336		DEFB	47	:600 BAUD
8385	1F	337		DEFB	23	:1200 BAUD
8386	0C	338		DEFB	12	:2400 BAUD
8387	06	339		DEFB	6	:4800 BAUD
8388	03	340		DEFB	3	:9600 BAUD
8389	61	341	INIURT	DEFB	URCNT	
838A	00	342		DEFB	0	:3 NULL BYTES RESET USART
838B	61	343		DEFB	URCNT	
838C	00	344		DEFB	0	
838D	61	345		DEFB	URCNT	
838E	00	346		DEFB	0	
838F	61	347		DEFB	URCNT	
8390	40	348		DEFB	40H	
8391	61	349		DEFB	URCNT	
8392	8E	350		DEFB	BEN	:MODE BYTE
8393	61	351		DEFB	URCNT	
8394	37	352		DEFB	37H	:COMMAND BYTE
8395	00	353	AMSG	DEFB	0DH	
8396	506C6561	354		DEFB	'PLEASE PRESS THE KEY ON CRT KEYBOARD , AND'	
8398	00	355		DEFB	0DH	
83C1	7468656E	356		DEFB	'THEN IT WILL DISPLAY ON THE CRT SCREEN.	
83E8	00	357		DEFB	0DH	
83E9	00	358	ZMSG	DEFB	0DH	
83EA	00	359	TMSG	DEFB	0DH	
83EB	2A2A2A2A	360		DEFB	'***** IOM_MPF_IP *****'	
8411	00	361		DEFB	0DH	
8412	20202020	362		DEFB	'FEB. 13TH 1983 BY CHARLES CHANG'	
8436	00	363		DEFB	0DH	
8437	494F405F	364		DEFB	'IOM_MPF_IP IS AN I/O MEMORY BOARD.	
845A	00	365		DEFB	0DH	
845B	54686520	366		DEFB	'THE ADDRESS OF ROM IS FROM 8000H TO BFFFH.'	
8485	00	367		DEFB	0DH	
8486	54686520	368		DEFB	'THE ADDRESS OF RAM IS FROM D000H TO EFFFH.'	
8489	00	369		DEFB	0DH	
84B1	54686520	370		DEFB	'THE I/O ADDRESS OF PIO IS FROM 6BH TO 6FH.'	
84DB	00	371		DEFB	0DH	
84DC	54686520	372		DEFB	'THE I/O ADDRESS OF CTC IS FROM 64H TO 67H.'	

IOM_MPF_IP				ASH 5.9
LOC	OBJ CODE	M STMT	SOURCE STATEMENT	
B506	0D	373	DEFB	0DH
B507	546B6520	374	DEFM	'THE I/O ADDRESS OF 8251 IS FROM 60H TO 63H.
B532	0D	375	DEFB	0DH
B533	49742063	376	DEFM	'IT CONTAINS THREE DEMO. PROGRAMS.
B554	0D	377	DEFB	0DH
B555	546B6520	378	DEFM	'THE FIRST PROGRAM USES PIO AS A TRAFFIC
B57C	206C6967	379	DEFM	' LIGHT CONTROLLER.
B58E	0D	380	DEFB	0DH
B58F	546B6520	381	DEFM	'THE SECOND PROGRAM USES CTC TO DESIGN A
B5B6	20636C6F	382	DEFM	' CLOCK.
B5B0	0D	383	DEFB	0DH
B5BE	546B6520	384	DEFM	'THE THIRD PROGRAM TRANSFERS DATA BETWEEN
B5E6	20494F4D	385	DEFM	' IOM_MPF_IP AND CRT THROUGH RS_232.
B609	0D	386	DEFB	0DH
B60A	0D	387	ENDMSG DEFB	0DH
B60B	20202020	388	COMPLE DEFB	' COMPLETE'
B619	0D	389	DEFB	0DH

4. Program Description

- 1) Statements 269-284 reads in an 8-bit value from the DIP switch and set the baud rate according to that value.
- 2) Statements 285-289 initiate the USART.
- 3) Statements 290-301 write the stored statements into the CRT and display them on the screen.
- 4) Statement 302 checks if the  key is pressed. If yes, the stored statements will be displayed again.
- 5) Statements 304-313 check if the  key is pressed. If yes, the CRT monitor will gain control and display the pressed key.
- 6) Statements 314-317 check if ESC key is pressed. If yes, the MPF-IP will regain control.
- 7) Statements 318-329 cause MPF-IP to read data to or write data from the CRT.
- 8) Statements 330-340 contains the elements of the Baud Rate Table.
- 9) Statements 341-352 is the comment to initiate USART.
- 10) Statements 353-389 are all used for storing screen messages.