WDT-01

User Manual

ICP DAS

Industrial Computer Products
Data Acquisition System

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assume no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software **on a single machine.** The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Table of Contents

1. INTRODUCTION 4

1.1 GENERAL DESCRIPTION 4
1.2 BOARD LAYOUT 5
1.3 THE BLOCK DIAGRAM 6
1.4 QUICK START 1 : DIAGNOSTIC1 8
1.5 QUICK START 2 : DIAGNOSTIC2 11
1.6 QUICK START 3 : WATCHDOG IRQ 14
1.7 QUICK START 4 : WATCHDOG RESET 17
1.8 QUICK START 5 : EXTERNAL POWER 19
1.9 THE DIAGNOSTIC PROGRAM 22
1.10 FEATURES AND SPECIFICATIONS 24

2. HARDWARE SETTING 26

2.1 BOARD LAYOUT 26
2.2 THE I/O BASE ADDRESS (J8) 26
2.2.1 The I/O Ports 28
2.2.2 Command / Data port 29
2.2.3 The Status port 31
2.2.4 Clear IRQ port 32
2.3 THE IRQ SELECTION (J9) 32
2.4 THE MERGE SELECTION (J2) 33
2.5 IRQ/RESET SELECT (J5) 34
2.6 RESET SIGNAL (J3 AND J4) 35
2.7 POWER-GOOD SIGNAL AND TERMINAL BLOCK (TB1) 36
2.8 BAUD RATE SELECT (J6) 37
2.9 TEMPERATURE RANGE SELECT (J7) 38
2.10 EXTERNAL SPEAKER CONNECTOR (J1) 38
2.11 USE EXTERNAL POWER 40
2.12 USE THE RELAY 40
2.13 THE RS232 CONNECTOR 41
2.14 THE DB25 I/O CONNECTOR 43
2.15 WDT-01 TERMINAL BOARD 43
2.16 DEFAULT SETTING 45

3. THE DIAG.EXE SOURCE LISTING 46

1. Introduction

1.1 General Description

It's a fact that the PC hardware and software may be fail. To prevent the failure, many different solutions are proposed. However, none of these solutions offer a 100% assurance. Since it's hard to prevent the failure, to detect the failure is becomes more and more important. The WDT-01 is used to detect the failure of the software and hardware. It can be used to reduce the risk of PC failure. The WDT-01 is useful even for those systems with built-in watchdog circuit.

The WDT-01 is a watchdog card that provides watchdog timer, temperature monitor and power monitor functions. It also provides many signals and mechanism (IRQ, TTL signals, RELAY control, RESET signal, power-good signal…) that will allow user to control their system as soon as the errors are occurred. There are also many signals (IRQ, I/O status, RS232, LED, buzzer…) will tell the user or operator that some errors are occurred. Once the WDT-01 card has been power on, it will monitor the power and the temperature of PC automatically. After the software enable the watchdog timer, the WDT-01 will monitor the operations of software and hardware in the most inexpensive cost with the excellent protections.

The **DIAG.EXE**, given in the companion floppy disk, provides five **quickstart** for the usage of WDT-01. The completely source program also given in the same directory. **It is recommended to executed these five quickstart and read the source program for easy programming and understanding the WDT-01.**

1.2 Board Layout

The top view of WDT-01 is given in Fig 1.1 and it's terminal board is given in Fig 1.2. The details of jumpers setting and I/O connector are described in Chapter 2.

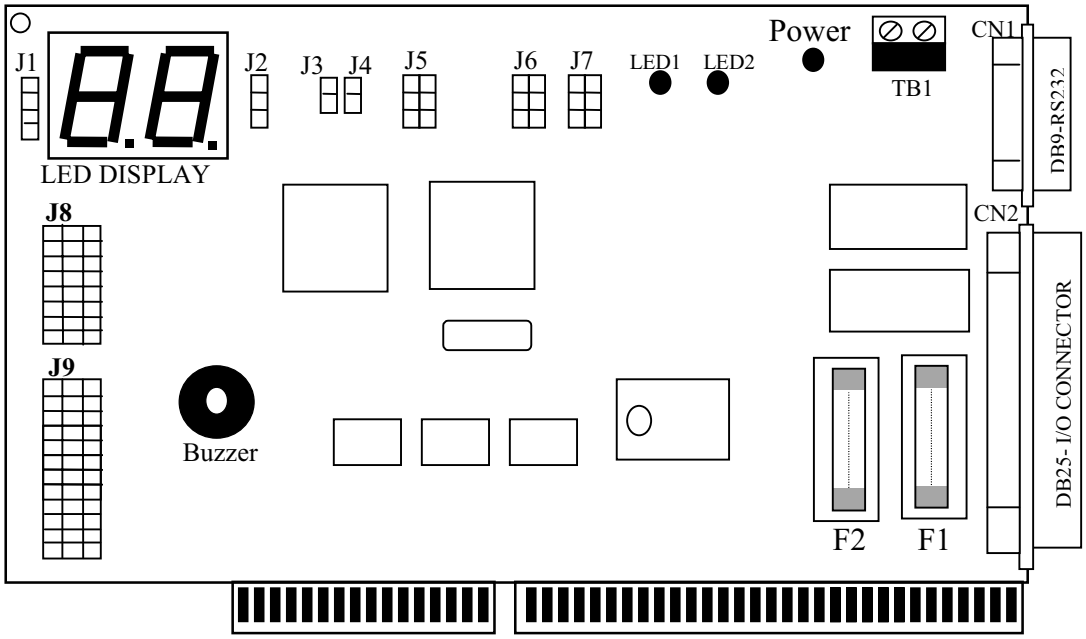


FIG 1.1: WDT-01 Top-View.

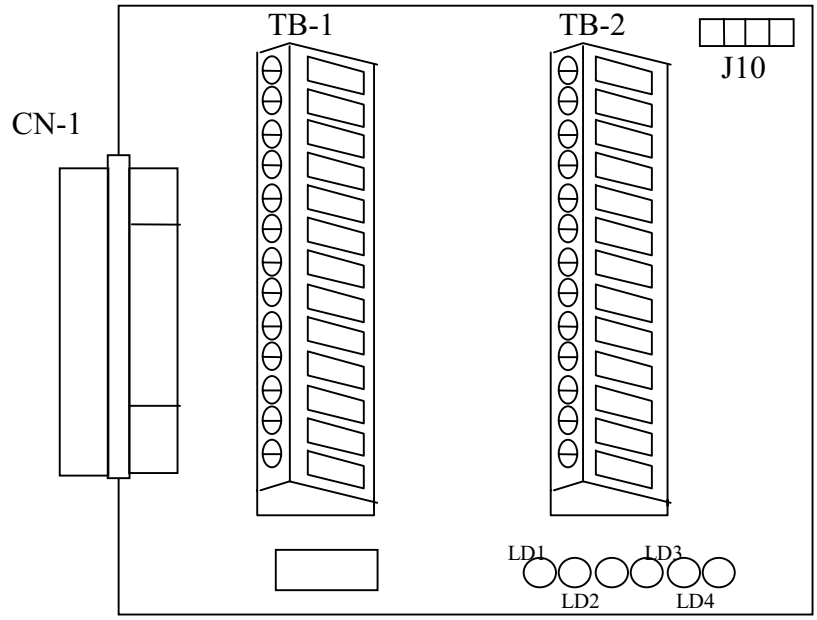


FIG 1.2: WDT-01 Terminal-board.

1.3 The Block Diagram

The block diagram of the WDT-01 is given in Fig 1-3.

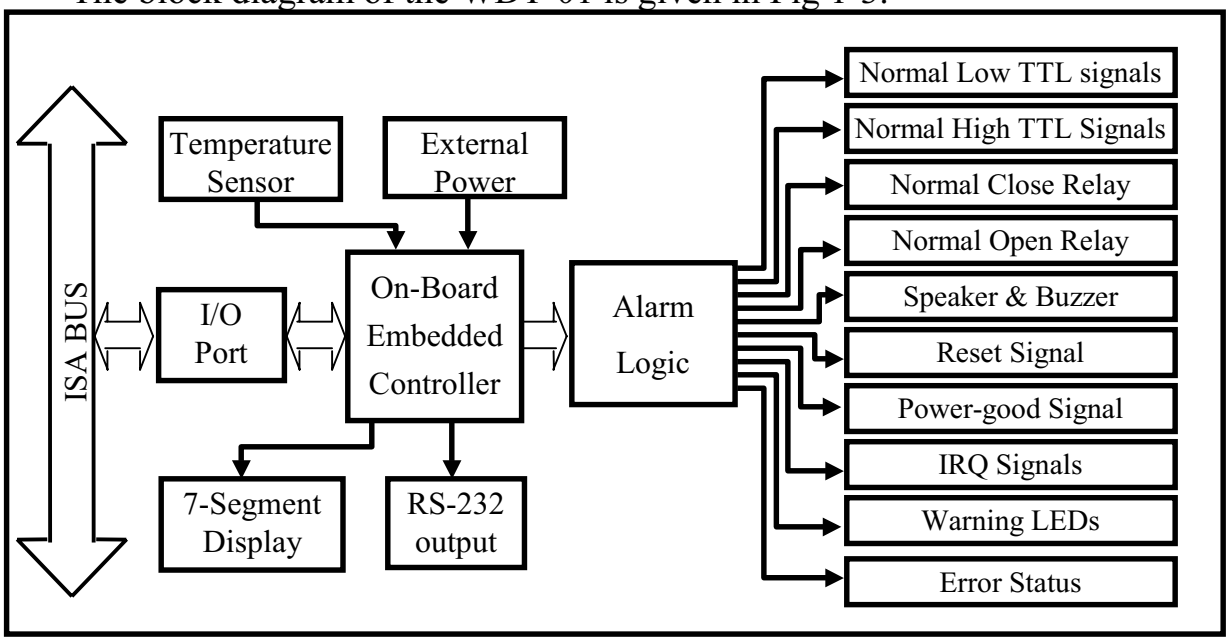


Fig 1-3: The block diagram of the WDT-01.

The WDT-01 equips with a on-board embedded controller to interpret and execute commands. The PC can send commands to WDT-01 through the I/O port. The command sets is described in Sec. 2.2.2. The I/O port address setting is given in Sec. 2.2.

The status of WDT-01 can be read back through the I/O port or RS232 output string. **The RS232 output string include all the status** of WDT-01, but **only the important status** is read back if read from I/O port. The status read from I/O port is given in Sec. 2.2.3. The RS232 output string format is provided in Sec. 2.13.

The 2 digits seven segment LEDs will show the temperature continuously. Also the temperature will compare with the hardware temperature-alarm-setting. If the temperature is higher or equal to the alarm setting, the buzzer, relay and TTL signal will active to predefined action. The temperature alarm setting is described in Sec. 2.9.

The built-in watchdog timer is disable when the PC is first power-on. The software can change the watchdog timer period and enable the watchdog timer. After the watchdog is enable, the software must reload the watchdog timer before the watchdog timer period is reached. If the hardware or software is fail, the watchdog timer will time-out. If the watchdog timer is time-out, the actions are given as following:

(1) buzzer and external speaker will beep
(2) the LED1 will on
(3) the status will toggle in both I/O port or RS232 output string
(4) maybe generate RESET pulse or IRQ signal to PC
(5) the relay will switch from normal_close to normal_open
(6) the TTL will toggle for both normal_low and normal_high signals

Refer to quickstart1/2/3/4 for details.

If the external +12V power is used, the WDT-01 will still function well even the PC power failure. The external power must be used with the watchdog timer enable. If the PC power failure and the external power is used, the LED, buzzer, relay, TTL are all still active as above. Refer to Sec. 1.8 for details.

1.4 Quick Start 1 : Diagnostic1

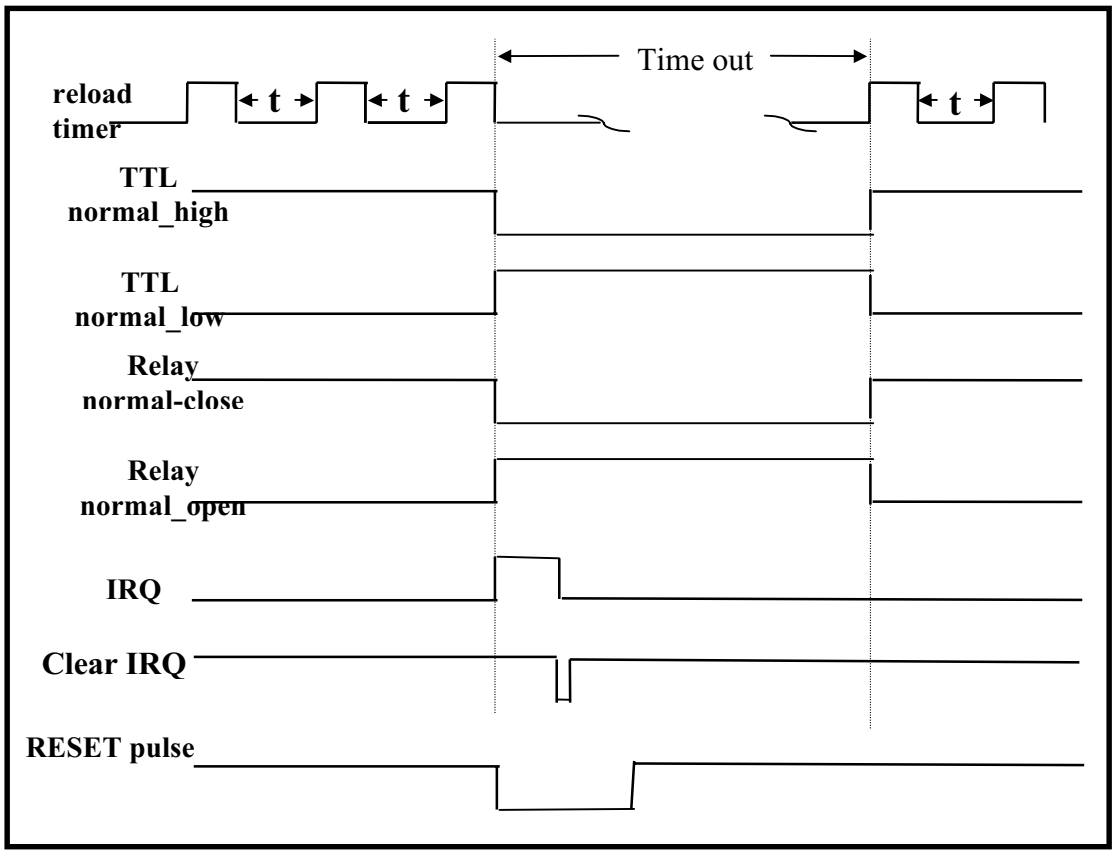
1. Power off the PC and install the WDT-01(no install any cable). **Make sure that no device use I/O address 0x278. If 0x278 is used, refer to Sec. 2.2 for jumper setting and Sec. 1.9 for DIAG.EXE configuration (use function 0).**
2. Power on the PC and execute the DIAG.EXE (given in the companion floppy disk)
3. Press **1** → to step 1. This program will show the status of the WDT-01 first. The normal status value of WDT-01 is 3, that is, watchdog timer OK, temperate OK, NO external power. Now the green power-on LED is ON and the red LED1 for watchdog timer and red LED2 for temperature are both OFF. The 2 digits seven-segment LEDs will show the current temperature continuously. The on-board buzzer is OFF and all relays and TTLs are all in their normal states. **Now the watchdog timer is 10 second and in disable state.**
4. Press any key → to step 2. This program will send a **T=1.0<CR>** command to the WDT-01. Now the LEDs, relays, buzzer and TTLs are all in the same state as in step 1. **This action only set the watchdog timer to 1.0 second but does not enable it.**
5. Press any key → to step 3. Now the program will **enable and reload continuously** the watchdog timer by **S<CR>** command. The user will see many **"."** in the screen, this means that the program is continuously reload the watchdog timer. **This action can be viewed as a simulation of the real word application which will perform it's normal function and reload the watchdog timer continuously.**
6. Press any key → to step 4. The program will **stop reload** the watchdog timer now. **This action can be viewed as the software or hardware is fail, so the program can not continue to reload the watchdog timer now**. The user must wait here for at least one second. **After one second, the watchdog timer will time-out and the red LED1 will be ON and the buzzer will beep. Also the relay and TTLs are active now**, that is to say, the relay will switch from nomal close to normal open and the TTL will toggle from high to low for normal high TTL and toggle from low to high for normal low TTL. **Because the IRQ and RESET signal are all disable in default setting, so no any other actions will be active** besides the LED1, buzzer, relay(for watchdog timer) and TTL(for watchdog for timer).
7. Press any key → to step 5. This program will **disable the watchdog timer** by **E<CR>** command. **Now all the active signals will return to their normal state**. So the LED1 will OFF, the buzzer will OFF, the relay and TTL will go to their normal states as in step 1
8. Press any key → to stop quickstart1 and return to the main program of DIAG.EXE.

The DIAG.EXE can be executed under DOS, Windows 3.1/95. Refer to Fig 1.4 for screen dump.The quickstart1 can be summary as following :

Step 1 : show the initial state of WDT-01
Step 2 : set the watchdog timer period=1.0 second
Step 3 : simulate the normal situation, reload the watchdog timer continuously and perform normal function
Step 4 : simulate the software or hardware failure, so the watchdog timer will time-out and show all actions
Step 5 : clear all actions and return to their normal states

Refer to Chapter3 for quickstart1 source program

The related signals diagram are given as following:



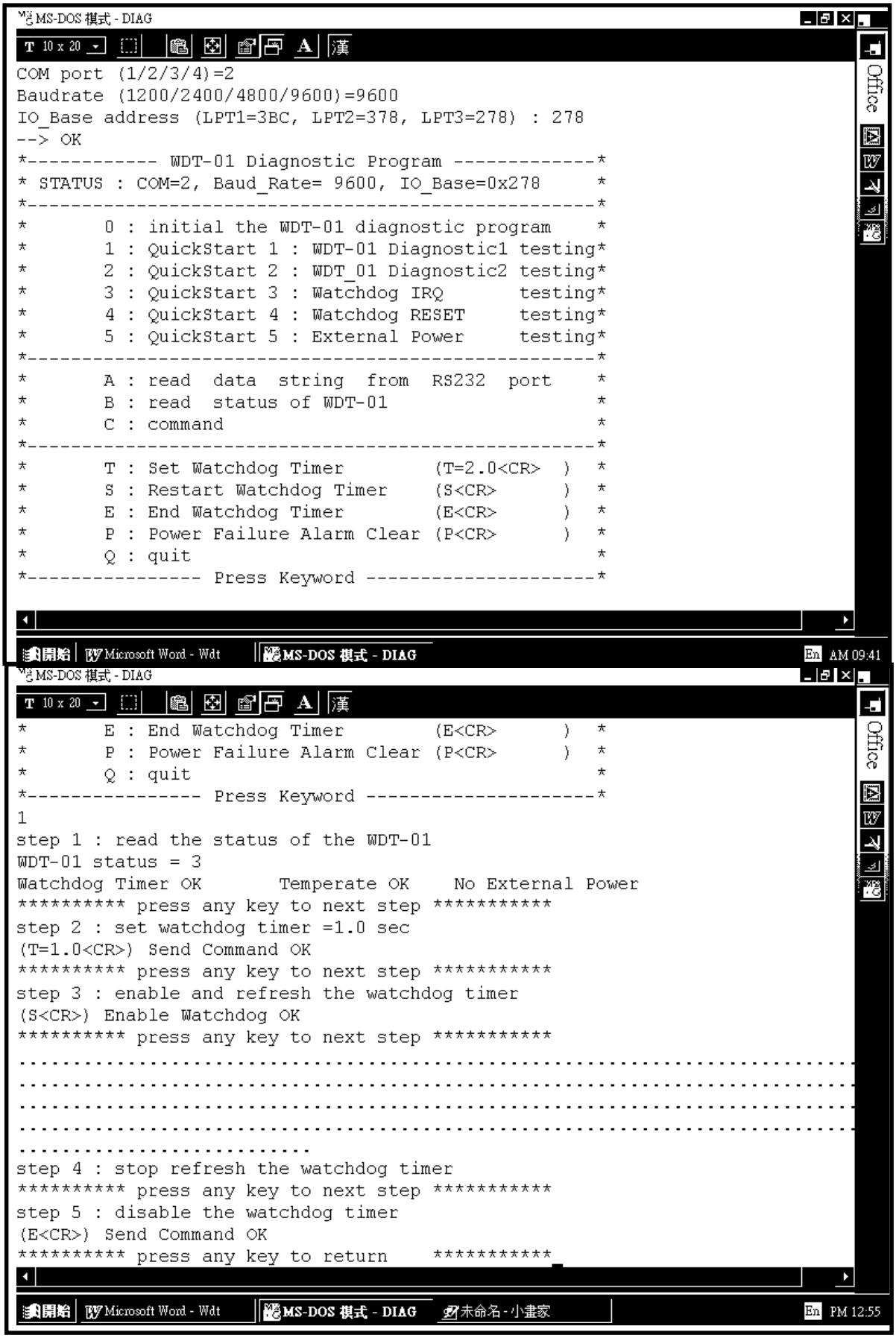


Fig 1.4 : screen dump of quickstart1 : diagnostic1 testing

1.5 Quick Start 2 : Diagnostic2

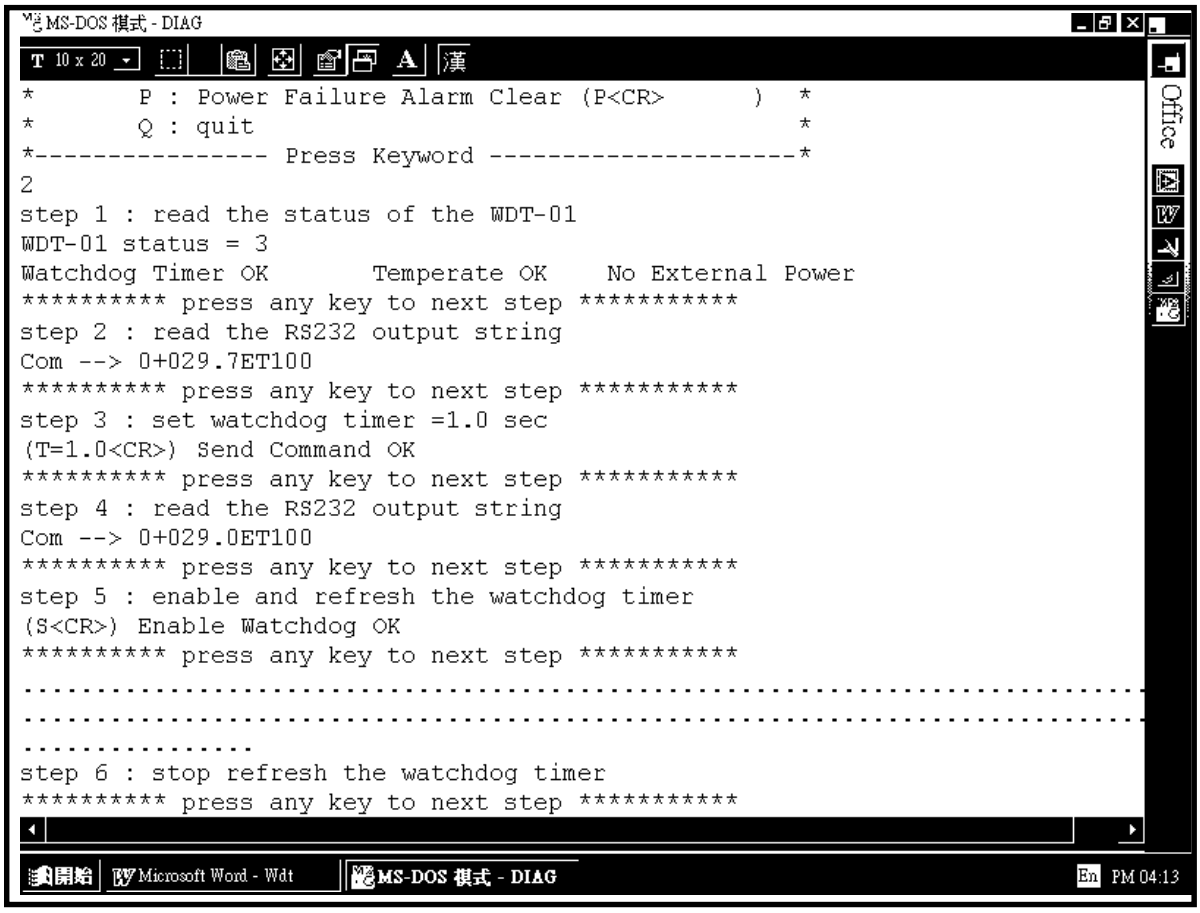
1. If you do not run the quickstart1 once, **please run once before start to quickstart2**. The quickstart2 is very similar to quickstart1 except the RS232 and the daughter board. We assume that the user is familiar with the quickstart1. **So the explanation is not so details as in quickstart1.**
2. Power off the PC and install the WDT-01.Also **install the companion RS232 cable** from the CN1 of WDT-01 to **COM2 of PC**. And **install the DB25 cable** from CN2 of WDT-01 to the WDT-01 daughter board. Make sure that there are only a LPT1 in the PC.
3. Power on the PC and execute the DIAG.EXE (given in the companion floppy disk)
4. Press **2** → to step 1. This program will show the status of the WDT-01 first as in quickstart1
5. Press any key → to step 2. This program will **read the RS232 output string** of WDT-01 and show the string in the screen. Refer to Sec. 2.13 for details.
6. Press any key → to step 3. This program will send a **T=1.0<CR>** command to **set the watchdog timer period** of the WDT-01 as in quickstart1.
7. Press any key → to step 4. This program will read the RS232 output string of WDT-01 and show the string in the screen to **make sure the watchdog timer period is correct**.
8. Press any key → to step 5. Now the program will **enable and reload continuously** the watchdog timer by **S<CR>** command as in quickstart1.
9. Press any key → to step 6. The program will **stop reload** the watchdog timer as in quickstart1.
10. Press any key → to step 7. This program will show the status of the WDT-01. Now the status of WDT-01 is 2, so **the watchdog timer is time-out** now.
11. Press any key → to step 8. This program will read the RS232 output string of WDT-01 and show the string in the screen to **make sure the state of watchdog timer**.
12. Press any key → to step 9. This program will **disable the watchdog timer** by **E<CR>** command as in quickstart1.
13. Press any key → to step 10. **Confirm** status read from the I/O port
14. Press any key → to step 11. **Confirm** status read from the RS232 output string.
15. Press any key → to stop quickstart2 and return to the main program of DIAG.EXE.

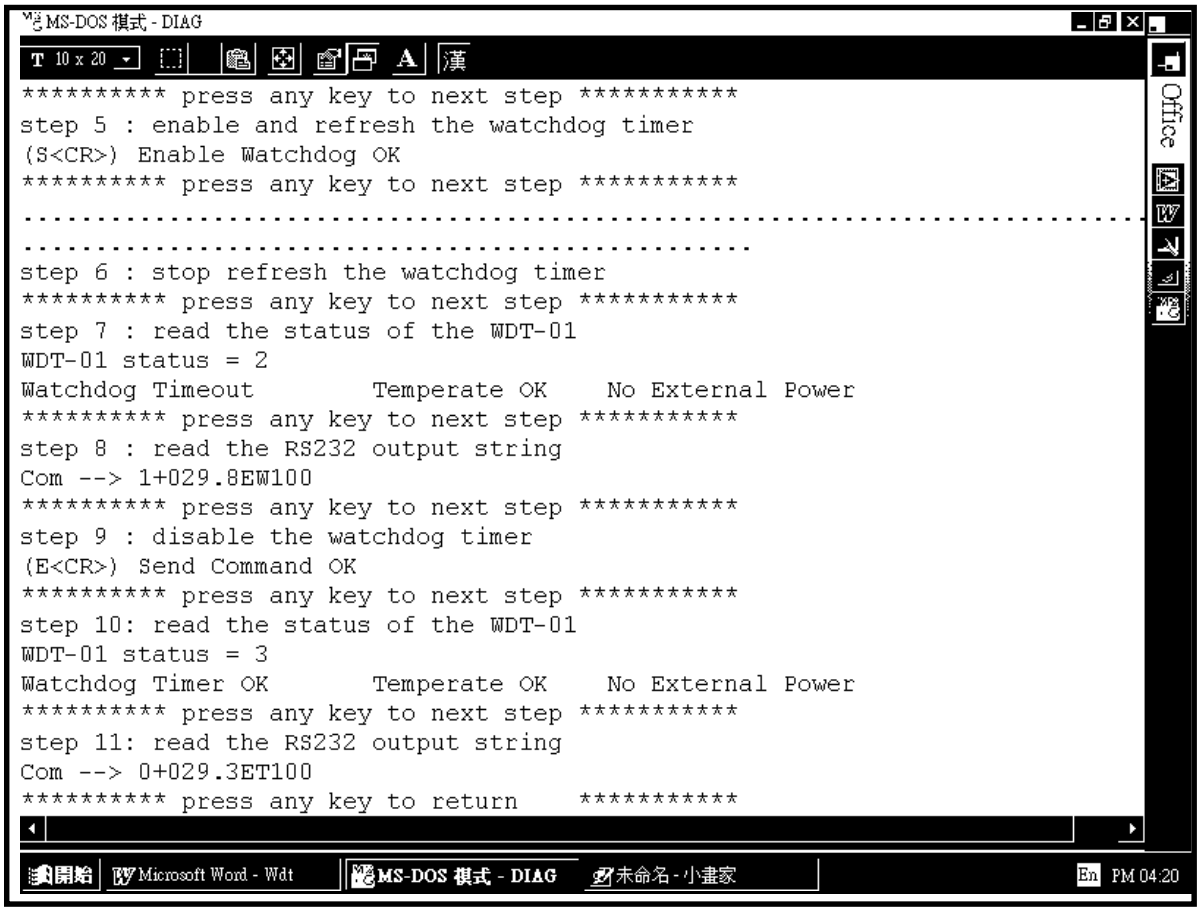
The diagnostic2 show that some information of the WDT-01 can be read back from the RS232 output ONLY. These information are given as following:

(a) **watchdog timer state, watchdog timer period and temperate state**

(b) **current temperate, current power state**

Refer to Sec. 2.13 for details.


MS-DOS 模式 - DIAG
* P : Power Failure Alarm Clear (P<CR>) *
* Q : quit *
---------- Press Keyword ----------
2
step 1 : read the status of the WDT-01
WDT-01 status = 3
Watchdog Timer OK Temperate OK No External Power
********** press any key to next step **********
step 2 : read the RS232 output string
Com --> 0+029.7ET100
********** press any key to next step **********
step 3 : set watchdog timer =1.0 sec
(T=1.0<CR>) Send Command OK
********** press any key to next step **********
step 4 : read the RS232 output string
Com --> 0+029.0ET100
********** press any key to next step **********
step 5 : enable and refresh the watchdog timer
(S<CR>) Enable Watchdog OK
********** press any key to next step **********
step 6 : stop refresh the watchdog timer
********** press any key to next step **********
開始
Microsoft Word - Wdt
MS-DOS 模式 - DIAG
PM 04:13


MS-DOS 模式 - DIAG
********** press any key to next step **********
step 5 : enable and refresh the watchdog timer
(S<CR>) Enable Watchdog OK
********** press any key to next step **********
step 6 : stop refresh the watchdog timer
********** press any key to next step **********
step 7 : read the status of the WDT-01
WDT-01 status = 2
Watchdog Timeout Temperate OK No External Power
********** press any key to next step **********
step 8 : read the RS232 output string
Com --> 1+029.8EW100
********** press any key to next step **********
step 9 : disable the watchdog timer
(E<CR>) Send Command OK
********** press any key to next step **********
step 10: read the status of the WDT-01
WDT-01 status = 3
Watchdog Timer OK Temperate OK No External Power
********** press any key to next step **********
step 11: read the RS232 output string
Com --> 0+029.3ET100
********** press any key to return **********
開始
Microsoft Word - Wdt
MS-DOS 模式 - DIAG
未命名 - 小畫家
PM 04:20

1.6 Quick Start 3 : Watchdog IRQ

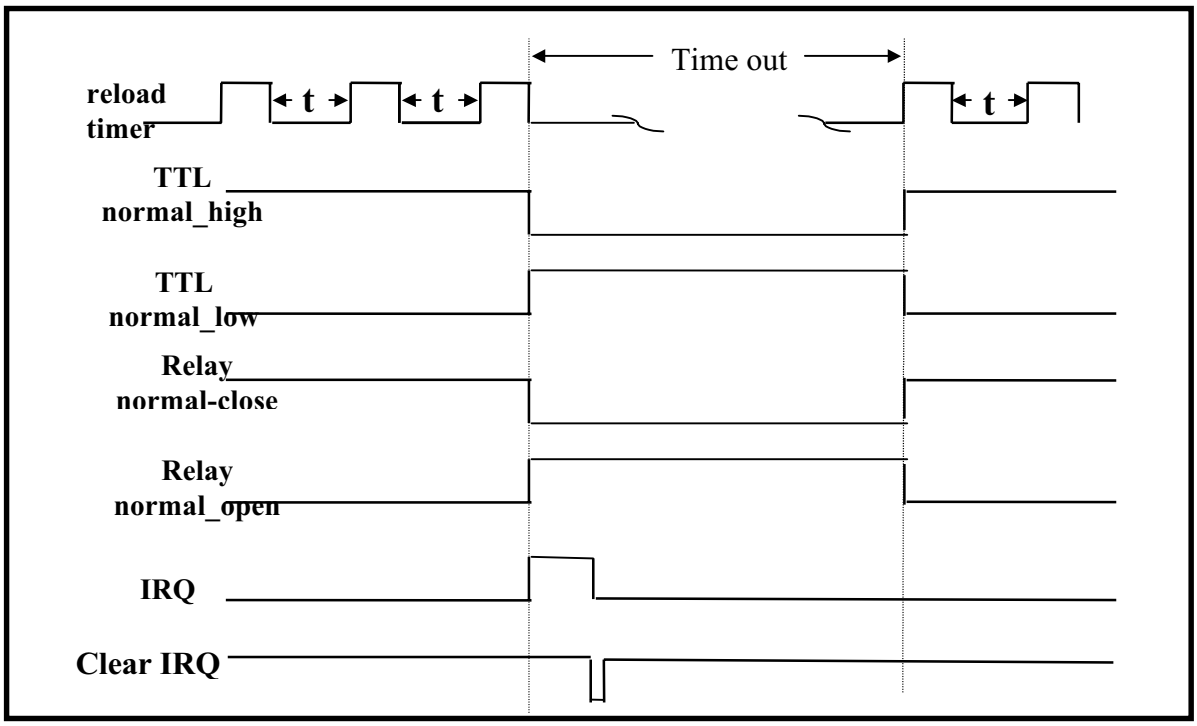
1. If you do not run the quickstart1 once, **please run once before start to quickstart3**. The quickstart3 is very similar to quickstart1 except the IRQ signal. We assume that the user is familiar with the quickstart1. **So the explanation is not so details as in quickstart1.**
2. Change the **J5 to select IRQ**, refer to Sec. 2.5 for details
3. Change the **J9 to select timer IRQ=5** , refer to Sec. 2.3 for details.
4. Power off the PC and install the WDT-01.
5. Power on the PC and execute the DIAG.EXE (given in the companion floppy disk)
6. Press **3** → to step 1. This program will show the status of the WDT-01 first as in quickstart1
7. Press any key → to step 2. This program will send a **T=1.0<CR>** command to **set the watchdog timer period** of the WDT-01 as in quickstart1.
8. Press any key → to step 3. Now the program will **enable and reload continuously** the watchdog timer by **S<CR>** command as in quickstart1.Also this program will show the **interrupt indicator uIrq=0(initial value)**.
9. Press any key → to step 4. The program will **stop reload** the watchdog timer as in quickstart1.
10. Press any key → to step 5. This program will show the **interrupt indicator Ireq=1** now. So the watchdog timer is **time-out and generate an hardware IRQ to PC**. **Then program transfer to the interrupt service routine.** The interrupt service routine set the uIrq=1 and **send the clear IRQ signal (refer to Sec. 2.2.4)** to WDT-01 then return to the main program.
11. Press any key → to stop quickstart3 and return to the main program of DIAG.EXE.

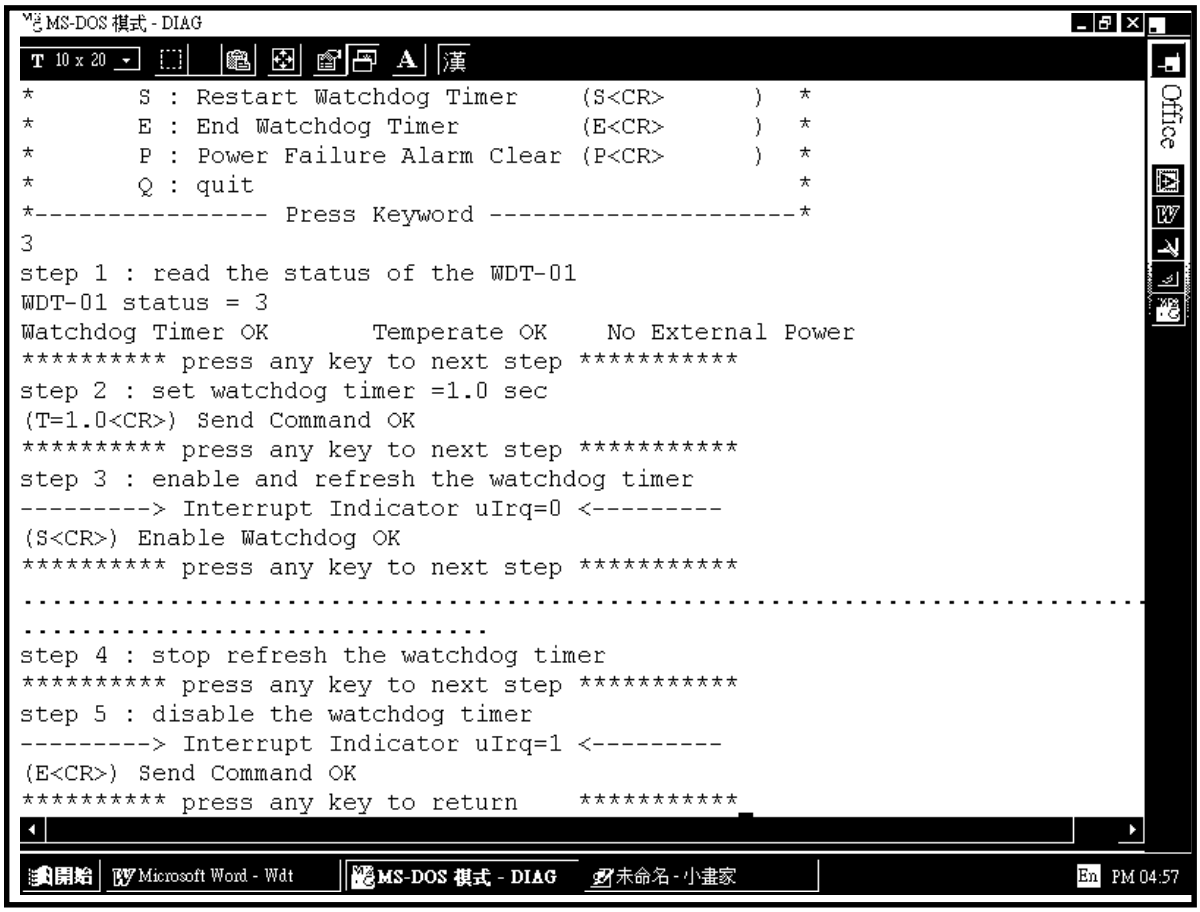
The key points of this quickstart are given as following:

1. Use J5 to **select IRQ signal**, the default setting is none.
2. Use J9 to **select IRQ channel number**, the default is setting is none
3. The hardware IRQ signal is **normal low**. When the watchdog timer is time-out, the hardware IRQ signal will **go to high and hold at high**. Then the control will transfer from the main routine to the interrupt service routine.
4. The IRQ service routine **must send out the clear-IRQ signal** to WDT_01 to enable next IRQ.(refer to Sec. 3.? For details) The hardware IRQ signal will **return to low after the clear-IRQ signal is sent**. So the next IRQ can be active

again. If the clear-IRQ signal is **absent**, the PC will receive **one IRQ only**.

The related signals diagram are given as following:





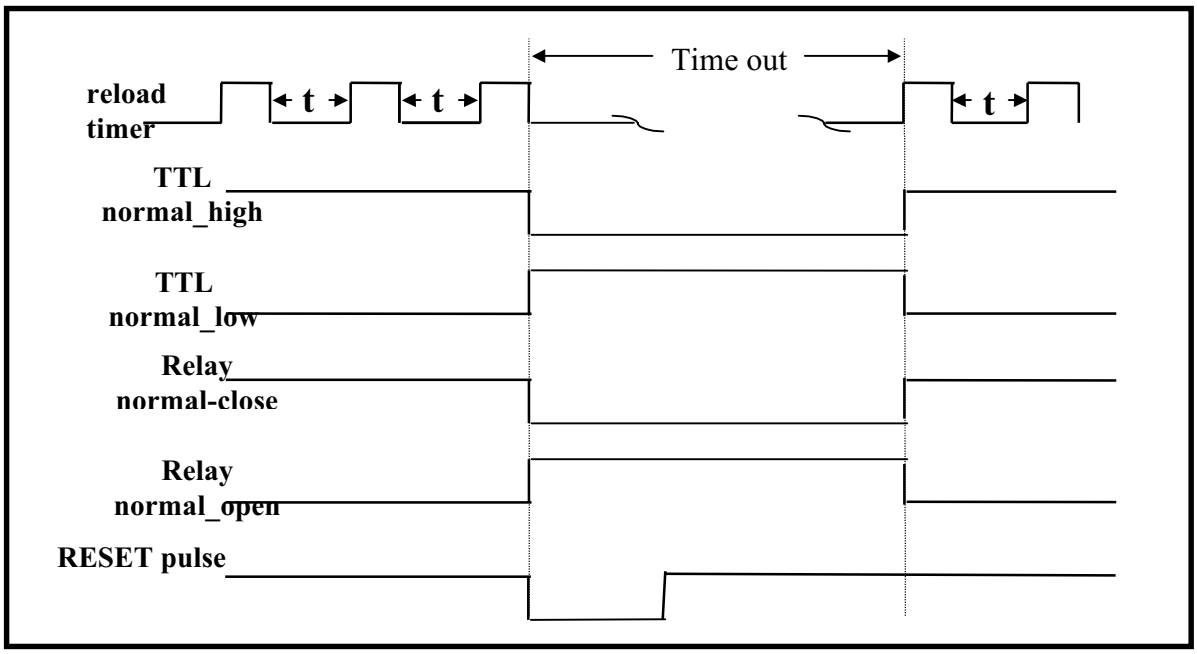
1.7 Quick Start 4 : Watchdog Reset

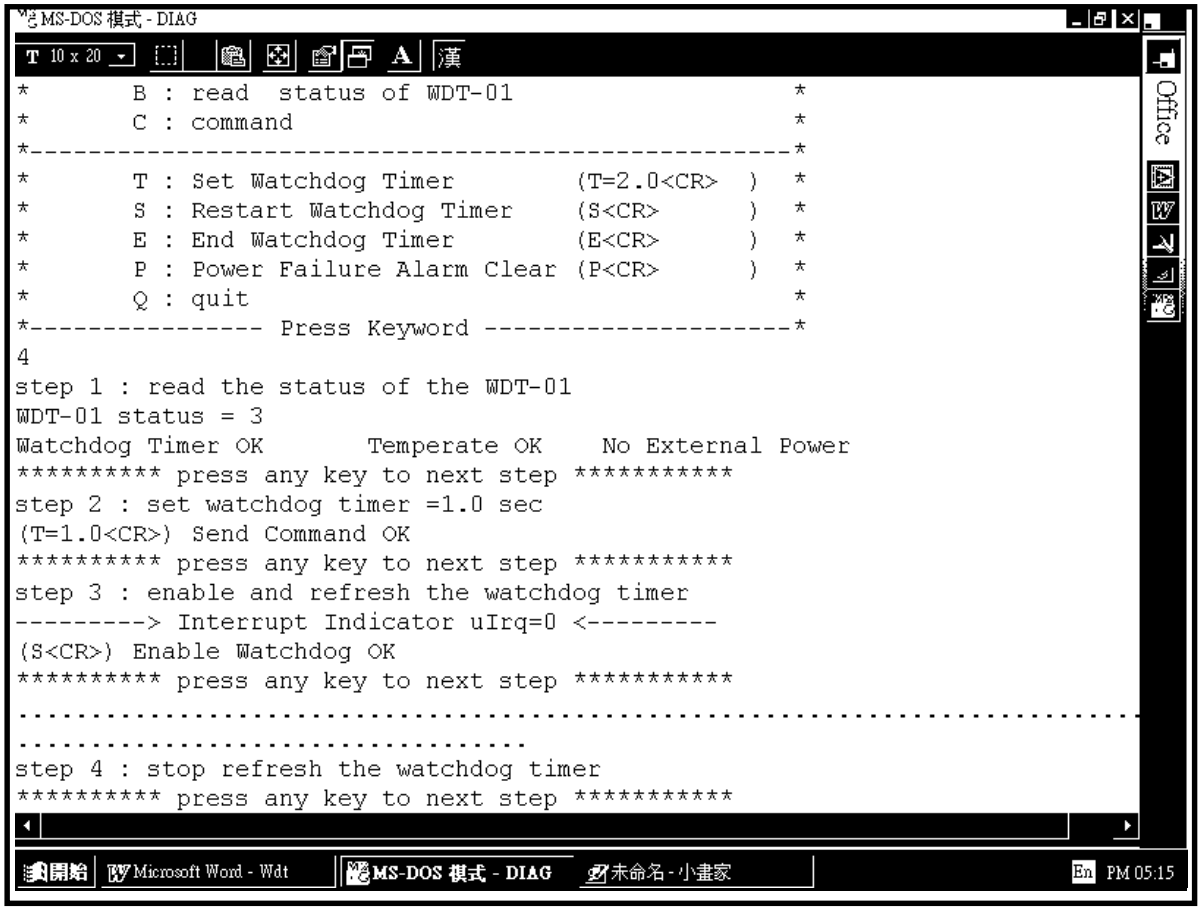
1. If you do not run the quickstart3 once, **please run once before start to quickstart4**. The quickstart4 is very similar to quickstart3 except the RESET signal. We assume that the user is familiar with the quickstart3. **So the explanation is not so details as in quickstart3.**
2. Change the **J5 to select RESET**, refer to Sec. 2.5 for details
3. **Connect the RESET signal** from WDT-01 to PC, refer to Sec. 2.6 for details.
4. Power off the PC and install the WDT-01.
5. Power on the PC and execute the DIAG.EXE (given in the companion floppy disk)
6. Press **4** → to step 1. This program will show the status of the WDT-01 first as in quickstart1
7. Press any key → to step 2. This program will send a **T=1.0<CR>** command to **set the watchdog timer period** of the WDT-01 as in quickstart1.
8. Press any key → to step 3. Now the program will **enable and reload continuously** the watchdog timer by **S<CR>** command as in quickstart1.
9. Press any key → to step 4. The program will **stop reload** the watchdog timer as in quickstart1 and **wait for RESET signal**. After one second, the WDT-01 will **generate a hardware RESET pulse** to reset PC. So the PC will re-boot. **During this interval the buzzer, LED1 and relay and TTL will be active as in quickstart1.** The user must execute the DIAG.EXE again and **press E** to **force all these actions return to their normal states**.

The key points of this quickstart are given as following:

1. Use J5 to **select IRQ signal**, the default setting is none.
2. **Connect the RESET signal from WDT-01 to PC**, the default is setting is none
3. The WDT-01 will **send a hardware reset pulse** to PC if **the watchdog timer is time-out** or **the reset-push-button in rack is pressed**.

The related signals diagram are given as following:





1.8 Quick Start 5 : External Power

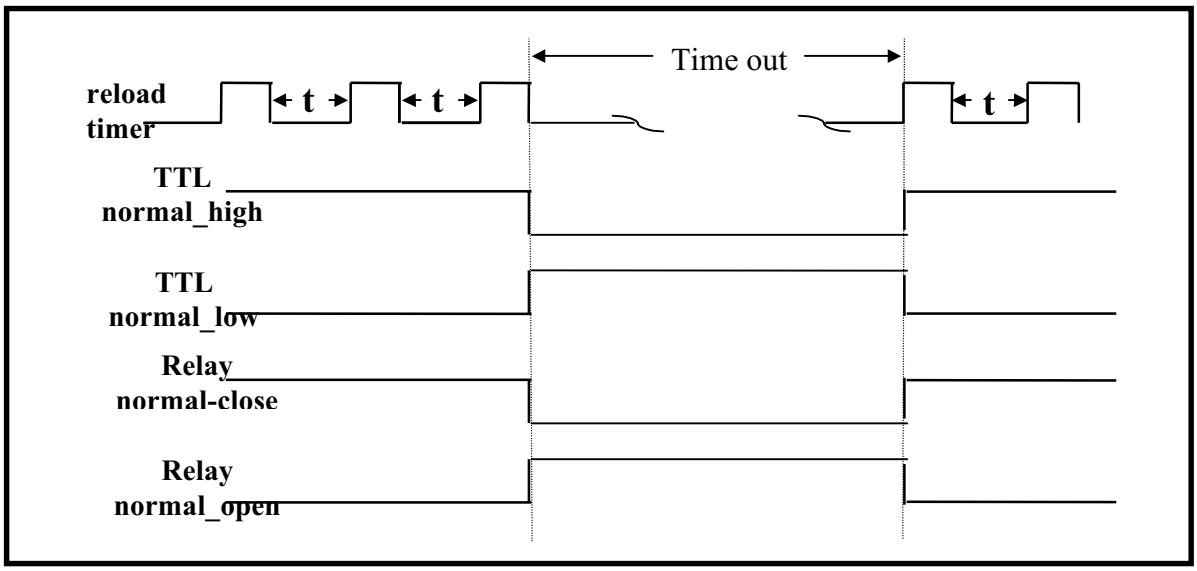
1. If you do not run the quickstart4 once, **please run once before start to quickstart5**. The quickstart5 is very similar to quickstart4 except the external power signal. We assume that the user is familiar with the quickstart4. **So the explanation is not so details as in quickstart4.**
2. Change the **J5 to select RESET**, refer to Sec. 2.5 for details
3. **Connect a +12V external power supply** to WDT-01 daughter board, refer to Sec. 2.11 for details.
4. Power off the PC and install the WDT-01.
5. Power on the PC and execute the DIAG.EXE (given in the companion floppy disk)
6. Press **4** → to step 1. This program will show the status of the WDT-01 first as in quickstart1
7. Press any key → to step 2. This program will send a **T=1.0<CR>** command to **set the watchdog timer period** of the WDT-01 as in quickstart1.
8. Press any key → to step 3. Now the program will **enable and reload** continuously the watchdog timer by **S<CR>** command as in quickstart1.
9. Power off the PC to **simulate hardware failure**. First **the buzzer will beep** for PC power failure warning. After one second, the **watchdog timer will time-out** (because the PC hardware failure). So **the LED1, relay and TTL will be active as in quickstart1**.
10. Power on the PC and execute the DIAG.EXE
11. Press P to **stop power failure alarm** → only buzzer OFF
12. Press E to **stop watchdog timer time-out actions** → the other actions OFF.

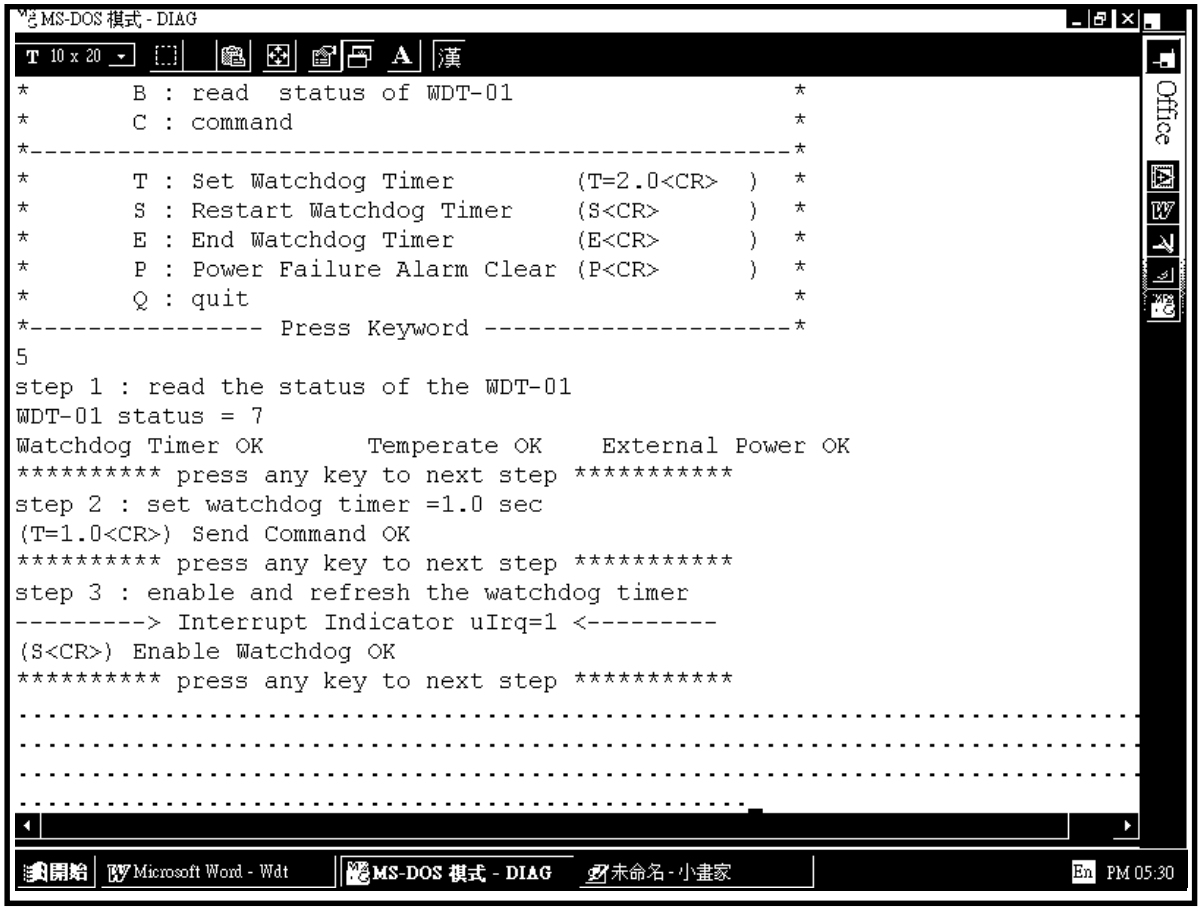
The key points of this quickstart are given as following:

1. Use J5 to **select RESET signal**, the default setting is none.
2. Connect a **+12V external power** to WDT-01
3. **Enable the watchdog timer**
4. **Power off the PC to simulate PC hardware power failure**
5. Show all actions

NOTE : External power must be used with watchdog timer enable

The related signals diagram are given as following:





1.9 The Diagnostic Program

The DIAG.EXE provides 14 functions given as following:

0 : program initial → set COM?, baudrate, LPT?
1 : for quickstart1
2 : for quickstart2
3 : for quickstart3
4 : for quickstart4
5 : for quickstart5
A : read RS232 output string from PC Com port 1/2/3/4
B : read status of WDT-01 form status I/O port
C : send command to WDT-01
T : set watchdog timer period=2.0 second
S : enable the watchdog timer
E : disable the watchdog timer
P : clear the power failure alarm
Q: quit the DIAG.EXE

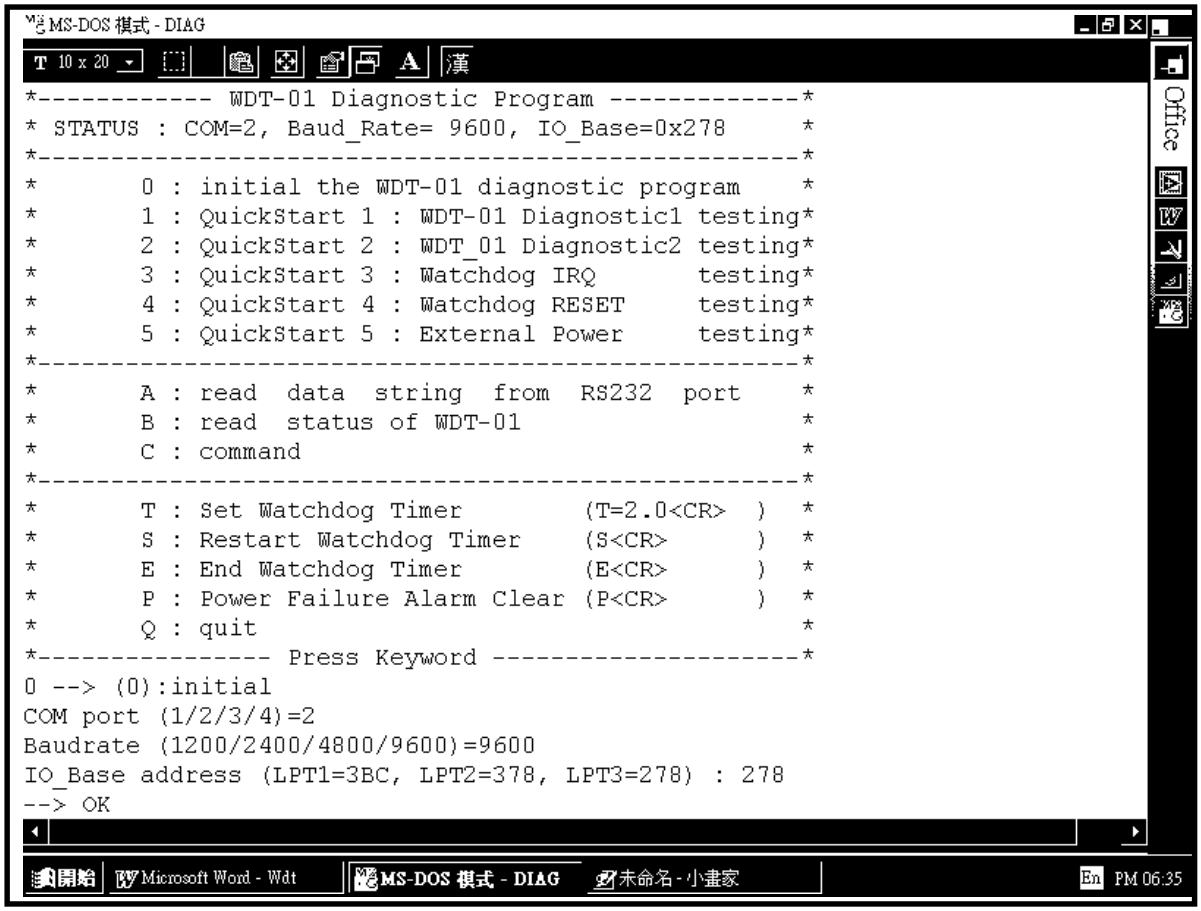
The function_0 can change the COM 1/2/3/4, baud rate 1200/2400/4800/9600 and the LPT 1/2/3. So the user must use this function to **setup the correct configuration.**

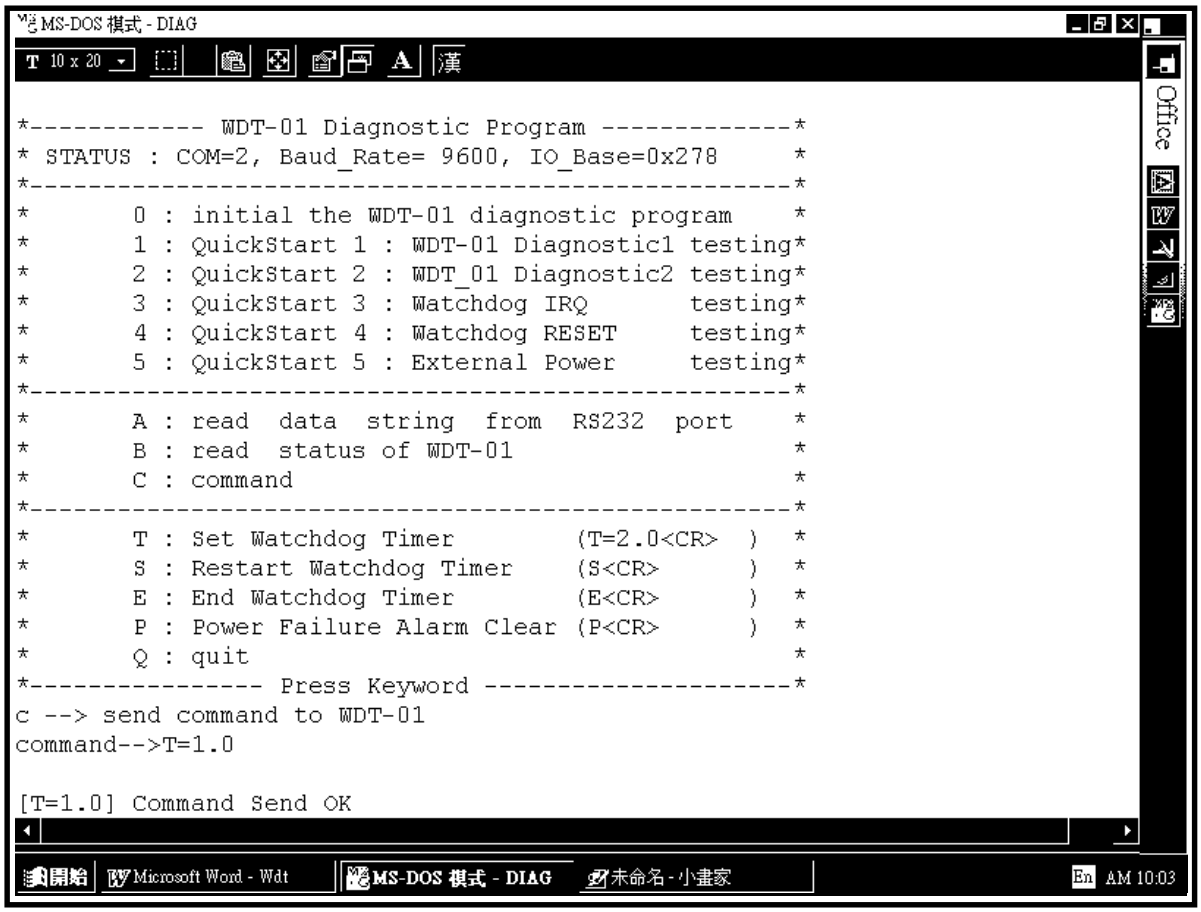
The function_1/2/3/4/5 are designed for quickstart1/2/3/4/5. **It is recommended to execute all the five quickstart to familiar with the WDT-01.** All the source program are given in the companion floppy disk. **It is recommended to scan all this codes quickly for easy start to program the WDT-01.**

The function_A/B/C/T/S/E/P are designed for easy testing and documentation. **It is recommended to scan all this codes in the companion floppy disk.**

The function_Q can stop the DIAG.EXE and return to the system.

The screen dump of **function_0** and **function_C** are given as below:


MS-DOS 模式 - DIAG
------------ WDT-01 Diagnostic Program -------------
* STATUS : COM=2, Baud_Rate= 9600, IO_Base=0x278   *
---
*   0 : initial the WDT-01 diagnostic program   *
*   1 : QuickStart 1 : WDT-01 Diagnostic1 testing*
*   2 : QuickStart 2 : WDT_01 Diagnostic2 testing*
*   3 : QuickStart 3 : Watchdog IRQ   testing*
*   4 : QuickStart 4 : Watchdog RESET   testing*
*   5 : QuickStart 5 : External Power   testing*
---
*   A : read data string from RS232 port   *
*   B : read status of WDT-01   *
*   C : command   *
---
*   T : Set Watchdog Timer   (T=2.0<CR>) *
*   S : Restart Watchdog Timer   (S<CR>   ) *
*   E : End Watchdog Timer   (E<CR>   ) *
*   P : Power Failure Alarm Clear (P<CR>   ) *
*   Q : quit   *
---------------- Press Keyword ----------------------
0 --> (0):initial
COM port (1/2/3/4)=2
Baudrate (1200/2400/4800/9600)=9600
IO_Base address (LPT1=3BC, LPT2=378, LPT3=278) : 278
--> OK
開始
Microsoft Word - Wdt
MS-DOS 模式 - DIAG
未命名 - 小畫家
PM 06:35


MS-DOS 模式 - DIAG
------------ WDT-01 Diagnostic Program -------------
* STATUS : COM=2, Baud_Rate= 9600, IO_Base=0x278   *
---
*   0 : initial the WDT-01 diagnostic program   *
*   1 : QuickStart 1 : WDT-01 Diagnostic1 testing*
*   2 : QuickStart 2 : WDT_01 Diagnostic2 testing*
*   3 : QuickStart 3 : Watchdog IRQ   testing*
*   4 : QuickStart 4 : Watchdog RESET   testing*
*   5 : QuickStart 5 : External Power   testing*
---
*   A : read data string from RS232 port   *
*   B : read status of WDT-01   *
*   C : command   *
---
*   T : Set Watchdog Timer   (T=2.0<CR>) *
*   S : Restart Watchdog Timer   (S<CR>   ) *
*   E : End Watchdog Timer   (E<CR>   ) *
*   P : Power Failure Alarm Clear (P<CR>   ) *
*   Q : quit   *
---------------- Press Keyword ----------------------
c --> send command to WDT-01
command-->T=1.0
[T=1.0] Command Send OK
開始
Microsoft Word - Wdt
MS-DOS 模式 - DIAG
未命名 - 小畫家
AM 10:03

1.10 Features and Specifications

- on board watchdog timer :
  1. Timer : 0.01second to 167772.15 seconds (software programmable).
  2. Relay output (normal-open , normal-close).
  3. TTL output (normal-high , normal-low).
  4. RS-232 output (status).
  5. IRQ output (jumper selectable).
  6. RESET output : reset signal and power-good signal
  7. Warning alarm (on-board buzzer, external speaker).
  8. On board LED warning signal.

- On board Temperature monitor :
  1. Operation temperature : 0°C ~ +75°C.
  2. Precision : ± 1.0°C.
  3. On board 7-segment display (2 digits).
  4. Relay output (normal-open , normal-close).
  5. TTL output (normal-high , normal-low).
  6. RS-232 output (temperature).
  7. Warning temperature range : 50°C /60°C /70°C /None (jumper selectable)
  8. IRQ output (jumper selectable).
  9. Warning alarm (on-board buzzer, external speaker).
  10. On board LED warning signal.

- On board power monitor :
  1. On board PC power monitor (+5V).
  2. On board external power monitor (+12V).
  3. Warning alarm (on board buzzer, external speaker).
  4. RS232 output (status).

- RS232 output :
  1. Selectable baud rate : 9600, 4800, 2400, 1200 (bits/second).
  2. Output :

  a. Watchdog timer state, temperature state.

  b. Current temperature.

  c. Power state .

  3. Data sending : 0.2 second per output.

- Power required (automatically select):

  1. External power : +12V ±5% / 3W.

  2. Internal power : +12V.

- External power protection :
  1. Over voltage protection.
  2. Over current protection.
  3. Reverse voltage protection.

- Least I/O ports used (only 3 I/O ports).

- Printer-like I/O interface.

2. Hardware Setting

2.1 Board Layout

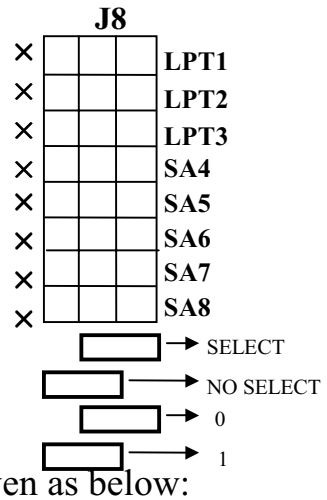
Refer to Sec. 1.2 for details.

2.2 The I/O Base Address (J8)

The WDT-01 card only uses three I/O locations of system I/O ports which provides a “printer-like” interface. When LPT[1-3] are selected the I/O ports are assigned to the system printer port and will be recognized as a printer drive. When LPT[1-3] are not selected, the I/O ports are determined by the given selected address SA[4-8].

NOTE : The WDT-01 only recognize I/O base address instead of LPT1/2/3. Select LPT1 means select I/O base address=0x3BC, select LPT3 means select I/O base address=0x278. The default setting of WDT-01 select LPT3, that is, the I/O base address is 0x278. **Because some BIOS can programming the I/O base address of LPT1/2/3. So the I/O address of LPT? may be changed by BIOS.**

	CMD/DATA PORT	STATUS PORT	CLEAR IRQ
LPT1	0x3BC	0x3BD	0x3BE
LPT2	0x378	0x379	0x37A
LPT3	0x278	0x279	0x27A
SA8	Base address	Base address+1	Base address+2
SA7			
SA6			
SA5			
SA4			



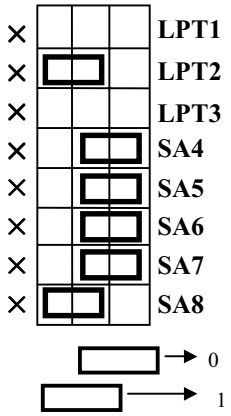
When the LPTn are not selected, the I/O base address is given as below:

SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
X	1	S	S	S	S	S	0	0	?	?

X : don't care. 1 : always 1.
0 : always 0. S : selected by jumper.
? : port address.

For example :

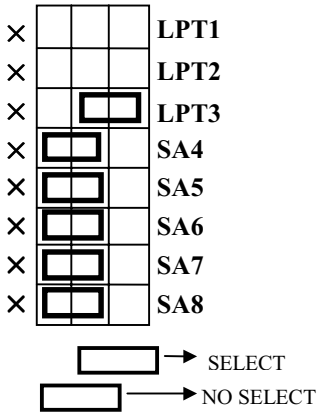
1. J8 is configured as shown right.
   The equivalent I/O base is :



SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1
x	1	1	0	0	0	0	0	0	0

The base address = 0x300.
CMD/DATA port = 0x300
STATUS port = 0x301
IRQ clear port = 0x302

2. J8 is configurable as shown right :
   The equivalent setup is LPT3.



CMD/DATA port = 0x278
STATUS port = 0x279
IRQ clear port = 0x27A

Note : When LPTn is selected, I/O base address selection SA8-4 will be ignored.

2.2.1 The I/O Ports

The WDT-01 use a " printer-like" I/O port. The same communication methods can be applied to this card. Command and data are pass through the **CMD/DATA port (base address + 0)** and the status can be obtained from **status port (base address + 1)**. To clear the IRQs, writes or reads the **"clear IRQ port" (base address + 2)**.

Address	Name	Descriptions
base address+0	CMD/DATA port	Command / Data port. Write : Command / data register. Read : Contains of CMD/DATA register.
base address+1	Status port	Status port. Read : Status word Write : Undefined.
base address+2	clear IRQ port	clear IRQ port Write : Clear watchdog timer IRQ. Read : Clear over temperature IRQ.

2.2.2 Command / Data port

The watchdog card equips with a built-in embedded controller to interpret and execute commands. All commands and data must pass through the command / data port, by the printer communication method (see section 2.2.3 for details). The command set is shown below:

Command	Description	Syntax
S	Start watchdog timer	➢ S <CR> Reload time-out period and start watchdog timer. For example: S<CR>
E	End watchdog timer	➢ E <CR> Stop watchdog timer. For example: E <CR>
P	Power failure buzzer stop	➢ P <CR> Stop the power failure buzzer. For example : P <CR>
T	Watchdog time-out period setting.	➢ T = nnnnnn.nn<CR> Set watchdog time-out period . For example : T=20 (timer = 20.00 sec) T=0.15 (timer = 0.15 sec).

Note :

1. Each command must be terminated by a <CR> character (ASCII = 0x0D) and **<u>can not be separated by blanks.</u>**
   - ➢ T = 1.0<CR> is illegal → separated by blanks.
   - ➢ T=1.0 <CR> is legal.

2. The power on watchdog timer state is :
   - Watchdog timer is **disable.**
   - Watchdog Time-out period = **10.00 sec.**

3. Whenever a watchdog timer is enable, the software must reload or disable the watchdog timer before time-out. The command **S<CR>** can reload the watchdog timer, and **E<CR>** can disable it. If the software fails to reload or disable the watchdog timer, the time-out error will occur.

4. Because the internal watchdog timer counter is 24 bit, the watchdog time-out period may vary from 0.01 sec to 167772.15 sec

5. The time-out period setting **T<CR>** command **only set the timer period** and will not take effect until a **S<CR>** command (reload and start the timer). To start a new setting, do as following :

   Step 1 : use **E<CR>** to disable watchdog timer
   Step 2 : use **T=??.??<CR>** to set the time-our period
   Step 3 : use **S<CR>** to enable the watchdog timer

   For example :
   - E<CR>
   - T=0.1<CR>
   - S<CR>

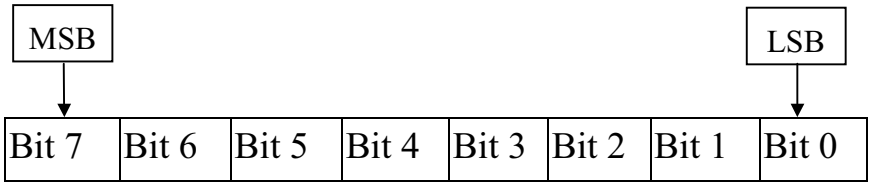
6. Both upper-case and lower-case are acceptable.

7. Illegal command will be ignored. Only the legal commands will be accepted and executed.

8. NOTE : Send a null command **<CR>** to the WDT-01 after first power-on.

2.2.3 The Status port

The format of status port is given as following :



Bit 7	Busy	0 : data buffer is full. 1 : data buffer is empty.
Bit 6	N/I	Not Implemented (read as 1)
Bit 5	N/I	Not Implemented (read as 0)
Bit 4	N/I	Not Implemented (read as 1)
Bit 3	N/I	Not Implemented (read as 1)
Bit 2	External Power	0 : no external power. 1 : external power.
Bit 1	Temperature Error	1 : normal temperature. 0 : abnormal temperature.
Bit 0	Time Out	1 : normal timer. 0 : watchdog timer out.

2.2.4 Clear IRQ port

Read or write to this port will clear hardware IRQ signal to enable next IRQ. If the software does not clear the IRQ signal after the first IRQ occurred, the next IRQ will not occur. So the interrupt service routine must clear the IRQ signal before return to the main routine.

Access	Descriptions
Write	clear the watchdog time-out IRQ.
Read	clear the over-temperature IRQ.

2.3 The IRQ Selection (J9)

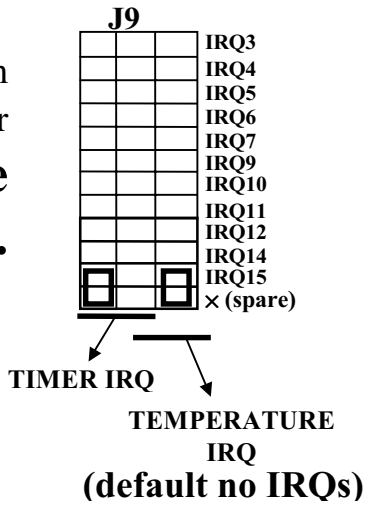
There are two IRQs : the **watchdog timer IRQ** and the **over-temperature IRQ**. Both can be selected in one jumper.

Note :

1. The IRQ number can be selected separately. User should check the IRQ number and the system IRQ to avoid conflict.

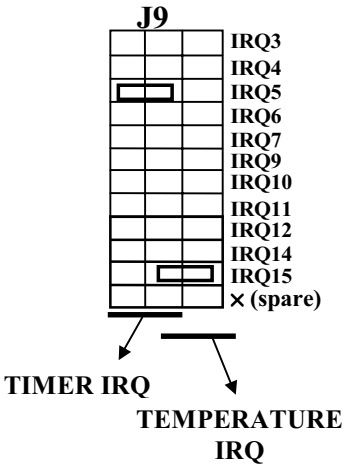
2. The spare pins are used to keep the unused plastic caps **(default setting is none, thus no IRQs are selected) .**

3. The IRQ is a normal_low_active_high signal with the initial state equal to low. Whenever the error occur, it will goes to high level and **will not be cleared until a CLEAR-IRQ signal.** Refer to Sec. 2.2.4 for clear-IRQ signal.



4. The jumper setup shown in the right is :

Timer IRQ = IRQ5
Temperature IRQ = IRQ15



2.4 The Merge Selection (J2)

The pin1 and pin2 of J2 can be connected to merge the two operations : the watchdog timer and the temperature monitor. After the operation is merged, both the **TTL output** and the **relay state** of the WDT-01 will be the same. The LED signals will be lightened after one of the error condition occurred.

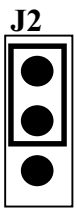
Note:

1. **Only LEDs, TTL signals and Relays are merged.**
2. IRQs signal no merged.

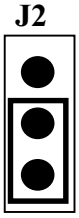
3. Status signal no merged (from RS232 or I/O port)

4. Reset signal no merged

5. Default setting is 2-3 → no merged



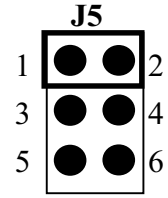
Merge



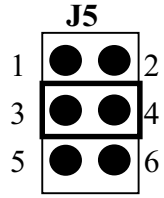
No merge
(default setting)

2.5 IRQ/RESET Select (J5)

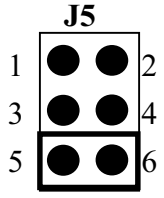
The watchdog time-out will generate the IRQ or reset signal The output type is jumper selectable. When pin1 and pin2 are connected, the TTL type IRQ signal is selected. When pin3 and pin4 are connected, the reset signal will be generated. Connect pin5 and pin6 will disable both signals.



Select IRQ



Select Reset



None
(default)

Note:

1. The reset signal will be held for about 0.05 second. Refer to Sec. ??.?? for details.

   When IRQ signal is selected, the application must send clear IRQ signal to enable next IRQ signal.

2.6 Reset Signal (J3 and J4)

When the watchdog timer is fail, there are two signals provided by the WDT-01 to reset the PC. One is **the reset signal** and another is **the power-good signal**. This section describe how to use the reset signal and the power-good signal is described in Sec. 2.7. **It is recommended to use the reset signal to reset the PC if needed.**

There is a reset push button in the PC rack. There is also a two lines wire connect from this button to the reset connector of main board or CPU card. The WDT-01 will first accept the two lines wire from push button, then connect to the reset connector of main board or CPU card. So the PC will be reset **if the WDT-01 reset signal is active** or **the push button is pressed.**

The install steps are shown as below:

step 1 : power off the PC and open the rack

step 2 : install the WDT-01

step 3 : unplug the old two lines wire(from push button to reset connector of main board or CPU card)

step 4 : install the old two lines wire into J3 (no direction)

step 5 : install the companion new two lines wire between the J4 and the reset connector in main board or CPU card.

step 6 : power on the PC

step 7 : if the PC can't work as normal, power off the PC and install the connection of step5 in reverse direction.

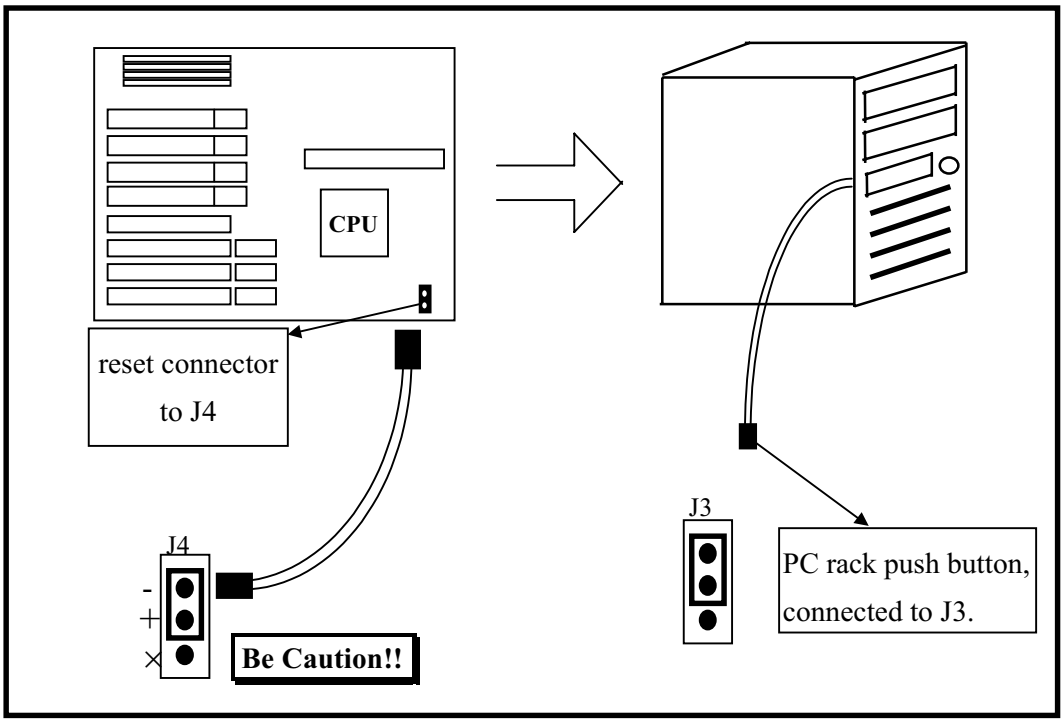


Fig 2-1 Reset signal installation.

2.7 Power-Good Signal and Terminal Block (TB1)

When the watchdog timer is fail, there are two signals provided by the WDT-01 to reset the PC. One is **the reset signal** and another is **the power-good signal**. This section describe how to use the power-good signal and the reset signal is described in Sec. 2.6. **It is recommended to use the reset signal to reset the PC if needed.**

The power-good signal and the terminal block, TB1, are designed to reset the PC. The user can connect one line from the "power-good" signal on the power supply to TB1. When the reset signal is active, the TB1 will signal the power supply to reset the PC. The TB1 will performs a reset pulse for about 0.04 second.

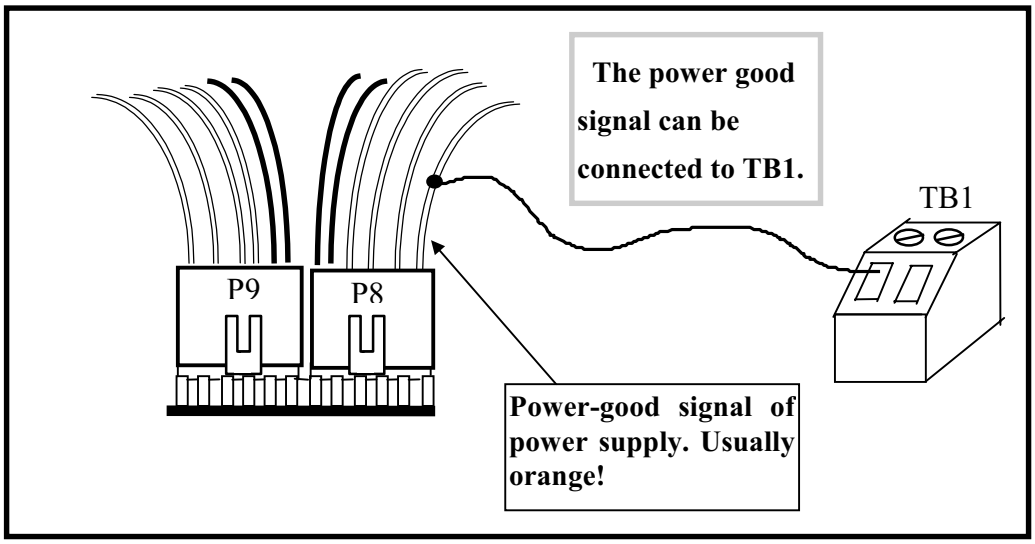


Fig 2-2 Power-good signal and TB1 connection.

Note :

1. Both outputs of TB1 can be connected to the power good signal of power supply.
2. The power good signal of power supply is usually in orange.

2.8 Baud Rate Select (J6)

The user can set J6 to select the baud rate of RS232 as following:

J6	J6	J6	J6	
9600	4800	2400	1200	(bits / second)
(default)				

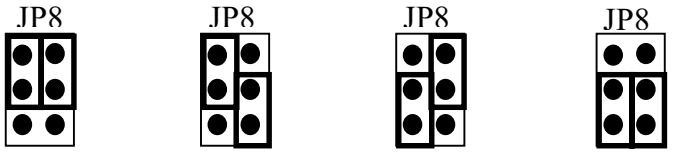
The data format of RS232 is automatically configured as following :

- 1 start bit.
- 8 data bit.
- 1 stop bit.
- no parity.

Refer to Sec. 2.11 for RS232 output string format.

2.9 Temperature Range Select (J7)

The user can use J7 to select the temperature range as following.



50°C 60°C 70°C **(default)** None.

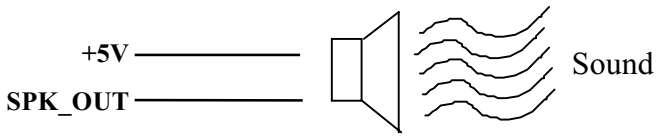
Note:

1. To disable the temperature monitor, please select the jumper to "None" .

2. If the temperature is higher than the selected temperature range, the over-temperature alarm will occur.

2.10 External Speaker Connector (J1)

Although there is an on-board buzzer that is used to indicate errors. WDT-01 also allows the external speaker to be used to indicate the alarm. The J1 is used to connected to the external speaker.

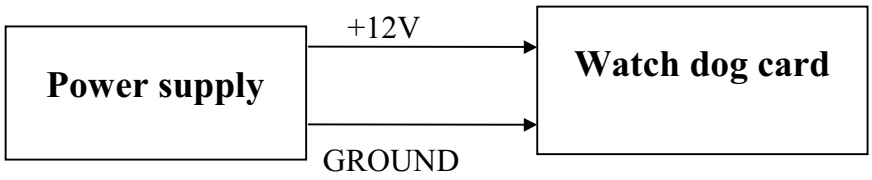


Note :Because of the limitation of power consumption, the external speaker should not exceed 0.5W. **The recommended**

speaker is 8Ω/0.5W.

2.11 Use External Power

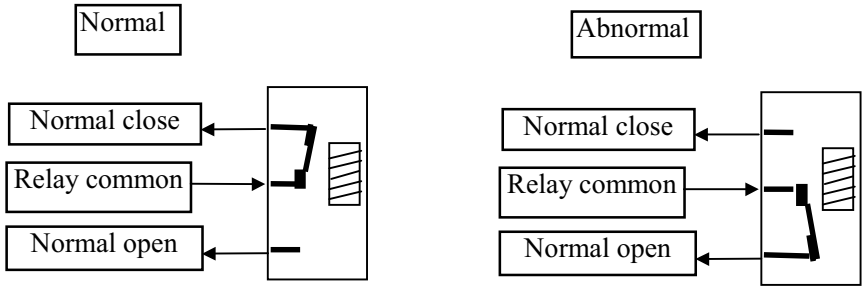
The WDT-01provides the mechanism that allows external power even after computer power is failure. In order to avoid any damage of the WDT-01 , the external power input must be +12V± 5% with the power consumption more than 3W. Higher voltage input will cause the zener diode to burnout and melt the fuse. Please remove the bad fuse and diode and install the new one after the error is fixed.



Device Number	Value
F1, F2	Fuse. 1 A.
ZD1	Zener Diode. 5V 1/2W. Used for external +12V
ZD2	Zener Diode. 12V 1/2W. Used for PC +12V

2.12 Use The Relay

The relay output of normal close and normal open signal is controlled by the watchdog timer and temperature monitor.

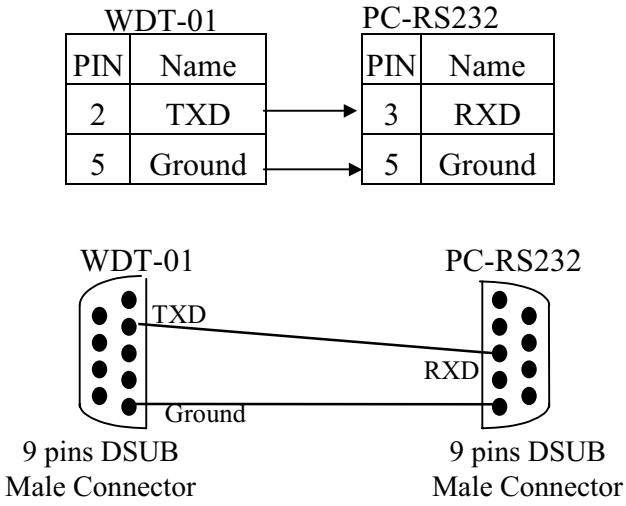


2.13 The RS232 connector

CN1 is the RS232 connector. The pin assignments is shown as below:

Pin	Name	Description
2	TXD (transmit data)	The output of RS232.
5	GROUND	signal ground
1,3,4,6 7,8,9	Not Connected	These pins are not connected internally.

The wire connection between PC RS232 connector and WDT-01 RS232 connector is shown below:



Note:

1. Pin 2 of WDT-01 should be connected to PC RS232 pin 3. Thus TXD(transmit data) of WDT-01 should be connected to PC RXD (receive data).
2. Both ground (WDT-01 pin5 and PC RS232 pin 5) should be connected together.
3. The WDT_01 RS232 output string format is given as following:

The RS232 will output a new string every 0.2 second. The output string includes the status of timer, temperature , power indicator and the timer settings.

BYTE	Name	Possible values and descriptions
1	Temperature and timer state	'0' : Normal. '1' : Watchdog time-out. '2' : Temperature error. '3' : Time-out and temperature error.
2-7	Current temperature.	Byte 2 : "+" or "–" Byte 6 : always "." Byte 3,4,5,7 : decimal digits. For example: –020.0 or +100.0 .
8	Current Power	"G" : Both external power and PC power are good. "P" : PC power fail. "E" : External power fail. "B" : Both PC power and external power are failed.
9	Watchdog Timer State	"T" : Watchdog timer is disabled. "W": Watchdog timer is enabled.
10..N	Watchdog Timer Period	The timer period setting is represent in a decimal string in 10mS. For example: 1000 ==> 1000 * 10ms = 10 s
N+1	End-of-String	Always carry-return (0x0D).

For example : RS232 output string : **0+045.7GW200<CR>**

This means : No watchdog time-out, NO over-temperature
Current temperature is +045.7°C
Both PC power and external power are good.
Watchdog timer enable.
Watchdog timer period is 2.00 second.

2.14 The DB25 I/O Connector

The pin assignments of DB25 I/O connector are listed below :

Pin	Name	Descriptions
1,14	NO_TIM	Normal-open watchdog timer relay.
2,15	NC_TIM	Normal-close watchdog timer relay.
3,16	TIM_COM	Watchdog timer relay common.
5,18	NO_TEM	Normal-open temperature relay.
6,19	NC_TEM	Normal-close temperature relay.
7,20	TEM_COM	Watchdog temperature relay common.
10	NH_TEM	Temperature normal-high TTL output.
22	NL_TEM	Temperature normal-low TTL output.
11	NL_TIM	Timer normal-low TTL output.
23	NH_TIM	Timer normal-high TTL output.
21	+5V	The on-board +5V for speaker.
9	SPK_OUT	The speaker signal output.
12,24	EX_GND	The external ground.
13,25	EX_12V	The external power supply (+12V).
4,8,17	Reserved	These pins are reserved for system used and will is not connected in the terminal board.

2.15 WDT-01 terminal board

The WDT-01 terminal board is used to connected to WDT-01 DB25 connector. The pin assignment is the same with WDT-01 DB-25 pin assignment. The LEDs in the terminal board indicate the current operation of the WDT-01.

Green LED : normal operation, indicates condition is normal .

Red LED : error indicator, indicates error occurred.

Orange LED : power indicator, an “on” means power.

2.16 Default Setting

The default setting of WDT-01 are given as following:

Jumper	Name	Default
J2	Merge Select	2-3 (no merge)
J3	to rack-reset-button	no install
J4	to PC-reset-connector	no install
J5	Reset/IRQ select	3-6 (no IRQ & no Reset)
J6	Baud-rate Select	1-2, 4-5 (9600 bps)
J7	Temperature range select	2-3, 4-5 (70°C)
J8	I/O port select	LPT3 (base = 0x278)
J9	IRQ number select	No IRQ

3. The DIAG.EXE Source Listing

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dos.h>
#include <io.h>

static void     interrupt irq5_service();
static void     interrupt    (*old_irq5)();

unsigned    uComPort,uBaseUart,uBaudRate,uBaseWdt,uIrq;

/* ---- main ------------------------------------------------------------ */

main()
{
char cChar;
```

```
disable();
old_irq5 = getvect(5+8);            /* save old irq5 vector */
setvect (5+8,irq5_service);
outportb (0x21,inportb(0x21) & 0xdf);      /* 8259a irq5 enable */
enable();
```

```
uComPort=2; uBaudRate=9600L; uBaseWdt=0x278;
open_com(uComPort,uBaudRate);     /* default */
```

```
/* important */
/* important */
/* important */
/* important */
/* important */

send_command("");    /* send a NULL command fisrt to WDT-01 */
```

```
/* important */
/* important */
/* important */
/* important */
/* important */

for(;;)
{
printf("\n*------------ WDT-01 Diagnostic Program -------------*");
show_status();
printf("\n*            0 : initial the WDT-01 diagnostic program       *");
printf("\n*            1 : QuickStart 1 : WDT-01 Diagnostic1 testing*");
printf("\n*            2 : QuickStart 2 : WDT_01 Diagnostic2 testing*");
printf("\n*            3 : QuickStart 3 : Watchdog IRQ            testing*");
printf("\n*            4 : QuickStart 4 : Watchdog RESET         testing*");
printf("\n*            5 : QuickStart 5 : External Power         testing*");
printf("\n*-------------------------------------------------*");
printf("\n*            A : read  data  string  from  RS232  port      *");
printf("\n*            B : read  status of WDT-01                      *");
printf("\n*            C : command                                     *");
printf("\n*-------------------------------------------------*");
printf("\n*            T : Set Watchdog Timer              (T=2.0<CR>    )    *");
printf("\n*            S : Restart Watchdog Timer       (S<CR>          )    *");
printf("\n*            E : End Watchdog Timer              (E<CR>          )    *");
printf("\n*            P : Power Failure Alarm Clear (P<CR>          )    *");
printf("\n*            Q : quit                                        *");
printf("\n*---------------- Press Keyword ----------------------*");
printf("\n");
cChar=getche();
switch (cChar)
      {
      case '0': init(); break;
      case '1': quick1(); break;
      case '2': quick2(); break;
      case '3': quick3(); break;
      case '4': quick4(); break;
      case '5': quick5(); break;
      case 'A':
```

```
        case 'a': read_from_com_port(); break;
        case 'B':
        case 'b': WDT_status_read(); break;
        case 'c':
        case 'C': WDT_command(); break;
        case 'T':
        case 't': Set_Watchdog_Timer(); break;
        case 's':
        case 'S': Start_Watchdog_Timer(); break;
        case 'e':
        case 'E': End_Watchdog_Timer(); break;
        case 'p':
        case 'P': Power_Failure_Alarm_Clear(); break;
        case 'q':
        case 'Q': goto ret_label;
        default : printf(" --> Error Keyword"); break;
        }

}
ret_label:
printf("\n*--------- WDT-01 Diagnostic Program stop -----------*");
setvect (5+8,old_irq5);
outportb (0x21,inportb(0x21) & 0x20);     /* 8259a irq5 disable */
}

/* ---------------------------------------------------------------------- */

void    interrupt irq5_service()
{
disable();
uIrq=1;          /* set the interrupt indicator */
outportb(0x20,0x20);    /* send EOI to 8259 */
outportb(uBaseWdt+2,0); /* clear IRQ for watchdog timer */
enable();
}

/* ---- show status ------------------------------------------------------ */
```

```
show_status()
{
printf("\n* STATUS : COM=%d,",uComPort);
printf(" Baud_Rate=%5d, IO_Base=0x%3x          *",uBaudRate,uBaseWdt);
printf("\n*-----------------------------------------------------*");
}

/* ---- send command to WDT-01 ------------------------------------------ */

send_command(char command[])
{
int i,t;

i=0; t=0;
while (command[i])
          {
          outportb(uBaseWdt,command[i++]);
          while ((inportb(uBaseWdt+1)&0x80)==0)
             {
             t++;
             if (t>32760) return(1);  /*    error */
             }
          }
t=0;
          outportb(uBaseWdt,0x0d);            /* send out the 0x0D to WDT-01 */
          while ((inportb(uBaseWdt+1)&0x80)==0)
             {
             t++;
             if (t>32760) return(1);  /*    error */
             }
return(0);
}
/* ---- open_com    ------------------------------------------------------ */
```

```
open_com(unsigned uPort, unsigned uBaudRate)
{
unsigned uVal,uCom;

switch(uPort)
      {
      case 1 : uBaseUart=0x3f8; uCom=0; break;
      case 2 : uBaseUart=0x2f8; uCom=1; break;
      case 3 : uBaseUart=0x3e8; uCom=2; break;
      case 4 : uBaseUart=0x2e8; uCom=3; break;
      default: return 1;          /* port must 1/2/3/4 */
      }

switch(uBaudRate)
      {
      case 1200 : uVal=0x83; break;
      case 2400 : uVal=0xA3; break;
      case 4800 : uVal=0xC3; break;
      case 9600 : uVal=0xE3; break;
      default     : return 2;               /* baud rate error */
      }

bioscom(0,uVal,uCom);
return(0);
}
```

```
/* ---- function 0 ------------------------------------------------*/

init()
{
unsigned iRet,iPort,i1,i2,i3;

printf(" --> (0):initial\n");
printf("COM port (1/2/3/4)="); scanf("%d",&i1);
printf("Baudrate (1200/2400/4800/9600)="); scanf("%d",&i2);
printf("IO_Base address (LPT1=3BC, LPT2=378, LPT3=278) : "); scanf("%x",&i3);
uBaseWdt=i3;
iRet=open_com(i1,i2);
```

```
if (iRet==0)
      {
      printf("--> OK");
      uComPort=i1; uBaudRate=i2;
      }
else if (iRet==1) printf("--> port error");
else if (iRet==2) printf("--> baudrate error");
getch();
}

/* ---- function 1 ------------------------------------------------ */

quick1()
{
int i,j,k;

printf("\nstep 1 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 2 : set watchdog timer =1.0 sec");
if (send_command("T=1.0")==0) printf("\n(T=1.0<CR>) Send Command OK");
else printf("\n(T=1.0<CR>) Send Command Error");
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 3 : enable and refresh the watchdog timer");
if (send_command("S")==0) printf("\n(S<CR>) Enable Watchdog OK");
else printf("\n(S<CR>) Enable Watchdog Error");
printf("\n********** press any key to next step **********\n");
for (;;)
      {
      send_command("S");    /* reload the watchdog timer */

/*    this part simulate normal program start */

      printf(".");
      if (kbhit()!=0) {getch(); break;}
```

```
        delay(300);

/*    this part simulate normal program stop */

        }

printf("\nstep 4 : stop refresh the watchdog timer");
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 5 : disable the watchdog timer");
if (send_command("E")==0) printf("\n(E<CR>) Send Command OK");
else printf("\n(E<CR>) Send Command Error");
printf("\n********** press any key to return        **********"); getch();
}

/* ---- function 2 ------------------------------------------------ */

quick2()
{
int i,j,k;

printf("\nstep 1 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 2 : read the RS232 output string");
read_from_com_port();
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 3 : set watchdog timer =1.0 sec");
if (send_command("T=1.0")==0) printf("\n(T=1.0<CR>) Send Command OK");
else printf("\n(T=1.0<CR>) Send Command Error");
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 4 : read the RS232 output string");
read_from_com_port();
printf("\n********** press any key to next step **********"); getch();
```

```
printf("\nstep 5 : enable and refresh the watchdog timer");
if (send_command("S")==0) printf("\n(S<CR>) Enable Watchdog OK");
else printf("\n(S<CR>) Enable Watchdog Error");
printf("\n********** press any key to next step ***********\n");
for (;;)
        {
        send_command("S");    /* reload the watchdog timer */

/*    this part simulate normal program start */

        printf(".");
        if (kbhit()!=0) {getch(); break;}
        delay(300);

/*    this part simulate normal program stop */

        }

printf("\nstep 6 : stop refresh the watchdog timer");
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 7 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 8 : read the RS232 output string");
read_from_com_port();
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 9 : disable the watchdog timer");
if (send_command("E")==0) printf("\n(E<CR>) Send Command OK");
else printf("\n(E<CR>) Send Command Error");
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 10: read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step ***********"); getch();
```

```
printf("\nstep 11: read the RS232 output string");
read_from_com_port();
printf("\n********** press any key to return        **********"); getch();

}

/* ---- function 3 ---------------------------------------------- */

quick3()
{
int i,j,k;

outportb(uBaseWdt+2,0); /* clear IRQ for watchdog timer */
uIrq=0;          /* clear the interrupt indicator */
printf("\nstep 1 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 2 : set watchdog timer =1.0 sec");
if (send_command("T=1.0")==0) printf("\n(T=1.0<CR>) Send Command OK");
else printf("\n(T=1.0<CR>) Send Command Error");
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 3 : enable and refresh the watchdog timer");
printf("\n---------> Interrupt Indicator uIrq=%d <---------",uIrq);
if (send_command("S")==0) printf("\n(S<CR>) Enable Watchdog OK");
else printf("\n(S<CR>) Enable Watchdog Error");
printf("\n********** press any key to next step **********\n");
for (;;)
      {
      send_command("S");
      printf(".");
      if (kbhit()!=0) {getch(); break;}
      delay(300);
      }

printf("\nstep 4 : stop refresh the watchdog timer");
printf("\n********** press any key to next step **********"); getch();
```

```
printf("\nstep 5 : disable the watchdog timer");
printf("\n---------> Interrupt Indicator uIrq=%d <---------",uIrq);
if (send_command("E")==0) printf("\n(E<CR>) Send Command OK");
else printf("\n(E<CR>) Send Command Error");
printf("\n********** press any key to return        ***********"); getch();
uIrq=0; /* clear the interrupt indicator */
}

/* ---- function 4 ------------------------------------------------ */

quick4()
{
int i,j,k;

printf("\nstep 1 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 2 : set watchdog timer =1.0 sec");
if (send_command("T=1.0")==0) printf("\n(T=1.0<CR>) Send Command OK");
else printf("\n(T=1.0<CR>) Send Command Error");
printf("\n********** press any key to next step ***********"); getch();

printf("\nstep 3 : enable and refresh the watchdog timer");
printf("\n---------> Interrupt Indicator uIrq=%d <---------",uIrq);
if (send_command("S")==0) printf("\n(S<CR>) Enable Watchdog OK");
else printf("\n(S<CR>) Enable Watchdog Error");
printf("\n********** press any key to next step ***********\n");
for (;;)
      {
      send_command("S");
      printf(".");
      if (kbhit()!=0) {getch(); break;}
      delay(300);
      }

printf("\nstep 4 : stop refresh the watchdog timer");
```

```
printf("\n********** press any key to next step **********"); getch();
}

/* ---- function 5 ---------------------------------------------- */

quick5()
{
int i,j,k;

printf("\nstep 1 : read the status of the WDT-01");
WDT_status_read();
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 2 : set watchdog timer =1.0 sec");
if (send_command("T=1.0")==0) printf("\n(T=1.0<CR>) Send Command OK");
else printf("\n(T=1.0<CR>) Send Command Error");
printf("\n********** press any key to next step **********"); getch();

printf("\nstep 3 : enable and refresh the watchdog timer");
printf("\n---------> Interrupt Indicator uIrq=%d <---------",uIrq);
if (send_command("S")==0) printf("\n(S<CR>) Enable Watchdog OK");
else printf("\n(S<CR>) Enable Watchdog Error");
printf("\n********** press any key to next step **********\n");
for (;;)
      {
      send_command("S");
      printf(".");
      if (kbhit()!=0) {getch(); break;}
      delay(300);
      }

printf("\nstep 4 : stop refresh the watchdog timer");
printf("\n********** press any key to next step **********");
}
```

```
/* ---- function A ----------------------------------------------- */

read_from_com_port()
{
char c,szString[80];
int    iRetVal,i;
float f1,fTimeOut;

fTimeOut=1000000.0;

/* part 1 : bypass the previous output string */
f1=0;
for (;;)
      {
      if ((inportb(uBaseUart+5)&0x01) != 0) /* check line ready */
          {
          c=inportb(uBaseUart);
          if (c==0x0d) break;       /* wait until 0x0d */
          f1=0;
          }
      else
          {
          f1++;
          if (f1>fTimeOut)    /* timeout control */
         {
         printf("\nCom Port Read Error --> Time Out ");
         return(1);
         }
          }
      }

/* part 2 : receive the new complete output string */
f1=0;
i=0;
for (;;)
      {
      if ((inportb(uBaseUart+5)&0x01) != 0) /* check line ready */
          {
```

```
            c=inportb(uBaseUart)&0xff;
            if (c==0x0d) break;       /* wait until 0x0d */
            szString[i++]=c;     /* save the output string */
            f1=0;                     /* reset the timeout timer */
            }
        else
            {
            f1++;
            if (f1>fTimeOut)    /* timeout control */
           {
           printf("\nCom Port Read Error --> Time Out ");
           return(1);
           }
             }
        }

szString[i]=0;              /* string must terminated by 0 */
printf("\nCom --> %s",szString);
return(0);
}
```

```
/* ---- function B ------------------------------------------------ */

WDT_status_read()
{
char cExtPower,cOverTem,cTimeOut,cVal;

/*
printf(" --> read    status of WDT-01");
*/

cVal=inportb(uBaseWdt+1)&0x07; /* read status */
cExtPower=cVal&0x04;                /* !=0 --> with External    Power  */
cOverTem=cVal&0x02;                 /* !=0 --> Temperature    OK       */
cTimeOut=cVal&0x01;                 /* !=0 --> Watchdog Timer OK    */

printf("\nWDT-01 status = %x",cVal);
if (cTimeOut!=0)    printf("\nWatchdog Timer OK");
```

```
else printf("\nWatchdog Timeout");

if (cOverTem!=0)    printf("\tTemperate OK");
else printf("\tOver Temperate");

if (cExtPower!=0) printf("\tExternal Power OK");
else printf("\tNo External Power");
}
```

```
/* ---- function C ------------------------------------------------ */

WDT_command()
{
char cmd[80];
printf(" --> send command to WDT-01");

printf("\ncommand-->"); scanf("%s",cmd);
if (send_command(cmd)==0) printf("\n[%s] Command Send OK",cmd);
else printf("\n[%s] Command Send Error",cmd);
getch();
}
```

```
/* ---- function T ------------------------------------------------ */

Set_Watchdog_Timer ()
{
printf(" --> Set Watchdog Timer=2.0");

if (send_command("T=2.0")==0) printf("\n(T=2.0<CR>) Send Command OK");
else printf("\n(T=2.0<CR>) Send Command Error");
}
```

```
/* ---- function S ------------------------------------------------ */

Start_Watchdog_Timer ()
{
printf(" --> Start Watchdog Timer");
```

```
if (send_command("S")==0) printf("\n(S<CR>) Send Command OK");
else printf("\n(S<CR>) Send Command Error");
}
```

```
/* ---- function E ------------------------------------------------ */

End_Watchdog_Timer ()
{
printf(" --> End Watchdog Timer");

if (send_command("E")==0) printf("\n(E<CR>) Send Command OK");
else printf("\n(E<CR>) Send Command Error");
}
```

```
/* ---- function P ------------------------------------------------ */

Power_Failure_Alarm_Clear()
{
printf(" --> Power Failure Alarm Clear");

if (send_command("P")==0) printf("\n(P<CR>) Send Command OK");
else printf("\n(P<CR>) Send Command Error");
}
```