#### 使用 MITSUBISHI GX-Configurator DP 配置 Profibus-DP 网络

叶工,我升级了新软件,重新配置了一下文件,请见下面内容,如有问题,再联系我。 图尔克 张工

一、使用 MITSUBISHI GX-Configurator DP 配置 Profibus-DP 网络

1、运行 GX-Configurator DP 软件,新建一个项目,选择主站模块。

Network Setup		×
Select Module Type	© Qn ⊂ QnA/A ⊂ EX	
MELSEC Device	QJ71PB92V (PROFIBUS DP V1/V2 Master) QJ71PB92D (PROFIBUS DP V0 Master)	<b>_</b>
	UJ71PB92V (PROFIBUS DP V1/V2 Master) QJ71PB93D (PROFIBUS DP VO Slave) QJ71PB92D - Mode O (PROFIBUS DP VO Ma	ster)

2、双击出现的主站图标,在出现的"Master Settings"对话框中可进行相关参数设置,如总 线波特率等。

Laste	er Settings		×
Modul	e QJ71PB92D		I/O Mode 🚺
Vendo	MITSUBISHI ELECTRIC CORPORATION		Revision AA (Mode E)
	Name	PROFIBUS M	aster
	B <u>a</u> udrate	1.5 Mbps	•
	FDL address	0	[0 - 125]
	Starting <u>I</u> /O number	000	[0x0 - 0xFE(
	Error action flag	<u> </u>	ear' State
	Min. slave interval	20	[1 - 65535] * 100 衽
	<u>P</u> olling timeout	50	[1 - 65535] * 1 ms
	Data c <u>o</u> ntrol time	100	[T_wd * 6 - 655 * 10 ms
	🔲 🖞 atchdog 🛛 Slave Watchdog ti	n  5	[1 - 65025] * 10 ms
	Estimated bus cycle time	3	ms
	🗖 Autom. Refresh 🛛 🗖 Consisten	cy	
	Watchdog for time sync.	0	[0 - 65535] * 10 ms
(0)	K Cancel De <u>f</u> ault	<u>B</u> us Para	um.

3、在出现的"Slave Parameter Settings"对话框中对该从站进行相关设置。 FDL Address:从站地址。

Sla	we Para	meter Settings				
Mod	el	BL67-GW-DPV1			Revision	
Ven	dor	Hans Turck GmbH & Co.	KG		V4.2	
	-Slave Pr	operties				_
	<u>N</u> ame			Slave_Nr	_001	
	F <u>D</u> L Addr	ess		6	[0 - 125]	
	🔽 <u>M</u> atch	dog Slave Watchd	og <u>t</u> ime	5	[1 - 6502	5] * 10
	min T_sd	r		11	[1 - 255]	
	Group id number	entification	□ Grp	1 🗖 Grp	2 🗖 Grp 3 🗖	Grp <u>&lt;</u>
			Grp	š   Grp	t   Grp ]   	Grp <u>č</u>
	▼ Sla <u>v</u> e	is active	Sync	(Output)	) 🗌 F <u>r</u> eeze	(Input)
	🔲 Ignor	e <u>A</u> utoClear	🔲 Ini t	ialize sl	lave when fai	ling to re
	🔲 Swa <u>p</u> (	I/O Bytes in Master				
	<u>DP ⊻</u> 1/V2	Slave Parameters				
	OK	Cancel	De <u>f</u> ault	<u>U</u> s	ser Param.	<u>S</u> elect Modules

4、点击"Slave Parameter Settings"对话框中"Select Modules",在出现的"Select Modules" 对话框中选择相应的 I/O 模块进行添加。完成从站模块配置后点击"OK"返回"Slave Parameter Settings"对话框。

Slave Todules
Info       Modules       64       are possible       Max. Data size       488       Byte(s         I/O usage       24       / 24       Byte(s)       Max. I/O sizes       244       / 244       Byte(s)         User_Prm_Data usage       14       Byte(s)       Max. User_Prm_Data size       237       Byte(s)
Module Configuration Available Modules BL67-ISSI [S] BL67-ISSI [T] BL67-1CVI [S] BL67-2RFID-A [S] BL67-2RFID-A [S] BL67-2RFID-C [S] BL67-2RFID-C [S] BL67-2RFID-S [T] BL67-2RFID-S [T] BL67-2RFID-S [S] BL67-2RFID-S [S]
OK Cancel

5、设置 I/O 映射



🐙 IELSOFT GX Configurator-DP - [u	ntitled]			
DP <u>F</u> ile <u>S</u> etup <u>O</u> nline <u>T</u> ools <u>V</u> iew <u>W</u> indow	Help			
D 🛎 🖃 🎒 🕌 👯 🍠	<b>№?</b> i			
	ROFIEUS Master			
ſ	0]			
Ľ	<u>M</u> odify S	ettings		
	Transfer	► F		
_				
s	lave_Mr_001			
	6]			
I/0 Tapping				×
I/0 Mapping				
I/O Espping	<b>DUT Var. Identifier</b> VHAOSLVG	Identifier	DUT Type	Humber :
<b>I/0 Tapping</b> 9J71PB92D 6: Slave_Nr_001 (BL67-GW-DFV1)	<b>DUT Var. Identifier</b> VHAOSLV6 VHAOSLV6	Identifier inputs outputs	<b>DUT Type</b> WORD WORD	<b>Humber</b>
I/0 ■apping     QJ71PB92D     6: Slave_Nr_001 (BL67-GW-DPV1)     BL67-2RFID-S [T]	<b>DUT Var. Identifier</b> wHAOSLVG wHAOSLVG	Identifier inputs outputs	BUT Type WORD WORD	<b>Hunber</b> :
I/0 Lapping	DUT Var. Identifier wH&OSLV6 wH&OSLV6	Identifier inputs outputs	DUT Type WORD WORD	Number           12           12
I/0 Tapping 9,771PB92D 6: Slave_Nr_001 (BL67-GW-DFV1) ∬ BL67-2RFID-S [T]	DUT Var. Identifier wH&OSLV6 wH&OSLV6	Identifier inputs outputs	DUT Type WORD WORD	Humber           12           12
I/0 Tapping 9,J71PB92D 6: Slave_Nr_001 (BL67-GW-DPV1) € BL67-2RFID-S [T]	DUT Var. Identifier wH&OSLV6 wH&OSLV6	Identifier inputs outputs	DVT Type WORD WORD	Humber           12           12
▼ 1/0 Tapping       Image: gjT1PB92D       Image: gjT1PB92D <td>DUT Var. Identifier wHAOSLV6 wHAOSLV6</td> <td>Identifier inputs outputs</td> <td>DUT Type WORD WORD</td> <td>Humber           12           12</td>	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier inputs outputs	DUT Type WORD WORD	Humber           12           12
▼ 1/0 Tapping QJ71PB92D 6: Slave_Nr_001 (BL67-GW-DFV1) Ø BL67-2RFID-S [T]	DUT Var. Identifier wH&OSLV6 wH&OSLV6	Identifier inputs outputs	BUT Type WORD WORD	Bunber : 12 12 12
▼ 1/0 Tapping 9,771PB92D 6: Slave_Nr_001 (BL67-GW-DFV1) Ø BL67-2RFID-S [T]	DUT Var. Identifier wH&OSLV6 wH&OSLV6	Identifier inputs outputs	DUT Type WORD WORD	Humber           12           12
I/0 Tapping 9,JT1PB92D 6: Slave_Nr_001 (BL67-GW-DPV1) € BL67-2RFID-S [T]	DUT Var. Identifier wHAOSLVG wHAOSLVG	Identifier inputs outputs	DUT Type WORD WORD	Humber           12           12
I/0 Tapping       ■     9J71PB92D       ●     ●	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier inputs outputs	DVT Type WORD WORD	Humber           12           12
I/0 Tapping          I/0 Tapping         9,711PB92D         Image: Size _ Nr_001 (BL67-GH-DFV1)         Image: Size _	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier inputs outputs	BUT Type WORD WORD	<b>Bunber</b> 12 12
▼ 1/0 Tapping QJ71PB92D 6: Slave_Nr_001 (BL67-GW-DFV1) Ø BL67-2RFID-S [T]	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier	BUT Type WORD WORD	Bunber : 12 12
I/0 Tapping 9,711PB92D 6: Slave_Nr_OO1 (BLA7-CW-DPV1) ∅ BLA7-2RFID-S [T]	DUT Var. Identifier wHAOSLVG wHAOSLVG	Identifier inputs outputs	BUT Type WORD WORD	Humber           12           12
▼ 1/0 Tapping       Image: Particular Structure     9JT1PB92D       Image: Structure     6: Slave_Nr_001 (BL67-GW-DPV1)       Image: Structure     6: Slave_Nr_001 (BL67-GW-DPV1	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier inputs outputs	DUT Type WORD WORD	Humber           12           12
Y 1/0 Tapping QJTIPB92D 6: Slave_Nr_001 (BL67-CH-DPV1) ■ BL67-2RFID-S [T]	DUT Var. Identifier wHAOSLV6 wHAOSLV6	Identifier inputs outputs	DUT Type WORD WORD	<b>Rumber</b>
I/0 Tapping 9,771PB92D 6: Slave_Nr_001 (BL67-C#-DFV1) ∅ BL67-2RFID-S [T]	DUT Var. Identifier WHAOSLV6 WHAOSLV6 VHAOSLV6	Identifier inputs outputs	BUT Type WORD WORD	<b>Bunber</b> 12 12

每个模块占用 12 WORD 输入和 12 WORD 输出。

- 6、保存项目
- 7、下载信息





8、启动 Profibus 连接

🐙 IELSOFT GX Configurator-DP - [untitled]
DP File Setup Online Tools View Window Help
□ ☞ ■ Transfer Setup Transfer • <b>? № i</b>
Start/Stop PROFIBUS [0]
Slave_ <b>H</b> r_001

## 二、使用 GX-Developer 编写 QJ71PB92D 启动程序并监控从站数据

1、在 GX-Developer 软件中新建一个工程,选择所使用的 PLC 类型。

FMELSOFT系列 GX Developer				_I_×
工程(E) 显示(V) 在线(O) 诊断(	D) 工具(I) 帮助(H)·			
	<u>u qqq 22 qc</u>			
	创建新工程		×	
	PLC系列 QCPV (Qmode)	<u> </u>	确定	
R RADE PR	的大型		48(6	
	Q02 (H) -程序类刑			
	<ul> <li>● 梯形图逻辑</li> <li>○ SFC 「 MELSAP-L</li> </ul>	<ul> <li>○ 无标号</li> <li>○ 标号程序</li> <li>○ 标号+FB程序</li> </ul>		
	□ 生成和程序名同名的软元件内 □ 工程名设置	存数据		
	□ 设置工程名			
	驱动器/路径 C:\MELSEC\GPPW			
	工程名		四先	
	标题			
工程				
准备完毕	<u></u>		1	

2、在主程序中添加如下程序段,用于启动 QJ71PB92D

注: QJ71PB92D本身占用 32bit 的 I/O 数据,用于 CPU 对该模块的控制。本例中 QJ71PB92D 为 CPU 机架上的第一个模块,则该模块的 32bit 的 I/O 数据为 X0-X31/Y0-Y31。其中 Y0 置 1 用于启动 PROFIBUS 总线。



3、编写程序后,点击"变换"菜单下的"变换"命令进行程序编译。

MELSOFT系列 GI Beveloper D:\	BELSEC\GPPT\TURCK_DP -	【梯形图(写入)	BAIR	39 步1
工程(E) 编辑(E) 查找/替换(E)	麦椿(r) 星米(V) 在线(0)	诊断(印) 工具(印)	窗口创	帮助①
	安换( <u>C</u> )	74		
程序	表換(全部程序)(的			
北백생생양이는	変換(通行中写入)(6)	Shift+74		
F5 sF5 F6 sF6 F7 F8 F9 sF	0 cF9 cF10 _sF7 sF8 aF7 aF	8 aF5 aF5 aF0 F	ele O	

4、点击"在线"菜单下"PLC写入"命令,打开"传输设置"对话框。在该对话框中选择 需要下载的选项,点击"执行"。本例中只是下载程序,所以只选中"Main程序"。

amelSOFT系列 GE Developer D:\MELSEC\GPP#\TURCK_DP - [條形图(写)	A) BAIR	37 步
□ I程(2) 编辑(2) 查找/替换(2) 变换(2) 显示(2) 在线(2) 诊断(2) I	AC BOO	帮助创
11 41 44 44 97 67 68 69 679 69 679 670 877 56 10 000 10 10 10 10 10 10 10 10 10 10 10	1. I.	
PLC写入		×
蜂猎猪口 COW1 <→>CPU 模块		_
編接FLC P14号 下 14号 本站 FLC类型 R0200		
対象内存 程序内存/软元件内存 _ 索引		
文件选择 軟元件数据 程序   程序共用   各程序	执行	
参数+程序 选择所有 取消所有选择	关闭	
▶ 国人程序标签(四), 扭, 结构体: 相关内存 程序因存/数元体因存 🖌	口令设置	
	美酥功能	
日 「「」 教元件注释	传输设置	
	重要关键子。	
□ <u>□</u> <u>PLL</u> /网络/远程口令	远程操作	
	冗余操作	
	清除PLC内存.	
- * 非市市到	格式化和C内存	4.0
C 所有防御	整理PLC内存.	
6 元田裕定 22 10 - 192767	生成标题	
空余容量 所有可用容量	字节	

#### 5、应用

PLC 的 DP 模块中的接收缓冲区地址为 0H – 3BFH PLC 的 DP 模块中的发送缓冲区地址为 3C0H – 77FH 每个 BL67-2RFID-S 模块占用 12 WORD 输入和 12 WORD 输出。

此处所有数据都要从 DP 模块的缓冲区中读取,写入。所以为了方便易懂,只监控缓冲区。如果想要操控 CPU 的地址单元,需自己写一段程序,才可以将该缓冲区的地址单

元 move 到 CPU 的地址单元。

1、原始状态:



# PLC WORLD

模块起始地址	: 0000	(16进制)						
缓冲存储区地	址 300 (	0 10进制 🖲 16进	制					
监视格式:	● 位& 字 :	显示: 💽 16位整	数	数值:	◉ 10进制		监视	开始
	○ 多点位	〇 32位書	数		○ 16进制	Ē	0440	
	〇 名古宝	○ 尔勒				_	监视	19止
	o swit	io <del>X</del> ai						
		C ASCII	字符					
地址	+FEDC+B	A 98 +7654	+3 2 1 0			•	选项	设置
03C0	0000 0	000 0000	0000		0			
03C1	00000	000 0000	0000		0		1	第1个通道
03C2	00000	000 0000	0000		0			
03C3	0 0 0 0 0	000 0000	0 0 0 0		0		野⊧	N Output
03C4			0000		0	- 1		
10305	00000		0000					
0305	00000		0000		0		_	
0301	000000		0000		0			第2个通道
0309	000000		0000		0	1		
03C4	0 0 0 0 0	000 0000	0 0 0 0	_	0			e) Output
03CB	00000	000 0000	0 0 0 0		0			
03CC	00000	000 0000	0 0 0 0		0			
03CD	0 0 0 0 0	0 0 0 0 0 0 0	0 0 0 0		0			
03CE	0 0 0 0 0	0 0 0 0 0 0 0	0 0 0 0		0			
03CF	0 0 0 0 0	0 0 0 0 0 0 0	0 0 0 0		0			
03D0	0 0 0 0	000 0000	0000		0			
03D1	0 0 0 0 0	000 0000	0000		0			
03D2	00000	000 000	0000		0			
03D3	00000	000 000	0000		0			
03D4	00000	000 0000	0000		0			
03D5	00000	000 0000	0000		0			
03D6	0 0 0 0 0	000 0000	0000		0	-		

2、设置读写数据长度,例如 8 bytes , RESET 操作
 3C0= "0781H"为 Reset 启动, 3C0= "0780H"为 Reset 完成,如下图。

模块起始地址	: 0000	(16进制	D		
缓冲存储区地:	t <u>it 300</u>	_ ○ 10进	制 🖲 16进制		
监视格式:	◉ 位&字	显示:	④ 16位整数	数值: 🔘 10进制	监视开始
	○ 多点位		○ 32位整数	ⓒ 16进制	1001011611
	〇 多点字		○ 实数		
	- Senier		C ASCII字符		
<u>#6 bl</u>	+REDC	+B A 9 8	+7654 +3210	-	选项设置
03C0	0000	$0\ 1\ 1\ 1$	1000 0000	0780	
03C1	0 0 0 0	0000	0000 0000	0000	第 1 个诵道
03C2	0 0 0 0	0000	0000 0000	0000	
03C3	0 0 0 0	0000	0000 0000	0000	的 Output
03C4	0 0 0 0	0000	0000 0000	0000	. 1
03C5	0000	0000	0000 0000	0000	
03C6	0 0 0 0	0000	0000 0000	0000	
03C7	0 0 0 0	0000	0000 0000	0000	第 2 个通道
03C8	0 0 0 0	0000	0000 0000	0000	· 的 Output
03C9	0 0 0 0	0000	0000 0000	0000	11 Output
OSCA	0 0 0 0	0000	0000 0000	0000	
03CB	0 0 0 0	0000	0000 0000	0000	
03CC	0 0 0 0	0000	0000 0000	0000	
03CD	0 0 0 0	0000	0000 0000	0000	
03CE	0 0 0 0	0000	0000 0000	0000	
03CF	0 0 0 0	0000	0000 0000	0000	
03D0	0 0 0 0	0000	0000 0000	0000	
03D1	0 0 0 0	0000	0000 0000	0000	
03D2	0 0 0 0	0000	0000 0000	0000	
03D3	0 0 0 0	0000	0000 0000	0000	
03D4	0 0 0 0	0000	0000 0000	0000	
03D5	0 0 0 0	0000	0000 0000	0000	
03D6	0 0 0 0	0000	0000 0000	0000 -	

模块起始地址: (0000 (16进制)		
缓冲存储区地址 00 〇 10进制 🕫 16进制		
监视格式: ⓒ 位& 字 显示: ⓒ 16位整数 数值: ○ 10进制	J	监视开始
○ 多点位	J	
○ 多点字 ○ 实数		
C ASCII字符		
絶址 +FEDC +BA98 +7654 +3210		选项设置
	0	<b>竺 1</b> 公子送码
0002 0000 0000 0000 0000 0000	0	
	0	Input
0004 0000 0000 0000 0000 0000	0	_
0005 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	
0005 0000 0000 1000 0000 0080	0	
0007 0000 0000 0000 0000 0000	0	<b>答 2</b> 众语送的
	0	_ 弗 2 个 通道的
0009 0000000000000000000000000000000000	0	Input
000A 0000 0000 0000 0000 0000	0	1
000B 0000 0000 0000 0000 0000	0	
000C 000000000000000000000000000000000	0	
000D 000000000000000000000000000000000	0	
000E 000000000000000000000000000000000	0	
000F 000000000000000000000000000000000	0	
0010 0000000000000000000000000000000000	0	
0011 0000 0000 0000 0000 0000	0	
0012 0000 0000 0000 0000 0000	0	
0013 0000 0000 0000 0000 0000	0	
0014 0000 0000 0000 0000 0000	0	
0015 0000 0000 0000 0000 0000	0	
0016 0000 0000 0000 0000 0000	0 -	

# PLC WORLD

## 3、Write 操作:

3C0= "0788H"为 Write 启动, 3C0= "0780H"为 Write 完成, 如下图。

模块起始地址	: 000	0 (16进制	b		
缓冲存储区地	t <u>il 300</u>	○ 10进	制 🖲 16进制		
监视格式:	◉ 位& 字	显示:	⊙ 16位整数	数值: 🔘 10进制	监视开始
	〇 多点位		○ 32位整数	⊙ 16进制	
	〇 多古字		C 实数		
	201111		C ASCII字符		
地址	+FEDO	: +B A 9 8	+7654 +321	0	]▲  选项设置
03C0	0000	0111	1000 000	0 0780	
03C1	0 0 0 0	0000	0000 000	0 0000	
03C2	1 1 1 1	1 1 1 1	1 1 1 1 1 1 1 1	1 FFFI	第 1 个通道
03C3	1 1 1 0	1110	1 1 1 0 1 1 1	0 EEEH	E 的 Output
03C4	1 1 0 1	1101	1 1 0 1 1 1 0	1 DDDI	
03C5	1100	) 1100	1100 110	0 CCCC	
03C6	0 0 0 0	) 0000	0000 000	0 0000	
03C7	0 0 0 0	0000	0000 000	0 0000	9 第 2 个通道
03C8	0 0 0 0	0000	0000 000	0 0000	
0309	0 0 0 0	0000	0000 000	0 0000	D B) Output
03CA	0000	0000	0000 000	0 0000	
03CB	0000	) 0000	0000 000	0 0000	1
03CC	0 0 0 0	0000	0000 000	0 0000	D
03CD	0 0 0 0	0000	0000 000	0 0000	D
03CE	0000	0000	0000 000	0 0000	D
03CF	0 0 0 0	0000	0000 000	0 0000	D
03D0	0000	0000	0000 000	0 0000	D
03D1	0 0 0 0	0000	0000 000	0 0000	D
03D2	0 0 0 0	0000	0000 000	0 0000	D
03D3	0000	0000	0000 000	0 0000	D
03D4	0000	0000	0000 000	0 0000	D
03D5	0000	0000	0000 000	0 0000	
03D6	0000	0000	0000 000	0 0000	) <b>-</b>

模块起始地址	: 0000	(16进制)	)					
缓冲存储区地址	L P	〇 10进行	制 🖲 16进制					
监视格式:	◉ 位&字	显示:	ⓒ 16位整数	数值: 🔘 10进制		监	视开始	
	〇 多点位		○ 32位整数	④ 16进制	Ē	ul <del>te</del> t		
	〇 多点字		○ 实数		_	μπ.		
			C ASCII字符					
地址	+FEDC	+BA98	+7654 +3210		•	选]	项设置	
0000	0 0 0 0	0000	0101 1000	0058	7 7	-		
0001	0 0 0 0	0000	0000 0000	0000	_		第 1 个诵i	首的
0002	0 0 0 0	0 0 0 0	0000 0000	0000	. г		NI I KUK	6H1
0003	0 0 0 0	0000	0000 0000	0000		软	Input	
0004	0 0 0 0	0 0 0 0	0000 0000	0000	-		_	
0005	0 0 0 0	0000	0000 0000	0000				
0006	0 0 0 0	0 0 0 0	1000 0000	0080				
0007	0 0 0 0	0000	0000 0000	0000			<b>箆</b> 2 个通	首的
0008	0000	0000		0000	-		37 Z T XQ	
0009	0000	0000		0000			Input	
AUUUA	0000	0000		0000				
0008	0000	0000		0000				
0000	0000	0000	0000 0000	0000				
0000	0000	0000		0000				
0005	0000	0000		0000				
0000	0000	0000		0000				
0010	0000	0000		0000				
0011	0000	0000		0000				
0012	0000	0000		0000				
0013	0000	0000		0000				
0014	0000	0000		0000				
0015	0000	0000		0000	-1			
0016	0000	0000	0000 0000	0000	•			

完成上述操作后,把载码体靠近读写头的读写区域,进行写数据到载码体操作。

# PLC WORLD

### 4、Read 操作:

3C0= "0790H"为 Read 启动, 3C0= "0780H"为 Read 完成, 如下图。

模块起始地址	: 0000	(16进制	)					
缓冲存储区地址	1 <u>1</u> 3C0	_ ○ 10进	制 🖲 16进制	钊				
监视格式:	◉ 位&字	显示:	④ 16位整	数	数值: 🔘	10进制		监视开始
	○ 多点位		○ 32位整	数		16进制		收测信止
	○ 多点字		○ 实数					
			C ASCII₽	符				
地址	+FEDC	+BA98	+7654	+3 2 1 0				选项设置
0300	0000	$0\ 1\ 1\ 1$	1000	0000		0780		
03C1	0 0 0 0	0000	0 0 0 0	0000		0000		
03C2	1 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1		FFFF		第 1 个逋道
03C3	1 1 1 0	1 1 1 0	$1 \ 1 \ 1 \ 0$	$1 \ 1 \ 1 \ 0$		EEEE		的 Output
03C4	1 1 0 1	1 1 0 1	1 1 0 1	$1 \ 1 \ 0 \ 1$		DDDD		
03C5	1 1 0 0	1100	1100	1 1 0 0		CCCC	4	
03C6	0 0 0 0	0000	0000	0000		0000		
03C7	0 0 0 0	0000	0000	0000		0000		第 2 个通道 📗
03C8	0 0 0 0	0000	0000	0000		0000		的 Output
03C9	0 0 0 0	0000	0000	0000		0000		HJ Output
03CA	0 0 0 0	0000	0000	0000		0000	L	
03CB	0 0 0 0	0000	0000	0000		0000		
03CC	0 0 0 0	0000	0000	0000		0000		
03CD	0 0 0 0	0000	0000	0000		0000		
03CE	0 0 0 0	0000	0000	0000		0000		
03CF	0 0 0 0	0000	0000	0000		0000		
03D0	0 0 0 0	0000	0000	0000		0000		
03D1	0000	0000	0000	0000		0000		
03D2	0 0 0 0	0000	0000	0000		0000		
03D3	0000	0000	0000	0000		0000		
03D4	0 0 0 0	0000	0000	0000		0000		
03D5	0 0 0 0	0000	0000	0000		0000	_	
0306	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0		0000	<b>T</b>	

模块起始地址:	0000	(16进制	)					
缓冲存储区地址	e o	〇 10进	制 🗭 16进制					
监视格式:	◉ 位&字	显示:	◉ 16位整数	数值: 🔘 10进制		监	视开始	
	○ 多点位		○ 32位整数	④ 16进制	Ē	g/e	初停止	
	○ 多点字		○ 实数		_			
			○ ASCII字符					
地址	+FEDC	+BA98	+7654 +3210		•	选	项设置	
0000	0000	0000	0101 1000	0058		- T		_
0001	0 0 0 0	0000	0000 0000	0000			第 1 个 诵 道 f	約
0002	0 0 0 0	0000	0000 0000	0000				11
0003	0 0 0 0	0000	0000 0000	0000		软	Input	
0004	0 0 0 0	0 0 0 0	0000 0000	0000	_			
0005	0000	0000		0000				
0006	0000	0000	1000 0000	0800				
0007	0000	0000	0000 0000	0000			第2个通道	的
0008	0000	0000	0000 0000	0000	-			
0003	0000	0000		0000			Input	
0008	0000	0 0 0 0	0000 0000	0000				
0000	0 0 0 0	0 0 0 0	0000 0000	0000				
0000	0 0 0 0	0 0 0 0	0000 0000	0000				
000E	0 0 0 0	0 0 0 0	0000 0000	0000				
000F	0 0 0 0	0 0 0 0	0000 0000	0000				
0010	0 0 0 0	0000	0000 0000	0000				
0011	0 0 0 0	0000	0000 0000	0000				
0012	0 0 0 0	0000	0000 0000	0000				
0013	0 0 0 0	0000	0 0 0 0 0 0 0 0	0000				
0014	0 0 0 0	0000	0000 0000	0000				
0015	0 0 0 0	0 0 0 0	0000 0000	0000				
0016	0 0 0 0	0000	0000 0000	0000	•			

完成上述操作后,把载码体靠近读写头的读写区域,进行从载码体中读数据操作。 如下图,数据读写成功。

模块起始地址:	0000	(16进制	D		
缓冲存储区地址	L O	〇 10进	制 🖲 16进制		
监视格式:	◉ 位&字	显示:	◎ 16位整数	数值: 🔘 10进制	监视开始
	〇 多点位		○ 32位整数	④ 16进制	些训练正
	○ 多点字		○ 实数		
			C ASCII字符		
地址	+FEDC	+B A 9 8	+7654 +3210		▲ 选项设置
0000	0000	0000	1001 1000	0098	
0001	0 0 0 0	0000	0000 0000	0000	—
0002	1 1 1 1	1 1 1 1	1 1 1 1 1 1 1 1 1	FFFF	- 用Ⅰ个 <b>进</b> 担的
0003	1 1 1 0	1 1 1 0	1110 1110	EEEE	Input
0004	1 1 0 1	1 1 0 1	1 1 0 1 1 1 0 1	DDDD	
0005	1100	1100	1100 1100	CCCC	
0006	0000	0000	1000 0000	0080	
0007	0000	0000	0000 0000	0000	第2个通道的
0008	0000	0000		0000	Input
0003	0000	0000		0000	mput
0008	0 0 0 0	0 0 0 0		0000	
0000	0 0 0 0	0 0 0 0	0000 0000	0000	
000D	0 0 0 0	0 0 0 0	0000 0000	0000	
000E	0 0 0 0	0 0 0 0	0000 0000	0000	
000F	0 0 0 0	0000	0000 0000	0000	
0010	0 0 0 0	0000	0000 0000	0000	
0011	0 0 0 0	0 0 0 0	0000 0000	0000	
0012	0 0 0 0	0 0 0 0	0000 0000	0000	
0013	0 0 0 0	0000	0000 0000	0000	
0014	0 0 0 0	0000	0000 0000	0000	
0015	0 0 0 0	0000	0000 0000	0000	
0016	0 0 0 0	0000	0000 0000	0000	<b>•</b>

#### RFID 读写过程参照资料

常用参数说明: INPUT DATA:

 1.DONE
 完成读写操作

 2.BUSY
 接收到读写指令,正在等待载码体

 3.XCVR CON
 读写头与模块正常通讯标志

 4.XCVR ON
 读写头处于激活状态

 5.READ DATA
 读出的数据(最多一次读 8Byte)

INPUT	BYTE	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
Channel 0	0	DONE	BUSY	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
	1	Error Cod	Error Code										
	2	Error Code 1											
	3	Reserved	Reserved										
	4	READ DA	TA (8 Byte)										
	5												
	10												
	10												
	10	-											
Channel 1	10 11 12	DONE	BUSY	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13	DONE Error Cod	BUSY	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14	DONE Error Cod Error Cod	BUSY e e 1	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14 15	DONE Error Cod Error Cod Reserved	BUSY e e 1	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14 15 16	DONE Error Cod Error Cod Reserved READ DA	e e 1 TA (8 Byte)	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14 15 16 17	DONE Error Cod Error Cod Reserved READ DA	e e 1 TA (B Byte)	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14 15 16 17	DONE Error Cod Error Cod Reserved READ DA	e e 1 TA (8 Byte)	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				
Channel 1	10 11 12 13 14 15 16 17  22	DONE Error Cod Reserved READ DA	BUSY e 1 TA (8 Byte)	ERROR	XCVR CON	XCVR ON	TP	TFR	Reserved				

#### OUTPUT DATA:

1.XCVR	激活读写头	5	
2.NEXT	读写下一个	载码体	
3.TAG ID	读取载码	体的 ID 号	
4.READ	读指令		
5.WRITE	写指令		
6.RESET	复位指令		
7.Byte Cou	nt 2.1.0	一次读写的字节数 000=1、001=2、010=3 111=8	(最多一次
		读写 8Byte)	
8.Address	high byte	读写载码体内数据的起始字节数(高8位)	
9. Address	low byte	读写载码体内数据的起始字节数(低 8 位)	
10.WRITE	DATA 要	写入的数据(最多一次写 8Byte)	

OUTPUT	BYTE	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0					
	- 3	- (s					- (s. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.	2	1					
Channel 0	0	XCVR	NEXT	TAG ID	READ	WRITE	TAG INFO	XCVR	RESET					
								INFO						
	1	Reserved					Byte	Byte	Byte					
	-				Count 2	Count 1	Count 0							
	2	Address 1	Address high byte											
	3	Address I	Address low byte											
	4	WRITE D	ATA (8 Byte)											
	5													
	an,													
	10													
	11													
Channel 1	12	XCVR	NEXT	TAGID	READ	WRITE	TAG INFO	XCVR	RESET					
								INFO						
	13	Reserved	2	1101			Byte	Byte	Byte					
							Count 2	Count 1	Count 0					
	14	Address I	nigh byte											
	15	Address I	ow byte											
	16	WRITE D	ATA (8 Byte)											
	17		ST 50.050											
	22													
	23													

1. 读数据(使用 Channel 0):

首先激活读写头:将 XCVR 置 1:指定要读取的数据大小 Byte Count 2.1.0=111, 一次读取 8byte:指定读取数据的起始字节数,Address high byte=0,Address low byte=0,从载码体的第0个字节开始读:将 READ 置 1,此时 BUSY 为 1,DONE 为 0, 等待读取载码体数据:将载码体放入读写范围内,读完成后,BUSY 由 1 变 0,DONE 为 1,读取的 8 个 byte 数据存储到相应的存储区中。

2. 写数据 (使用 Channel 0):

首先激活读写头:将 XCVR 置 1:指定要写入的数据大小 Byte Count 2.1.0=111, 一次读取 8byte:指定写入数据的起始字节数,Address high byte=0,Address low byte=0,从载码体的第0个字节开始写:将要写入的数据放入相应的存储区中Address high byte,Address low byte:将 WRITE 置 1,此时 BUSY 为 1,DONE 为 0,等待写入 载码体数据:将载码体放入读写范围内,写完成后,BUSY 由 1 变 0,DONE 为 1,写入 的 8 个 byte 数据存储到载码体中。