UNIVERSITY OF MICROCHIP
UOFM
MASTERs 2007
第八届中国技术精英年会

实验一
协议栈配置与网络构成

目标：在本实验中，你将：

- 设置独特的长地址
- 加入网络
- 发送一条消息（翻转 LED RA1）给对等节点

给有经验的 C 程序员的实验指导：

> 注：对于 C 编程经验较少的学员，请参阅实验指导的第二部分，获得更多详细信息。

1. 设置独特的长地址

☐ 启动 MPLAB IDE

☐ 在 MPLAB IDE 菜单中，选择 **Project > Open**，而后浏览 **C:\MASTERS\11052\MiW End Device\MiWi – End Device – PIC18 – C18.mcp**。

☐ 打开文件 **`MiWiDefs.h`**。找到从 **EUI_0** 一直到 **EUI_7** 的`#define`。这是设备特有的扩展地址。就本实验而言，只使用 ASCII 码（见附录 A）。键入地址，比如你的姓名或昵称之类。确认你的地址的独特性。例如：

```
#define EUI_0 ‘M’
#define EUI_1 ‘r’
#define EUI_2 ‘.’
#define EUI_3 ‘S’
#define EUI_4 ‘m’
#define EUI_5 ‘i’
#define EUI_6 ‘t’
#define EUI_7 ‘h’
```

不要忘记删除注释，注释前以“!!”标记

注：在实际应用中，EUI_5 到 EUI_7 是从 IEEE 申请得到的唯一 OUI。

2. 加入网络

在最初打开节点的电源时，事件发生顺序如下：

A. 信标请求（已有网络存在吗？）
Yes－－决定你是否要加入网络（要加入则发送**关联请求**）。
No －－如果你是 **PAN 协调器**，把节点升级成为网络的 **PAN 协调器**（即，建立新的网络）。否则，继续发送**信标请求**，直到你接收到**信标**为止。
B. 加入网络（发送**关联请求**）

接下来，你将添加代码，开始发送信标请求。

☐ 打开文件 **MiWi.h**。找到宏 **DiscoverNetworks( )**。这个宏是课程中讨论过的（幻灯片第 38 页）信标请求过程。宏将开始查找任何可用的网络。接下来，你将把这个宏添加到 **main** 函数中去，开始发送信标请求（即，寻找附近的网络）。

☐ 打开文件 **main.c**。（**main.c** 是将调用 MiWi API 的应用程序）。

☐ 去掉 **LAB_1** 定义（**//#define LAB_1**）前面的注释标记

☐ 找到

```
!! ToDo: DiscoverNetworksHere
```

☐ 在此标记处插入宏 **DiscoverNetworks( )**。

接下来，你将插入代码，开始发送一则关联请求。

☐ 打开文件 **MiWi.c**。找到函数

```
void JoinNetwork(BYTE handle)
```

阅读函数说明，了解其工作原理。

☐ 打开文件 **main.c**。找到标记

!! ToDo: JoinNetworkHere

☐ 插入函数 **JoinNetwork( )**，函数要带有必须的句柄（提示：关于程序如何搜索具有最强信号的网络，请看前面的代码）。

☐ 编译程序（Make 或者 Build All）

> 注：由于代码中插入了“!!”，可能会提示你语法错误。把这些行简单地注释掉就行。

☐ 把 PICDEM Z 连接到 PC 的串口。用 9V 电源给 PICDEM Z 供电。把 ICD-2 连接到 ICSP 端口（RJ11 插孔）。

☐ 在 MPLAB IDE 中，选择 ICD-2 为编程器

☐ 对 PICDEM Z 进行编程

☐ 打开超级终端（中文系统：启动->程序->附件->通讯->超级终端，英文系统：Start->All Programs->Accessories->Communications->HyperTerminal）。将设定设置为 COM 端口（COM1 或 COM2）以及 19200-8-无-1-无。

☐ 从 RESET 中释放 ICD-2，然后在 PICDEM Z 中运行 MiWI 程序。

☐ 观察指令屏幕，你的 MiWi 节点已经加入网络。你在步骤 1 中键入的名字，应该已经显示在了屏幕上。

3. 发送一条消息（翻转 LED RA1）给对等节点

在这一步，你将发送一条消息（翻转 LED RA1）给你所选择的对等节点。指令屏幕上会显示已经加入网络的全部设备。

☐ 选择消息发送目的节点的独特地址（八字节 ASCII 名称）。

☐ 打开文件 **MiWi.c**。找到函数

```
BYTE SendReportByLongAddress(BYTE *pLongAddress)
```

阅读函数说明，了解函数的工作原理。你将使用此函数向所选节点发送消息。

☐ 打开文件 main.c。找到标记

!! TODO: finish LAB_1 second part here

如果按键 RB5 按下的话，程序将运行到此处。请阅读此标记处的注释。

☐ 在这里插入源代码，发送一条消息（翻转 LED）给目的节点。消息格式是

USER_REPORT_TYPE,
LIGHT_REPORT,
8 字节的源地址（你的地址，LSB 在前），
有效数据（LIGHT_TOGGLE）

使用 **WriteData( )**宏（在 **MiWi.h** 中），创建要发送的消息。使用函数 **SendReportByLongAddress(***目的节点的长地址***)** 来发送消息。

☐ 编译程序（Make 或 Build All）

☐ 对 PICDEM Z 进行编程

☐ 运行程序。按下按键 RB5，观察，目的节点上的 LED RA1 将翻转。如果你收到了消息，观察，LED RA1 将翻转，超级终端将显示消息以及源节点的地址。

给C编程经验较少的学员的实验指导：

1. 设置独特的长地址

☐ 启动 MPLAB IDE

☐ 在 MPLAB IDE 菜单中，选择 **Project > Open**，而后浏览 **C:\MASTERS\11052\MiW End Device\MiWi – End Device – PIC18 – C18.mcp**。

☐ 打开文件 **MiWiDefs.h**。找到从 **EUI_0** 一直到 **EUI_7** 的#define。这是设备特有的扩展地址。就本实验而言，只使用 ASCII 码（见附录 A）。键入地址，比如你的姓名或昵称之类。确认你的地址的独特性。例如：

```
#define EUI_0 'M'
#define EUI_1 'r'
#define EUI_2 '.'
#define EUI_3 'S'
#define EUI_4 'm'
#define EUI_5 'i'
#define EUI_6 't'
#define EUI_7 'h'
```

不要忘记删除注释，注释前以“!!”标记

注：在实际应用中，EUI_5 到 EUI_7 是从 IEEE 申请得到的唯一 OUI。

2. 加入网络

在最初打开节点的电源时，事件发生顺序如下：

A. 信标请求（已有网络存在吗？）
Yes——决定你是否要加入网络（要加入则发送**关联请求**）。
No ——如果你是 **PAN 协调器**，把节点升级成为网络的 **PAN 协调器**（即，建立新的网络）。否则，继续发送**信标请求**，直到你接收到**信标**为止。
B. 加入网络（发送**关联请求**）

接下来，你将添加代码，开始发送信标请求。

☐ 打开文件 **MiWi.h**。找到宏 **DiscoverNetworks( )**。这个宏是课程中讨论过的（幻灯片第 38 页）信标请求过程。宏将开始查找任何可用的网络。接下来，你将把这个宏添加到 **main** 函数中去，开始发送信标请求（即，寻找附近的网络）。

☐ 打开文件 **main.c**。（**main.c** 是将调用 MiWi API 的应用程序）。

☐ 去掉 **LAB_1** 定义（**//#define LAB_1**）前面的注释标记

☐ 找到

```
!! ToDo: DiscoverNetworksHere
```

☐ 在此标记处插入宏 **DiscoverNetworks( )**。

```
DiscoverNetworks();
```

接下来，你将插入代码，开始发送一则关联请求。

☐ 打开文件 **MiWi.c**。找到函数

```
void JoinNetwork(BYTE handle)
```

阅读函数说明，了解其工作原理。

☐ 打开文件 **main.c**。找到标记

```
!! ToDo: JoinNetworkHere
```

☐ 插入函数 **JoinNetwork( )**，函数要带有必须的句柄（提示：关于程序如何搜索具有最强信号的网络，请看前面的代码）。

```
JoinNetwork(handleOfBestNetwork);
```

☐ 编译程序（Make 或者 Build All）

注：由于代码中插入了“!!”，可能会提示你语法错误。把这些行简单地注释掉就行。

☐ 把 PICDEM Z 连接到 PC 的串口。用 9V 电源给 PICDEM Z 供电。把 ICD-2 连接到 ICSP 端口（RJ11 插孔）。

☐ 在 MPLAB IDE 中，选择 ICD-2 为编程器

☐ 对 PICDEM Z 进行编程

☐ 打开超级终端（中文系统：启动->程序->附件->通讯->超级终端，英文系统：Start->All Programs->Accessories->Communications->HyperTerminal）。把设定置为 COM 端口（COM1 或 COM2）以及 19200-8-无-1-无。

☐ 从 RESET 中释放 ICD-2，然后在 PICDEM Z 中运行 MiWI 程序。

☐ 观察指令屏幕，你的 MiWi 节点已经加入网络。你在步骤 1 中键入的名字，应该已经显示在了屏幕上。

3. 发送一条消息（翻转 LED RA1）给对等节点

在这一步，你将发送一条消息（翻转 LED RA1）给你所选择的对等节点。指令屏幕上会显示已经加入网络的全部设备。

☐ 选择消息发送目的节点的独特地址（八字节 ASCII 名称）。

☐ 打开文件 **MiWi.c**。找到函数

```
BYTE SendReportByLongAddress(BYTE *pLongAddress)
```

阅读函数说明，了解函数的工作原理。你将使用此函数向所选节点发送消息。

☐ 打开文件 main.c。找到标记

```
!! TODO: finish LAB_1 second part here
```

如果按键 RB5 按下的话，程序将运行到此处。请阅读此标记处的注释。

☐ 在这里插入源代码，发送一条消息（翻转 LED）给目的节点。消息格式是

USER_REPORT_TYPE,
LIGHT_REPORT,
8 字节的源地址（你的地址，LSB 在前），
有效数据（LIGHT_TOGGLE）

使用 **WriteData( )**宏（在 **MiWi.h** 中），创建要发送的消息。使用函数 **SendReportByLongAddress(***目的节点的长地址***)** 来发送消息。

```
WriteData(USER_REPORT_TYPE);
WriteData(LIGHT_REPORT);
for(i = 0; i < 8; i++)
{
   WriteData(myLongAddress[i]);
}
WriteData(LIGHT_TOGGLE);

tempLongAddress[0] = 'M'
tempLongAddress[1] = 'r'
tempLongAddress[2] = '.'
tempLongAddress[3] = 'S'
tempLongAddress[4] = 'm'
tempLongAddress[5] = 'i'
tempLongAddress[6] = 't'
tempLongAddress[7] = 'h'
SendReportByLongAddress(tempLongAddress);
```

☐ 编译程序（Make 或 Build All）

☐ 对 PICDEM Z 进行编程

☐ 运行程序。按下按键 RB5，观察，目的节点上的 LED RA1 将翻转。如果你收到了消息，观察，LED RA1 将翻转，超级终端将显示消息以及源节点的地址。

附录 A：可打印的 ASCII 代码

Binary	Dec	Hex	Glyph
010 0000	32	20	SP
010 0001	33	21	!
010 0010	34	22	"
010 0011	35	23	#
010 0100	36	24	$
010 0101	37	25	%
010 0110	38	26	&
010 0111	39	27	'
010 1000	40	28	(
010 1001	41	29	)
010 1010	42	2A	*
010 1011	43	2B	+
010 1100	44	2C	,
010 1101	45	2D	-
010 1110	46	2E	.
010 1111	47	2F	/
011 0000	48	30	0
011 0001	49	31	1
011 0010	50	32	2
011 0011	51	33	3
011 0100	52	34	4
011 0101	53	35	5
011 0110	54	36	6
011 0111	55	37	7
011 1000	56	38	8
011 1001	57	39	9
011 1010	58	3A	:
011 1011	59	3B	;
011 1100	60	3C	<
011 1101	61	3D	=
011 1110	62	3E	>
011 1111	63	3F	?

Binary	Dec	Hex	Glyph
100 0000	64	40	@
100 0001	65	41	A
100 0010	66	42	B
100 0011	67	43	C
100 0100	68	44	D
100 0101	69	45	E
100 0110	70	46	F
100 0111	71	47	G
100 1000	72	48	H
100 1001	73	49	I
100 1010	74	4A	J
100 1011	75	4B	K
100 1100	76	4C	L
100 1101	77	4D	M
100 1110	78	4E	N
100 1111	79	4F	O
101 0000	80	50	P
101 0001	81	51	Q
101 0010	82	52	R
101 0011	83	53	S
101 0100	84	54	T
101 0101	85	55	U
101 0110	86	56	V
101 0111	87	57	W
101 1000	88	58	X
101 1001	89	59	Y
101 1010	90	5A	Z
101 1011	91	5B	[
101 1100	92	5C	\
101 1101	93	5D	]
101 1110	94	5E	^
101 1111	95	5F	_

Binary	Dec	Hex	Glyph
110 0000	96	60	`
110 0001	97	61	a
110 0010	98	62	b
110 0011	99	63	c
110 0100	100	64	d
110 0101	101	65	e
110 0110	102	66	f
110 0111	103	67	g
110 1000	104	68	h
110 1001	105	69	i
110 1010	106	6A	j
110 1011	107	6B	k
110 1100	108	6C	l
110 1101	109	6D	m
110 1110	110	6E	n
110 1111	111	6F	o
111 0000	112	70	p
111 0001	113	71	q
111 0010	114	72	r
111 0011	115	73	s
111 0100	116	74	t
111 0101	117	75	u
111 0110	118	76	v
111 0111	119	77	w
111 1000	120	78	x
111 1001	121	79	y
111 1010	122	7A	z
111 1011	123	7B	{
111 1100	124	7C	|
111 1101	125	7D	}
111 1110	126	7E	~