应用笔记 155

1-WIRE软件资源指南和器件说明

摘要 : 包括*iButton*®器件在内，*Maxim*目前生产的*1-Wire*®器件已有*30*多种。如何选择已有的应用程序接口*API*、软件范例及其它资源与这一类器件进行通信或为某个器件选择正确的资源是一件非常令人头疼的事。本文提供了该类资源的概述和选择指南。另外，还提供了一个目前*1-Wire*器件的功能说明和家族代码列表，便于用户查询。

所提供的*API*包括：*TMEX (Microsoft Windows® API)*、*1-Wire*公共文件资源*(*一种跨平台*API)*、*Java™ 1-Wire API (OWAPI)*及其变形*.NET 1-Wire API (OW.NET)*和精简型*.NET 1-Wire API (OW.NET.Compact)*。本文所描述的所有*API*都是免费的，而且大多数情况下还包括完整的源代码。

引言

包括iButton在内，Maxim目前生产的1-Wire器件已有30多种。如何选择已有的应用程序接口(API)、软件范例及其它资源与这一类器件进行通信，或为某个器件选择正确的资源是一件非常令人头疼的事。本应用笔记提供了该类资源的概述和选择指南。本文中所描述的所有API都是免费的，而且大多数情况下还包括完整的源代码。

1-Wire概述

1-Wire总线是一种简单的信号交换架构，通过一条线路在主机与外围器件之间进行双向通信。所有的1-Wire总线都具有一个共同的特征：无论是芯片内还是iButton内，每个器件都有一个互不重复的、工厂光刻的序列号，因此，每个器件都是唯一的。这样就允许从众多连到同一总线的器件中独立选择任何一个器件。当1个、2个甚至多个1-Wire器件能共用一条线路进行通信，可以采用二进制位检索法依次查找每一个器件。一旦器件的序列号已知，通过寻址该序列号，就可以唯一地选出该器件进行通信。

所有通信的第一步都需要总线控制器发出一个复位信号以使总线同步，然后选择一个受控器件进行随后的通信，这可以通过选择所有的受控器件或者选择一个特定的受控器件(利用该器件的序列号进行选择)，或者通过对半检索法找到总线上的下一个受控器件来实现。上文所提到的这些指令都是网络指令或者只读存储器(ROM)指令。一旦一个特定的器件被选中，那么在下次复位信号发出之前，所有其它器件都被挂起而忽略随后的通信。

一旦一个器件被用于总线通信，主机就能向它发出特定的器件指令，对它进行数据读写。这是因为每类器件具有不同的功能和不同的用途，而且一旦器件被选定，就有了唯一的协议。虽然每类器件具有不同的协议和特征，但其工作过程却是相同的并且遵循如**图1**所示的工作流程。

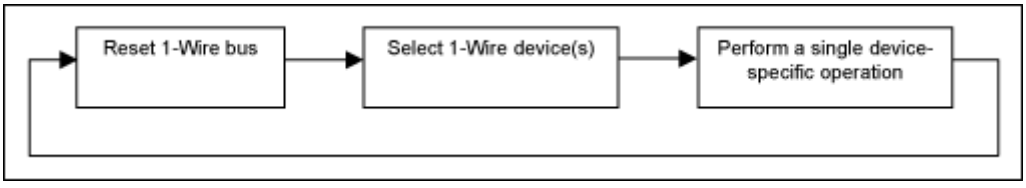

图*1.* 典型的*1-Wire*通信流程

每个受控器件的序列号的整数部分是一个8位的家族代码。这个代码对器件模型来说是特定的。因为每种器件模型执行不同的功能，所以可以用代码来选择用于控制或者查询器件的协议。**表1**是Maxim 1-Wire器件型号家族代码。

表1. 家族代码对照

家族代码	器件 () - iButton封装	说明 (除非特别说明，存储器尺寸以位为单位)
01 (hex)	(DS1990A), (DS1990R), DS2401, DS2411	仅1-Wire网络地址(注册码)
02	(DS1991)[1]	多密钥iButton，1152位安全存储器
04	(DS1994), DS2404	4Kb非易失RAM存储器及时钟、定时器和闹钟
05	DS2405[1]	单路可编址开关
06	(DS1993)	4Kb非易失RAM存储器
08	(DS1992)	1Kb非易失RAM存储器
09	(DS1982), DS2502	1Kb EPROM存储器
0A	(DS1995)	16Kb非易失RAM存储器
0B	(DS1985), DS2505	16Kb EPROM存储器
0C	(DS1996)	64Kb非易失RAM存储器
0F	(DS1986), DS2506	64Kb EPROM存储器
10	(DS1920)	带报警门限的温度检测
12	DS2406, DS2407[1]	1Kb EPROM存储器，2路可编址开关
14	(DS1971), DS2430A[1]	256位EEPROM存储器和64位OTP寄存器
1A	(DS1963L)[1]	带写周期计数器的4Kb非易失RAM存储器
1C	DS28E04-100	4096位 EEPROM存储器，2路可编址开关
1D	DS2423[1]	带外部计数器的4Kb非易失RAM存储器
1F	DS2409[1]	2通道可编址耦合器，用于子网络
20	DS2450	4通道A/D转换器(ADC)
21	(DS1921G), (DS1921H), (DS1921Z)	Thermochron®温度记录仪
23	(DS1973), DS2433	4Kb EEPROM存储器
24	(DS1904), DS2415	实时时钟(RTC)
27	DS2417	带中断的RTC
29	DS2408	8路可编址开关
2C	DS2890[1]	单路数字电位器
2D	(DS1972), DS2431	1024位1-Wire EEPROM
37	(DS1977)	密码保护的32KB (字节) EEPROM
3A	(DS2413)	2路可编址开关
41	(DS1922L), (DS1922T), (DS1923), DS2422	大容量Thermochron (温度)和Hygrochron™ (湿度)记录器
42	DS28EA00	可编程分辨率的数字温度计，具有顺序检测和PIO
43	DS28EC20	20Kb 1-Wire EEPROM

*该表没有列出Maxim的全部1-Wire器件(家族)，这些仅是Automatic Information事业部(BU)提供的软件库可直接支持的器件。

[1]不推荐在新的设计中使用这些器件。

API基础

1-Wire通信器件的不同应用程序接口(API)有着许多共性，反映了协议的基本数据通信原理。**图2**是根据不同API功能进行的分类。因为大多数1-Wire器件具有存储器，尽管存储器输入输出功能并不适用于所有的器件，我们还是把它们视为一个通用API集。其它所有不具备存储器专用功能的划分

为一类-专用器件集。

会话

独占使用*1-Wire*总线。这对于操作系统或几个进程或线程尝试同时使用同一总线的情况下是非常重要的。当多项操作在同一器件上运行而又不能被打断的时候，需要独占总线的使用权。

链路

基本的*1-Wire*总线通信功能。所有的1-Wire总线通信功能可以归结为：复位所有的器件和读写位。这也包括设置总线电特性的功能，如提供专用的EPROM编程脉冲或进行供电。

网络

查找和选择器件的网络功能。每个1-Wire器件都有一个固定的序列号，作为它的唯一的网络地址。这些功能可以通过链路层功能进行构造。在1-Wire器件数据资料中将其称为ROM指令，这是因为序列号是只读的。还有一些1-Wire控制器包括一些内置功能，因为它们比通过链路功能进行构造有效得多。

传输

块通信和基本的存储器读/写功能。这还包括存储器信息包读/写功能。这些功能是通过网络层和链路层功能构造的。

文件

使用1-Wire文件结构(参考应用笔记114：“1-Wire File Structure”)的文档存储层功能。这些功能是通过网络层和传输层功能构造的，且只对使用多页存储器的器件有效。

器件

专用器件的‘高层’功能。这些功能通常是通过网络、传输和链路类功能构造的，并执行诸如读取温度值或设置开关状态等操作。

图*2. API*功能集

图3概括了使用这些功能的典型顺序。SESSION功能围绕着调用器件进行通信，具有代表性的是先使用一个NETWORK功能，然后运行存储器或DEVICE的特定操作。

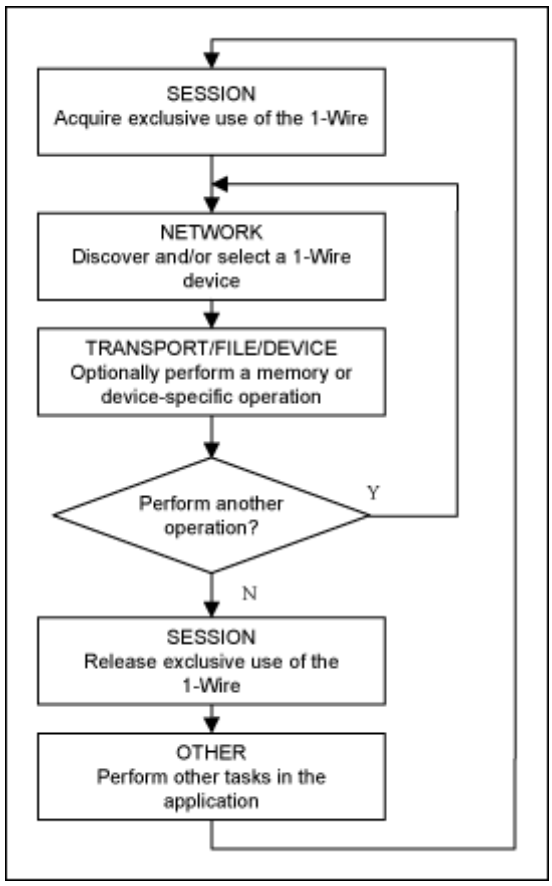

图*3. API*用法流程

iButton通信实质上是通过与其触头相'接触'来实现的。这意味着与器件的联系有时是不可靠的。iButton也可能被安装到阅读器里，在阅读的时候弹出，从而必须有一个相容的纠错方法紧跟其后。当检测到虚假错误时必须重发数据并在数据通信中进行CRC校验。API中的文件输入输出功能所利用的标准文件结构在应用笔记114："1-Wire File Structure"中的*1-Wire File Structure*部分进行了详细说明。这种结构在每页数据上都使用CRC16，以快速地校验所读数据的正确性。大多数1-Wire API功能很少或者不能自动重发。重发受应用软件控制，请参见应用笔记159："在iButton®应用中通过软件方法实现可靠的1-Wire®通信"，其中讲述了纠错方法和1-Wire通信的风险评估。

API的选择

本文主要列举了五种不同的应用程序接口(API)。这些API运行在不同的工作平台上，使用不同的语言，具有不同的性能。**表2**简要地描述了这五种API；**表3**说明了按语言划分的可利用API及其操作系统。

表2. API概述

API	缩写	说明
1-Wire公共文件资源	PD	源代码完全开放的C语言公共文件资源API，设计用于在多种PC操作系统、手持设备操作系统和微控制器平台之间移植。对于PC平台，通过本地驱动程序库在Microsoft Windows系统上支持所有的1-Wire适配器(主机)；利用跨平台库来支持其它PC操作系统上的特定1-Wire适配器(DS9097U串行适配器和DS9490 USB适配器)。
Java 1-WireAPI	OWAPI	完全公开的、高级Java API，支持几乎所有的1-Wire器件。除了支持本地1-Wire主机外，它还可通过跨平台库来支持DS9097U串行适配器和DS9490 USB适配器。
.NET 1-WireAPI	OW.NET	OWAPI代码基于Microsoft的.NET框架下的J#。
精简型.NET 1-WireAPI	OW.NET.Compact	用于Windows CE计算机或平台的精简型.NET框架不包含Microsoft Visual J#®发行套件，目前只包含与C#接口的底层1-Wire链路和网络层。
TMEX API	TMEX	在Windows平台(32位和64位)上支持所有的1-Wire主适配器。提供链接和文件输入输出功能，但不包括访问器件的功能。驱动程序是不公开的资源。这种API可以被其它的API调用来访问所有类型的1-Wire适配器。

表3. API操作系统和语言适用范围

语言	TMEX /OW.NET/OW.NET.Compact	C	Java
操作系统	(与Microsoft Windows语言无关)		
Windows Vista®	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows Vista x64	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows XP	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows XP x64	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows 2008	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI

Windows 2008 x64	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows 2000[1]	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows ME[1]	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows 98[1]	TMEX /OW.NET/**OW.NET.Compact**	PD	OWAPI
Windows 95[1]	TMEX	PD	OWAPI
Win3.1[1]	TMEX	PD	
DOS[1]	TMEX	PD	
Pocket PC/CE	OW.NET.Compact	PD	
Linux®和其它基于UNIX®的操作系统		PD	OWAPI
MxTNI*		PD - 无MxTNI操作系统	OWAPI

*MxTNI™是由Maxim研制的带有基于Java的操作系统的嵌入式平台。
[1]不再进行支持，之前的驱动程序仍可从Maxim网站下载。

API不同，器件族的支持亦不同。**表4**列出了目前所有可利用的1-Wire器件，并用标志说明它们受哪种API支持。标志的注释在表4的底部。注：表中无阴影的器件单元全部受API支持。浅色阴影的单元表示部分地受API支持，而深色阴影单元表示受API支持最少。

表4. 器件支持的API

器件	族号	说明	TMEX	PD	OWAPI	OW.NET	OW.NET Compact
DS1982	09	1Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS1985	0B	16Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS1986	0F	64Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS1904	24	RTC	AB	AB	ABI	ABI	AB
DS1920	10	带报警输出的温度传感器	AB	ABI	ABCI	ABCI	AB
DS1921G DS1921H DS1921Z	21	Thermochron温度记录器	ABDE	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS1922L DS1922T DS1923	41	高存储容量Thermochron (温度)和/或Hygrochron (湿度)记录器	AB	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS1963L[1]	1A	带写周期计数器的4Kb NV RAM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1971	14	256位EEPROM存储器和64位OTP寄存器	ABD	ABCDI	ABCDI	ABCDI	AB
DS1972	2D	1024位EEPROM存储器	AB	ABCDEI	ABCDEFGHI	ABCDEFGHI	
DS1973	23	4Kb EEPROM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1977	37	口令保护的32KB (字节) EEPROM	AB	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1990A DS1990R	01	只有1-Wire地址	AB	AB	AB	AB	AB

DS1991[1]	02	多密钥iButton, 1152位加密存储器	AB	ABC	ABC	ABC	AB
DS1992	08	1Kb NV RAM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1993	06	4Kb NV RAM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1994[1]	04	4Kb NV RAM存储器和时钟、定时器、闹钟	ABDE	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS1995	0A	16Kb NV RAM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS1996	0C	64Kb NV RAM存储器	ABDE	ABCDE	ABCDEFGH	ABCDEFGH	AB
DS2401	01	只有1-Wire地址	AB	AB	AB	AB	AB
DS2405[1]	05	单路开关	AB	ABI	ABI	ABI	AB
DS2404[1]	04	4Kb NV RAM存储器和时钟、定时器、闹钟	ABDE	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS2406 DS2407[1]	12	1Kb EPROM存储器，2路可编址开关	ABCE	ABCDEI	ABCDEI	ABCDEI	AB
DS2408	29	8路可编址开关	AB	ABI	ABI	ABI	AB
DS2409[1]	1F	双路开关，耦合器	AB	ABI	ABI	ABI	AB
DS2411	01	只有1-Wire地址	AB	AB	AB	AB	AB
DS2413	3A	双路可编址开关	AB	ABI	ABI	ABI	AB
DS2415	24	RTC	AB	AB	ABI	ABI	AB
DS2417	27	带中断的RTC	AB	AB	ABI	ABI	AB
DS2422	41	高存储容量Thermochron (温度)/Hygrochron (湿度)记录器	AB	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS2423[1]	1D	带外部计数器的4Kb NV RAM存储器	ABDE	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS2430A[1]	14	256位EEPROM存储器和64位OTP寄存器	ABD	ABCDI	ABCDI	ABCDI	AB
DS2431	2D	1024位EEPROM存储器	AB	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS2450	20	四路ADC	AB	ABI	ABI	ABI	AB
DS2502	09	1Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS2505	0B	16Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS2506	0F	64Kb EPROM存储器	ABCE	ABCDE	ABCDE	ABCDE	AB
DS2890[1]	2C	单路数字电位计	AB	AB	ABI	ABI	AB
DS28E04-100	1C	4096位EEPROM存储器、两路可编址开关	AB	ABCDEI	ABCDEFGHI	ABCDEFGHI	AB
DS28EA00	42	分辨率可编程的数字温度计，带有顺序检测和PIO	AB	AB	AB	AB	AB
DS28EC20	43	20Kb EEPROM	AB	AB	AB	AB	AB

阴影标志	支持标志说明
全部支持	A. 支持1-Wire简单连接 B. 支持1-Wire网络

部分支持	C. 支持存储器的字节读写 D. 支持存储器的包读写 E.支持AA型1-Wire文件结构(关于文件结构类型的信息，请参见应用笔记114："1-Wire File Structure")
很少支持	F. 支持AB型1-Wire文件结构 G. 支持BA型1-Wire文件结构 H. 支持BB型1-Wire文件结构 I. 支持其它专用器件

[1]不推荐在新的设计中使用这些器件。

1-Wire公共文件(PD)资源概述

1-Wire PD API提供的功能都是用C语言写的，用于TMEX API所不支持的平台。'1-Wire网络'(或者MicroLAN™)是一种在主机和一个或多个外设之间只有一条数据线和地线的网络。这个API创建了用于识别从机器件、并与从机通信的1-Wire主机。它提供所有的1-Wire、传输和文件操作类服务，可以与包括iButton在内的1-Wire器件进行通信。该类API软件包和平台示例可从iButton网站下载。

用'C'语言设计的API是可移植的。提供专用平台上完成的'TODO'模板，也提供了几种平台范例的实现：包括64位Windows、32位Windows和Linux。还有用这些平台实现的几个应用。

有三类可移植的源文件：分别定义为'通用'、'userial'和'其它'。第一类是通用源文件，适用于已经与1-Wire通信功能建立了基本连接的平台(通用)，是与硬件有关的最底层源文件。第二类是假设用户已经有了一个串口(RS-232)，并请求使用'通用串行1-Wire总线驱动程序：DS2480B' (userial)。这种芯片从串口读入命令后，执行1-Wire操作，然后再把结果送回串口。源代码把预期的1-Wire操作转换为串行通信包传送给DS2480B。唯一需要提供给平台的参数就是串口的读写初始值。所有DS9097U系列的串行适配器的接口芯片用的都是DS2480B。最后，第三类是处理特定1-Wire适配器功能的可移植的源文件和/或不属于上述前两类的源文件(其它)。比如用来处理USB端口例子，是特定利用DS2490 USB至1-Wire桥接芯片。多数情况下，这一类源文件很像通用类的源文件，只不过已经过特定DS2490的改进。不管所用到的是哪类源文件(通用、userial或其它)，最终，同一API是要公布给软件开发者的。

值得注意的是，这篇应用笔记还提供完全开放源代码、1-Wire公用工具包的跨平台libusb构件(归在'其它'源文件类里)。Libusb是源代码完全开放的USB库，可在多种不同操作系统间移植。1-WirePD API的特定源文件在必要的libusb功能中编译，然后用于构建1-Wire PD API 。

有关可用的各类文件可参考下表，表中的这些文件均已在每个平台上建立，并提供下载。

表5. 1-Wire文件类别和预置二进制

可移植的源文件类别(构件)	平台	端口	说明
userial	Win64, Win32, Linux, (其它UNIX), DS550	COM	支持DS9097U 1-Wire串行接口适配器和其它基于DS2480B的方案(包括嵌入的)。
	Win64, Win32	LPT	LPT构件支持Windows上的DS1410E[1]并行接口适配器。
通用	DS550	微处理器(µP)端口引脚	DS550通用源文件用到了µP端口引脚。

其它'libusb'	Win64, Win32, Linux, Macintosh, (其它UNIX)	USB	支持DS9490 USB 1-Wire适配器和其它任何基于DS2490的USB解决方案，特定的安装了对应的libusb驱动程序。
其它'WinUsb'	Win64, Win32	USB	支持DS9490 USB 1-Wire适配器和安装了WinUSB器件驱动程序的Windows下的其它任何基于DS2490的USB适配器。
其它'wrapper' (TMEX)	Win32	USB, COM, LPT	捆绑了TMEX API，可在Windows下提供多端口支持(分别对应DS9490、DS9097U和DS1410E[1])。
其它'multiport'	Win64, Win32	USB, COM, LPT	通过直接与最底层的本地Windows驱动程序进行通信，来支持多端口。

注：关于最新的平台源文件可从1-Wire公用工具包网站获得。
[1]不推荐在新设计中使用DS1410E并行端口适配器。

这些不同类的可移植源代码文件执行同样的1-Wire API，而且可以互换。**图4**所示API为可用于1-Wire公用代码基础的第3.xx版。要注意的是考虑到非存储器器件的特定功能数量过多，所以未详细列出。图4列出了为新平台提供的功能和所需模块的源文件。

除了能够下载可移植的公用套件'C'语言模块外，还可以下载有限数量的微处理器(µP)汇编实例，进行1-Wire通信。

该API中的文件功能实现了1-Wire的'AA'类文件结构，应用笔记114："1-Wire File Structure "定义了这种文件结构。

顾名思义，API所提供的源代码有一个尽可能接近公用文件资源的许可证。开发者可以自由使用，并可与他们的应用软件不受任何限制地结合为一体。

会话

owAcquire - 获得1-Wire网络的使用权。
owRelease - 释放以前得到的1-Wire网络使用权。

链路

owHasOverDrive - 说明适配器是否具有快速模式适配性能。
owHasPowerDelivery - 说明适配器是否能供电。
owHasProgramPulse - 说明EPROM编程电压是否可用。
owLevel - 设置1-Wire传输线上某点电平为标准模式(5V弱上拉)、供电(5V强上拉)、或者编程电平(12V EPROM编程电平)。
owProgramPulse - 发同步编程脉冲写EPROM 1-Wire器件。
owReadBitPower - 读取1位数据，并有选择地提供供电。
owReadByte - 通过发送全1 (0xFF)接收来自1-Wire网络的8位数据。
owSpeed - 设置1-Wire网络的速度为标准模式(16kb)或快速模式(142kb)。
owTouchBit - 发送并接收1位1-Wire网络的数据。
owTouchByte - 发送并接收8位1-Wire网络的数据。
owTouchReset - 复位1-Wire网络的所有器件并返回结果。
owWriteByte - 向1-Wire网络发送8位数据并验证收到的应答是否匹配。
owWriteBytePower - 向1-Wire网络发送8位通信数据然后供电。

网络

owAccess - 选定当前器件并准备运行特定器件的指令。
owFamilySearchSetup - 确定随后的搜索顺序(*owNext*)以找到特定的家族类型。
owFirst - 搜索并找到1-Wire网络上的第一个1-Wire器件。
owNext - 搜索并找到1-Wire网络上的下一个1-Wire器件。
owOverdriveAccess - 选择当前的器件，并设置它的速度为快速模式。
owSerialNum - 保持或设定当前选定的器件的序列号(ROM号)。
owSkipFamily - 略过最后一次查询到的家族类型对应的所有1-Wire器件。
owVerify - 选择并验证当前的器件正在运行(报警选项)。

传输

owBlock - 发送和接收1-Wire网络上的一个数据块，并有选择地进行复位。
owCanLockPage - 检查给定的存储区是否有可被锁定的页。
owCanLockRedirectPage - 检查给定的存储区是否有在重定向中能够被锁定的页。
owGetAlternateName - 获得改变的器件型号或器件名。
owGetBankDescription - 获得存储区的字符性描述。
owGetDescription - 获得对1-Wire器件类型的简单描述。
owGetExtraInfoDesc - 获得对附加信息的内容描述。
owGetExtraInfoLength - 获得存储区中附加信息的长度，单位为字节。
owGetMaxPacketDataLength - 获得一个数据包中最大的数据长度，单位为字节。
owGetName - 以字符串的形式获得1-Wire器件的型号。
owGetNumberBanks - 获得某一1-Wire家族类的存储区号。
owGetNumberPages - 获得给定存储区的页数。
owGetPageLength - 获得给定存储区的原始页长度，单位是字节。
owGetSize - 获得给定存储区的大小，单位是字节。
owGetStartingAddress - 获得给定存储区的物理首地址。
owHasExtraInfo - 检测读数据时存储区的页是否传送附加信息。
owHasPageAutoCRC - 检测读数据页时存储区是否有器件产生的CRC校验码。
owIsGeneralPurposeMemory - 检测存储区是否是通用用户存储器。
owIsNonvolatile - 检测当前的存储区是否是非易失的。
owIsReadOnly - 检测存储区是否是只读的。
owIsReadWrite - 检测存储区是否是可读写的。
owIsWriteOnce - 检测存储区是否是只可写入一次，如EPROM。
owNeedsPowerDelivery - 检测写存储区时是否需要‘供电’。
owNeedsProgramPulse - 检测写存储区时是否需要‘编程脉冲’。
owRead - 读初始状态下存储区的一部分(没有包，CRC校验码)。
owReadPage - 读初始状态下存储区的完整的一页(没有包，CRC校验码)。
owReadPageCRC - 在器件产生CRC校验码的状态下，读存储区的完整的一页。
owReadPageExtra - 读初始状态下包括所有‘附加’信息在内的存储区的完整一页(没有包，CRC码)。
owReadPageExtraCRC - 读取包括所有‘附加’信息和器件生成的CRC码在内的存储区的完整的一页。
owReadPagePacket - 从存储区的页中读取通用数据包(有关通用数据包结构的说明，参见应用笔记114："1-Wire File Structure")。
owReadPagePacketExtra - 从带有‘附加’信息的存储区页中读取通用数据包。
owRedirectPage - 检测存储区中是否有能被重定向的页。
owWrite - 写初始模式下的一部分存储区。
owWritePagePacket - 把一个通用数据包写入存储区的一页中。

文件

owAttribute - 改变文件的属性。
owChangeDirectory - 改变当前目录。
owCloseFile - 关闭一个文件。
owCreateDir - 创建一个目录。
owCreateFile - 创建一个可写文件。
owCreateProgramJob - 创建一个写缓冲器，以记录EPROM编程未完成的任务。
owDeleteFile - 删除一个文件。
owDoProgramJob - 写未完成的EPROM编程任务。
owFirstFile - 寻找当前目录下的第一个文件。
owFormat - 格式化1-Wire文件结构文件系统。
owGetCurrentDir - 获取当前目录。
owNextFile - 寻找当前目录的下一个文件。
owOpenFile - 打开一个可读文件。
owReadFile - 读一个已打开的文件。
owReadFile - 从文件读取数据。
owRemoveDir - 删除一个目录。
owReNameFile - 更改文件名。
owWriteFile - 写一个已创建的文件。

器件

DoAtoDConversion - 在DS2450上进行A/D转换。
ReadSwitch12 - 读取DS2406开关的状态。
ReadCounter - 读取DS2423 1-Wire芯片上特定存储器页相关的计数器值。
…(由于专用器件的功能太多，这里不能一一列举)。
图*4. PD API*函数

下面的**例1**就是一段PD编码示例，它遵循图3所概括的API使用流程。简便起见，在程序的每次循环中找到一个1-Wire网络上的器件。更复杂的应用一次可能找到一类器件或者可能选定一个在上次查找过程中找到的器件。

```
int rslt, portnum=0, doing_work=1;
char portString[50]; // set to platform appropriate port string

// work loop
while (doing_work)
{
  // acquire the 1-Wire Net (SESSION)
  if (owAcquire(portnum, portString))
  {
   // find all devices (NETWORK)
   rslt = owFirst(portnum, TRUE, FALSE);
   while (rslt)
   {
     // do SOMETHING with device found (TRANSPORT/FILE/DEVICE)
     // . . .

     // find the next device (NETWORK)
     rslt = owNext(portnum, TRUE, FALSE);
   }

   // release the 1-Wire Net (SESSION)
   owRelease(portnum);
  }
  else
  {
   // Could not acquire 1-Wire network
   // . . .
  }

  // do other application work
  // . . .
}
```

例*1. PD*编码示例

图5a和**5b**列出了C语言模块，它们分别构成了1-Wire的两个公用程序库。同时也列出了必需的‘TODO’功能，以把程序库传送到新平台。在这个软件包里也有执行‘TODO’功能的几个平台链接文件的例子。

会话

owsesu.c

链路

owllu.c ds2480ut.c ds2480.h

网络

ownetu.c crcutil.c (需要编译)

传输

mbappreg.c	mbappreg.h	mbee.c	mbee.h	mbee77.c	mbee77.h
mbeewp.c	mbeewp.h	mbeprom.c	mbeprom.h	mbnv.c	mbnv.h
mbnvcrc.c	mbnvcrc.h	mbscr.c	mbscr.h	mbscrcrc.c	mbscrcrc.h
mbscree.c	mbscree.h	mbscrex.c	mbscrex.h	mbscrx77.c	mbscrx77.h
mbsha.c	mbsha.h	mbshaee.c	mbshaee.h	owtrnu.c	pw77.c
pw77.h	rawmem.c	rawmem.h			

文件

owcache.c	owfile.c	owfile.h	owpgrw.c	owprgm.c	

器件

ad26.c	ad26.h	atod20.c	atod26.c	atod26.h	cnt1d.c
humutil.c	humutil.h	jib96.c	jib96.h	jib96o.c	ps02.c
ps02.h	sha18.c	sha33.c	shadbtvm.c	shadebit.c	shaib.c
shaib.h	swt05.c	swt12.c	swt12.h	swt1c.c	swt1c.h
swt1f.c	swt29.c	swt29.h	swt3a.c	swt3a.h	temp10.c
thermo21.c	hermo21.h	time04.c	time04.h	weather.c	weather.h

其它功能

ioutil.c	owerr.c	findtype.c	ownet.h	screenio.c	sprintf.c
crcutil.c					

TODO

提供一个可以运行以下函数的串口模块：
*BreakCOM** - 在串口上发送一个至少持续2ms的‘BREAK’指令。
CloseCOM - 关闭以前打开的串口(只对一些平台适用)。
*FlushCOM** - 允许任意未完成的写操作执行完毕，并清空输入缓冲器。
*msDelay** - 至少延迟指定的ms数。
msGettick - 返回一个增量ms计数器(只对一些示例适用)。
OpenCOM - 打开指定的串口以进行通信(只对一些平台适用)。
*ReadCOM** - 从串口读取指定数目的字节。
SetCOMBaud - 更改串口的波特率为指定值(也可设置为快速模式)。
*WriteCOM** - 写指定数目的字节到串口。
*基本操作所必需的功能。
图*5a. PD ‘USERIAL*执行程序

会话

(参见TODO)

链路

(参见TODO)

网络

ownet.c　crcutil.c (需要编译)

传输

除‘owtrnu.c’换为‘owtran.c’外，其余的均与USERIAL执行程序相同。

文件

与USERIAL执行程序相同。

器件

与USERIAL执行程序相同。

其它功能

与USERIAL执行程序相同。

TODO

提供可以运行以下函数的链接和会话接口：
owAcquire - 获得1-Wire网络。
owRelease - 释放以前得到的1-Wire网络。
owHasOverDrive - 说明适配器是否具有快速模式适配性能。
owHasPowerDelivery - 说明适配器是否能供电。
owHasProgramPulse - 说明EPROM编程电平是否可用。
owLevel - 设置传输线上某点电平为标准模式(5V弱上拉)，供电(5V强上拉)，或者为编程电平(12V EPROM编程电平)。
owProgramPulse - 发送同步编程脉冲写EPROM 1-Wire器件。
owReadBitPower - 读取1位数据，然后有选择地提供供电。
owReadByte - 通过发送全1信号(0xFF)，接收来自1-Wire网络的8位数据。
owSpeed - 设置1-Wire网络的速度为标准模式(16kb)或者快速模式(142kb)。
*owTouchBit** - 发送并接收一位1-Wire网络的数据。
owTouchByte - 发送并接收8位1-Wire网络的数据。
*owTouchReset** - 复位1-Wire网络上的所有器件并返回结果。
owWriteByte - 向1-Wire网络发送8位数据，并验证响应是否匹配。
owWriteBytePower - 向1-Wire网络发送8位通信数据，并提供供电。
*执行基本操作所必须的功能。
图*5b. PD 'GENERAL*执行程序

安装

由于1-Wire PD API是一套C模块，所以没有正规的安装程序。正如示例构件那样，所需模块直接编译到应用程序。当然这并不排除开发者把模块组合到可加载的程序库里，比如Windows动态链接库。

一些构件需要本地1-Wire适配器驱动程序或等效程序。因为多数操作系统平台具有内置的串口驱动程序，1-Wire PD API的'userial'源文件无需任何其它驱动程序。但对于支持USB和并口的1-Wire适配器，就需要安装本地或跨平台驱动程序。参考对应的驱动程序，这些驱动程序可从网上以现行的1-Wire公用程序工具包构件文件格式下载。

Java的1-Wire API (OWAPI)概述

为创建1-Wire在Java中的应用，适用于Java的1-Wire API从一开始就被设计成一种高度面向对象的基础牢靠的API。它增强了程序员设计可移植的交互平台软件的能力，并缩短了基于1-Wire器件设计开发的产品进入市场的时间。

API由很多Java类和接口构成。1-Wire API的Java类中有一个特殊的库：容器(OneWireContainer类)。容器支持包括iButton在内的特殊1-Wire器件。API有30多种容器类型，代表了大多数1-Wire器件。每个容器都可以压缩和执行单一器件的功能。

容器通过1-Wire物理适配器(DSPortAdapter类)和1-Wire器件进行通信。适配器由提供者类(OneWireAccessProvider类)生产。1-Wire适配器由于平台的不同而不同，但是它们都有同样的接口。有一些平台使用本地适配器，但大多数都至少支持通过使用RXTX (一种跨平台的串行COM端口API)来实现的DS9097U-XXX串行适配器。这种API可以从RXTX网站找到。

Java软件开发包1-Wire API可从iButton网站上找到。与1-Wire公共文件资源工具包一样，通过公共文件资源许可证也可以得到OWAPI的完整的Java源程序。

图6所示为API对象生成的典型顺序。'提供者'提供一个'适配器'实例(或枚举)，反过来它又生成器

件‘容器’实例。然后与器件的通信几乎直接通过容器来进行。

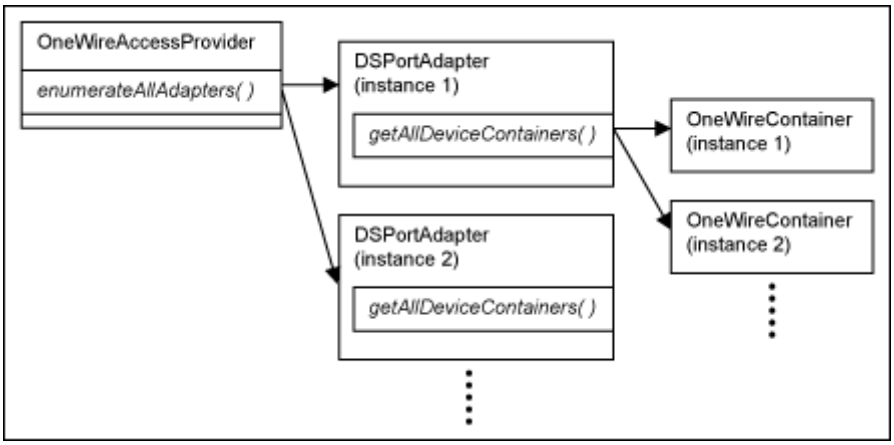

图*6. OWAPI*对象生成

图7说明了容器的共同特征。具有存储器的器件将为每个存储区生成一个存储区实例。根据特征可以把存储器划分成几组。比如：一组存储器是易失的，而另一组却是非易失的，或者一个区域能够作通用存储器，或能够进行存储器映射，以改变器件功能。

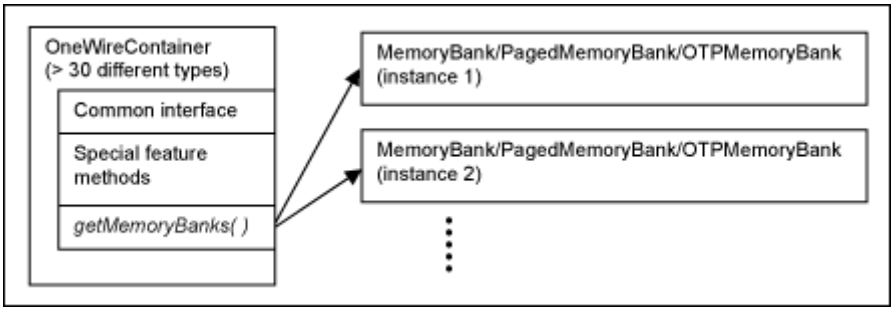

图*7. OWAPI OneWireContainer*特征

图8说明了基于OWAPI类库的方法。类或包用粗体字表示。注：因为每个容器都有操作器件类型的高级方法，所以一般不直接访问适配器中链路层的方法。

会话

com.dalsemi.onewire.adapter.DSPortAdapter
beginExclusive - 获取1-Wire网络的独占使用权。
endExclusive - 释放对1-Wire网络的独占使用权。

链路

com.dalsemi.onewire.adapter.DSPortAdapter
canBreak - 检测适配器是否支持1-Wire的‘break’ (长时间的低电平)操作。
canDeliverPower - 检测适配器是否支持‘强上拉’的供电。
canDeliverSmartPower - 检测适配器是否支持‘智能’供电。所谓‘智能’供电指的是能够检测何时能量消耗开始减少并能自动停止供电。
canFlex - 检测适配器是否支持远距离同步通信。
canHyperdrive - 检测适配器是否支持超光速通信。
canOverdrive - 检测适配器是否支持快速模式通信。
canProgram - 检测适配器是否支持12V的EPROM编程电压。
dataBlock - 发送和接收1-Wire网络的一个数据块。
getBit - 从1-Wire网络读取一位数据。
getBlock - 通过发送全1 (0xFF)信号从1-Wire网络读取数据块。
getByte - 通过发送全1 (0xFF)信号从1-Wire网络读取一个字节。
getSpeed - 获取当前1-Wire网络的通信速度。
putBit - 向1-Wire网络写一位数据。
putByte - 向1-Wire网络写一个字节数据，并验证响应是否正确。
reset - 复位所有1-Wire网络的器件。
setPowerDuration - 设定供电持续时间。
setPowerNormal - 停止供电。
setProgramPulseDuration - 设定编程电平持续时间。
setSpeed - 设定1-Wire网络通信速度。
startBreak - 在1-Wire网络上开始一次中断(低电平)。
startPowerDelivery - 开始供电。
startProgramPulse - 启动编程脉冲。

网络

com.dalsemi.onewire.adapter.DSPortAdapter
excludeFamily - 在搜索过程中把一个家族都排除在外。
findFirstDevice - 不经过容器的自动生成而找到1-Wire网络的第一个器件。
findNextDevice - 不经过容器的自动生成而找到1-Wire网络上的下一个器件。
getAllDeviceContainers - 找出1-Wire网络上所有具有容器的器件。
getDeviceContainer - 获取当前找到的器件的容器。
getFirstDeviceContainer - 找到第一个器件，并为之建立一个容器。
getNextDeviceContainer - 找到下一个器件，并为之建立一个容器。
setNoResetSearch - 设置1-Wire网络搜索未被复位的器件。
setSearchAllDevices - 设置1-Wire网络搜索所有器件(除了报警器件)。
setSearchOnlyAlarmingDevices - 设置1-Wire网络使其只搜索报警器件。
targetAllFamilies - 设置1-Wire网络搜索所有器件(除被排除在外的器件)。
targetFamily - 在1-Wire网络的搜索过程中，只搜索某一特定的家族器件
(也在**com.dalsemi.onewire.container***内)。
isAlarming - 检测器件是否处于报警状态。
isPresent - 检测器件是否在1-Wire网络上。
select - 选择1-Wire网络器件，准备执行器件的特定操作指令。

传输

com.dalsemi.onewire.container.MemoryBank
getBankDescription - 返回存储区的文本描述。
getSize - 获取存储区的大小，单位为字节。
getStartPhysicalAddress - 获得存储区的物理首地址。
isGeneralPurposeMemory - 检测存储区是否是通用的(不是存储器映射的)。
isNonVolatile - 检测存储区是否是非易失的。
isReadOnly - 检测存储区是否是只读的。
isReadWrite - 检测存储区是否是可读可写的。
isWriteOnce - 检测存储区是否像EPROM一样只能写入一次。
needsPowerDelivery - 检测存储区写时是否需要供电。
needsProgramPulse - 检测存储区写时是否需要编程脉冲。
read - 不译码直接读取存储区(没有包结构)。
setWriteVerification - 设置API写数据后进行校验。
write - 写存储区(没有包结构)。
com.dalsemi.onewire.container.PagedMemoryBank
getExtraInfoDescription - 获取和存储区有关的附加信息说明。
getExtraInfoLength - 获取每一页上附加信息的长度，单位为字节。
getMaxPacketDataLength - 获取包结构所能容纳的最大数据长度，以适合存储区的每一页。
getNumberPages - 获取存储区的页数。
getPageLength - 获取存储区空白页的长度，单位为字节。
hasExtraInfo - 检测存储区是否有与每一页有关的附加信息。
hasPageAutoCRC - 检测存储区中的页是否有器件产生的CRC校验码。
readPage - 从存储区中读取一页。
readPageCRC - 利用器件产生的CRC校验码从存储区读取一页数据。
readPagePacket - 从存储区的一页中读取包结构。
writePagePacket - 写包结构到存储区的页中。
com.dalsemi.onewire.container.OTPMemoryBank
canLockPage - 检测存储区的页是否可以写保护。
canLockRedirectPage - 检测存储区的重定向器件是否可以被锁，以防再次重定向。
canRedirectPage - 检测存储区是否有被重定向的页，以此更改一次性写入的页。
getRedirectedPage - 获取被重定向的页的数目。
isPageLocked - 检测页是否加了写保护。
isRedirectPageLocked - 检测页是否被锁，以防被重定向。
lockPage - 锁定一页。
lockRedirectPage - 锁定页以防被重定向。
redirectPage - 重定向一页到新页，更新一次性写器件时使用。

com.dalsemi.onewire.utils.OWFile
除了与java.io.File (JDK的版本1.2)相同的方法以外，还有下面的方法：
close - 关闭文件和释放所有相关的资源。
format - 格式化提供给OWFile的与器件有关联的1-Wire文件系统。
getFD - 获得文件的OWFile描述，以使文件操作与器件同步。
getFreeMemory - 获得1-Wire文件系统中可利用的剩余存储空间。
getLocalPage - 从1-Wire文件系统页获得存储区的当前页。
getMemoryBankForPage - 获取可以用来读写1-Wire文件系统页的存储区实例。
getOneWireContainer - 获得构成文件系统的容器。
getPageList - 获得含有文件的1-Wire文件系统列表。
com.dalsemi.onewire.utils.OWFileDescriptor
同样的方法见java.io.FileDescriptor (JDK的版本1.2)。
com.dalsemi.onewire.utils.OWFileOutputStream
同样的方法见java.io.FileOutputStream (JDK的版本1.2)。
com.dalsemi.onewire.utils.OWFileInputStream
同样的方法见java.io.FileInputStream (JDK的版本1.2)。

器件

com.dalsemi.onewire.container.*
包括六种不同的传感器类接口，共有30多种器件专用容器执行程序：
ADContainer - 模数转换器
ClockContainer - 时钟
SwitchContainer - 开关
TemperatureContainer - 温度传感器
PotentiometerContainer - 数字电位器
HumidityContainer - 湿度传感器
MissionContainer - 用于特定任务的温度和湿度记录器
OneWireSensor - 1-Wire传感器
PasswordContainer - 受口令保护的存储器
com.dalsemi.onewire.application.*
安全散列算法(SHA)和1-Wire标志实用程序类
*执行基本操作所必需的函数。
图*8. OWAPI*函数

下面的**例2**是OWAPI的代码段，它也遵循图3所概括的API使用流程。和例1中的公用代码实例一样，每次循环只找出一个1-Wire网络的器件；复杂点的应用程序一次循环中可以找出一类器件或是上次循环中找到的器件。

```
boolean doing_work=true;

// get the default adapter from the service provider
DSPortAdapter adapter = OneWireAccessProvider.getDefaultAdapter();

// work loop
while (doing_work)
{
  // get exclusive use of adapter (SESSION)
  adapter.beginExclusive(true);

  // clear any previous search restrictions (NETWORK)
  adapter.setSearchAllDevices();
  adapter.targetAllFamilies();
  adapter.setSpeed(adapter.SPEED_REGULAR);

  // enumerate through all the 1-Wire devices found (NETWORK)
  for (Enumeration owd_enum = adapter.getAllDeviceContainers();
    owd_enum.hasMoreElements(); )
  {
   // get a 'container' for each device
   OneWireContainer owd = ( OneWireContainer ) owd_enum.nextElement();

   // do SOMETHING with device found (TRANSPORT/FILE/DEVICE)
   // . . .
  }

  // end exclusive use of adapter (SESSION)
  adapter.endExclusive();

  // do other application work
  // . . .
}
```

例*2. OWAPI*编码实例

1-Wire标记

随着1-Wire传感器数量和种类的增多，1-Wire网络的管理也越来越困难。比如，ADC可以测量各种不同的值，所以给这种检测器加上标签说明它的功能也很重要。一种用XML标记1-Wire的方案已经确定并应用于Java的1-Wire API。这些标志允许应用程序动态的载入和设置传感器并赋给它一定的环境。详细内容可参考应用笔记158："用XML实现1-Wire标签"。

安装

图8所提到的调用API所需的任何模块都包含在一个jar文件里：OneWireAPI.jar。将模块导入正确的位置或类路径可提供全部的API。但有两个特例：一个是本地或通信API需要安装在特定的平台上，另一个是平台有容量的限制以致不能满足所有API的需要。这两个特例将在OWAPI软件包里进行详细的说明。

1-Wire .NET (OW.NET)概述

处于全面考虑，我们提供的.NET支持是对应Java的1-Wire API，仅用Microsoft的J#语言编译。所有的.NET 1-Wire应用程序仅需参考OneWire.NET.dll。这包括支持最新的.NET语言，如：C#、J#和VB.NET。有关1-Wire .NET的实例请参考iButton网站的软件开发包(SDK)下载。

注意：OneWire.NET.dll还需要在1-Wire应用程序正常工作前将以下发行程序安装在PC上：

1. 本地1-Wire端口适配器驱动程序。该驱动程序是1-Wire驱动程序，且可从iButton网站下载。
2. Microsoft .NET 2.0框架。
3. Visual J# .NET 2.0发行版本。

对于这些期望精简的.NET框架1-Wire可支持的器件，比如：Pocket PC、掌上电脑(PDA)、手机和机顶盒，OneWire.NET.dll的一个新链路层(仅仅)版本可供使用。这段程序完全用C#编写的，目前，没有OneWireContainers与之对应，但是有DSPortAdapter类可供使用。更多详细信息，请参阅针对Windows的1-Wire SDK。

1-Wire驱动包提供了OW.NET API及TMEX API。关于1-Wire驱动定制安装的详细信息，请参考应用笔记1740，"White Paper 6: 1-Wire Drivers Installation Guide for Windows"。

精简型1-Wire .NET (OW.NET Compact)概述

该对象的接口与1-Wire .NET (OW.NET)相同，但只使用了会话、链路和网络层。该对象没有正式的发行安装版本，而是采用单个动态连接库(OneWireLinkLayer.dll)的形式，直接与32位和64位Windows平台上的TMEX API通信。动态连接库文件可以包含在2.0 .NET应用中，以支持低级1-Wire。该对象用C#语言编写，无需J# 2.0 .NET发行包。

TMEX API (TMEX)概述

TMEX API是一套独立于Windows动态链接库的语言，它提供所有1-Wire器件的基本功能，包括用以支持存储器件的有限的1-Wire文件结构。这种API被设计为在多进程、多线程程序争用同一或不同1-Wire端口时工作。它可以支持16种不同的1-Wire适配器，每种适配器又有16个截然不同的端口。它支持Maxim生产的所有1-Wire适配器。

这种API的最底层用作Windows平台(1-Wire驱动程序)下Java 1-Wire API的本地驱动程序。由于1-Wire .NET API同样基于Java的1-Wire API，因此也被使用。**图9**给出了其它API是如何利用TMEX API提供的本地支持的。图中也包括实际驱动程序的文件名和它们是如何分层的。

1-Wire驱动器安装包(包括TMEX API、OW.NET库)和包含了许多链接到TMEX (以及配合API)的程序实例的软件开发工具在Maxim网站上均可找到。

TMEX SDK所提供的例子都有源代码，但还未提供低级驱动程序的源代码。驱动程序可自由发布。

表6出了目前支持的1-Wire适配器以及每种适配器的特征。

表6. TMEX所支持的适配器

适配器	端口	特征
DS9490R, DS9490B	USB	供电快速模式RJ-11或iButton座DS2401 ID
DS1410E[1]	并口	供电快速模式RJ-11或双iButton座DS2401 ID
DS1410D[1]	并口继承	双iButton DS2401 ID
DS9097U-009	串口	供电快速模式RJ-11连接器DS2502 ID
DS9097U-S09	串口	供电快速模式RJ-11连接器
DS9097U-E25	串口	供电快速模式RJ-11连接器可写EPROM
DS1411	串口	供电快速模式单iButton座

DS9097E[1]	串口继承	RJ-11连接器可写EPROM
DS9097[1]	串口继承	RJ-11连接器
DS1413[1]	串口继承	单iButton座

[1]不推荐在新设计中使用这些适配器。

TMEX支持的Windows平台

TMEX支持以下Microsoft Windows平台：Windows 2008、Windows 2003、Windows Vista以及Windows XP SP2，包含x86 (32位)和x64 (64位)两种操作系统。请注意，如果需要使用更早版本的Windows操作系统的驱动程序，Maxim网站还提供有TMEX继承版本(但不是主要支持的版本)。请下载4.00或更低版本的1-Wire驱动程序(将安装TMEX API库)。1-Wire驱动安装包可以从iButton：针对Windows的1-Wire驱动程序获得。

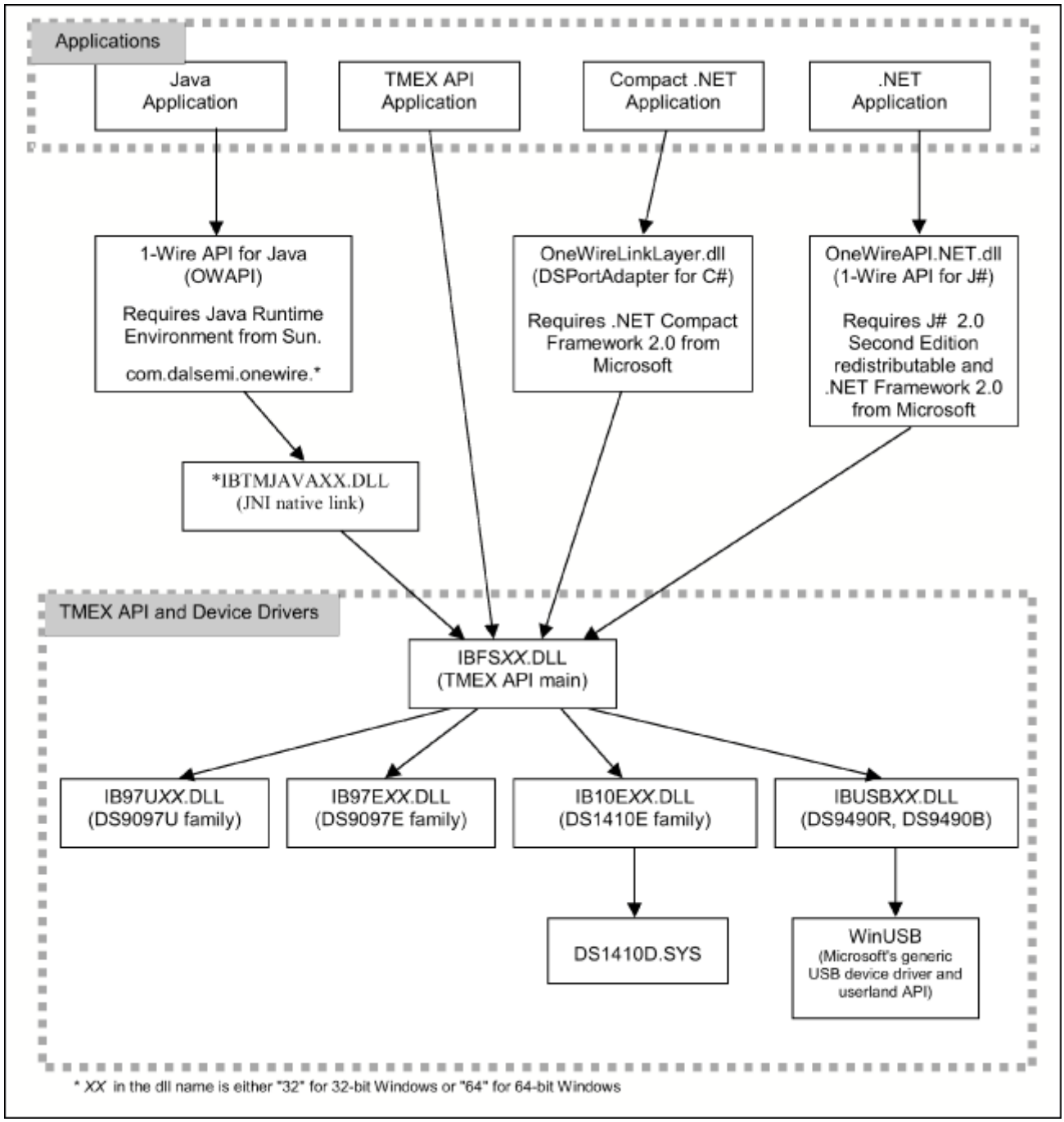

图*9. TMEX API*驱动程序和其它*API*的连通性

图10列出了TMEX API提供的功能。注：这种API不提供任何无存储器件的专用功能。

会话

TMEndSession - 放弃1-Wire网络的使用权。
TMExtendedStartSession - 请求独占1-Wire网络的使用权。
TMValidSession - 检测当前1-Wire网络会话是否有效。

链路

TMClose - 为开放端口释放资源(不总是可用)。
TMOneWireCom - 设置1-Wire网络的速度为标准模式(16kb)或高速模式传输(142kb)。
TMOneWireLevel - 设置传输线上某点电平为标准模式(5V弱上拉)，供电传输(5V强上拉)或者编程电平(12V EPROM编程电平)。
TMProgramPulse - 向1-Wire网络发送同步编程脉冲进行EPROM的编程。
TMSetup - 检测和验证端口和适配器是否正在运行。
TMTouchBit - 发送并接收1-Wire网络的1位数据。
TMTouchByte - 发送并接收1-Wire网络的1个字节。
TMTouchReset - 复位1-Wire网络上的所有数据并返回结果。

网络

TMAccess - 选择当前器件并准备执行一条专用器件指令。
TMAutoOverDrive - 设置驱动程序自动地决定器件是否进行快速模式传输。
TMFamilySearchSetup - 设置当前的搜索状态为在下一次搜索(TMNext或TMNextAlarm)中找出指定的家族类型。
TMFirst - 找出1-Wire网络上第一个1-Wire器件。
TMFirstAlarm - 找出1-Wire网络上第一个报警1-Wire器件。
TMNext - 找出1-Wire网络上下一个1-Wire器件。
TMNextAlarm - 找出1-Wire网络上下一个报警1-Wire器件。
TMOverAccess - 选定当前器件并将其速度设为快速模式传输。
TMRom - 找回或设置当前器件的串口号(ROM编号)。
TMSkipFamily - 跳过最后一次搜索发现的所有家族类型。
TMStrongAccess - 选择和验证当前器件是否存在。
TMStrongAlarmAccess - 选择和验证当前器件是否存在并且处于报警状态。

传输

TMBlockIO - 在1-Wire复位之前，发送和接收1-Wire网络的一个数据块。
TMBlockStream - 发送和接收带有NO 1-Wire复位的1-Wire网络的一个数据块。
TMExtendedReadPage - 读取存储区中含有器件产生的CRC校验码的完整的一页(并不适用于所有器件类型)。
TMProgramByte - 将一个字节编入1-Wire器件的EPROM基址。
TMReadPacket - 从存储器的页中读取通用数据包(通用数据包结构的说明请参见应用笔记114："1-Wire File Structure")。
TMWritePacket - 向存储器的页中写入通用数据包。

文件

TMAttribute - 更改文件或目录属性。
TMChangeDirectory - 读取或更改当前的工作目录。
TMCloseFile - 关闭已打开或新建的文件以释放文件句柄。
TMCreateFile - 新建一个可写文件。
TMCreateProgramJob - 新建一个写缓冲器以记录EPROM未完成的编程工作。
TMDeleteFile - 删除一个文件。
TMDirectoryMR - 新建或移走一个子目录。
TMDoProgramJob - 写EPROM未完成的编程工作。
TMFirstFile - 找出当前目录下第一个文件。
TMFormat - 格式化1-Wire文件结构文件系统。
TMNextFile - 找出当前目录下的下一个文件。
TMOpenFile - 打开一个可读文件。
TMReadFile - 读已打开的文件。
TMReNameFile - 改变文件或目录的名称。
TMTerminateAddFile - 结束‘AddFile’，‘AddFile’是EPROM上的一种特殊的文件类型，不用重写就可以打开它。
TMWriteAddFile - 添加或改变EPROM器件上的‘AddFile’。
TMWriteFile - 写已创建的文件。

器件

无

其它

TMGetTypeVersion - 获得适配器驱动程序的版本信息。
Get_Version - 获得所有驱动程序的版本。
图*10. TMEX API*函数

下面的**例4**是TMEX的代码段，它遵循图3所概括的API使用流程。和上面的三个例子一样，一个简单的程序允许每次循环期间只找出一个1-Wire网络上的器件。复杂点的应用程序一次循环中可以找出一类器件或者可以选定在上次循环中找到的器件。

```
int PortNum, PortType; // port number and type set for adapter present
long session_handle; // session handle
unsigned char state_buffer[5120];
int doing_work=1, did_setup=0;
short rslt;

// work loop
while (doing_work)
{
  // aquire the 1-Wire Net (SESSION)
  session_handle = TMExtendedStartSession(PortNum,PortType,NULL);
  if (session_handle > 0)
  {
   // check to see if TMSetup has been done once
   if (!did_setup)
   {
     if (TMSetup(session_handle) == 1)
      did_setup = 1;
     else
     {
      // error setting up port, adapter may not be present
      // . . .
     }
   }
   else
   {
     // find all devices (NETWORK)
     rslt = TMFirst(session_handle, state_buf);
     while (rslt > 0)
     {
      // do SOMETHING with device found (TRANSPORT/FILE/DEVICE)
      // . . .

      // find the next device (NETWORK)
      rslt = TMNext(session_handle, state_buf);
     }
   }

   // release the 1-Wire Net (SESSION)
   TMEndSession(session_handle);
  }
  else
  {
   // Could not acquire 1-Wire network
   // . . .
  }

  // do other application work
  // . . .
}
```

例*4. TMEX 'C'*代码实例

安装

TMEX API通过之前提到的1-Wire驱动程序安装包(也安装OW.NET API库)进行安装。它是一个Microsoft安装包，加载了所有支持1-Wire适配器的Windows驱动程序和注册表。用于客户安装的驱动程序和API文件(未安装)均可在网站上免费下载。OneWireViewer是针对大多数1-Wire和iButton器件的演示程序，可在安装完1-Wire驱动程序后进行安装。

有关1-Wire驱动程序和OneWireViewer演示软件的更多信息可在iButton网站的“软件资源”处找到。

其它工具

本文中所讨论的这些API代表了与1-Wire器件进行通信的相当一大部分可利用资源，但决不是唯一的资源。这一部分将介绍一些其它的可利用资源。

软件授权API

软件授权是应用软件对所需硬件令牌的一种锁定。这里的令牌是一种连接到用户工作站的iButton。在应用软件运行之前，轮询iButton存在与否及其有效性。还可以是在程序执行时不断地查询。实际上，本文提到的任何API都能用于这种应用程序，但也有一些必须注意的问题。由于允许用户替换驱动程序，这些外部的驱动便产生了安全漏洞，从而使锁定失败。构建在1-Wire公用工具包之上的软件授权 API就是专门为这种应用程序开发的，且它用可链接的模块代替了外部驱动。

软件授权API的设计是以使用最简单为中心目的，以便更容易地移植到软件开发者现有的代码中。API主要提供三种服务：

- 初始化一个1-Wire器件
- 验证器件
- 清除器件(因此系统中不再有有效器件)

有关1-Wire软件授权工具的信息可在iButton网站上找到。

软件实例搜索引擎

Maxim不断开发1-Wire实例应用程序，用于演示本文中提到的API。未包含在SDKs中的实例可通过在线软件实例搜索引擎找到。其中很多实例(带源代码)在加载到对应的SDK之前首先会被添加到搜索引擎(作为独立的一部分来供下载)。搜索引擎可按下列软件实例来搜索：

- 1-Wire器件(Thermochron、SHA iButton等)
- 平台(Win32、Linux、MxTNI等)
- API (TMEX、1-Wire公共资源、1-Wire .NET等)
- 编程语言(C、Java、C#、Visual Basic、Delphi等)

搜索结果包括：程序的详细说明和其对应的下载链接。

第三方

有大量的适用于1-Wire器件的第三方软件。其中有一些应用程序是购买Maxim授权方案开发者(ASD)的解决方案。iButton网站给出了完整的ASD和方案列表，讨论组也有开源程序，如SourceForge.net。

结束语

本文对各种1-Wire API的特性做了概括性论述。详细介绍了几种不同的API，包括支持它们的平台及编程语言。速查表概括了每种器件类型的API，使我们很容易选出所要的API。有了正确的API，使用1-Wire可以很容易地设计应用程序。