

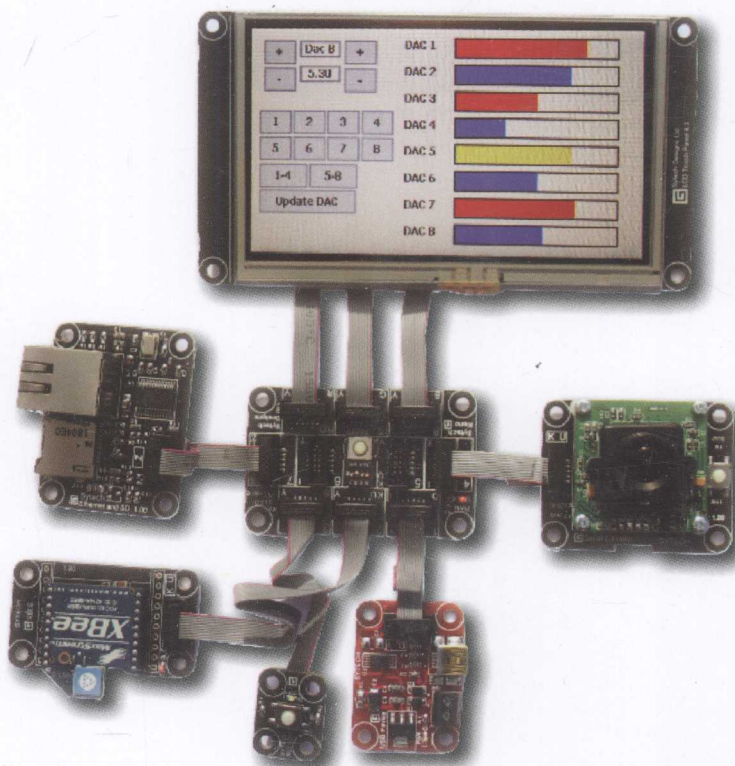
玩
转

.NET Gadgeteer

电子积木

(英) Simon Taylor 著

叶帆 林子轩 牛彦青 译



科学出版社

014035125

TP393.02

55

物联网学堂

玩转.NET Gadgeteer 电子积木

[英] Simon Taytor 著

叶帆 林子轩 牛彦青 译



科学出版社

北京

TP 393.02
55



北航

C1714618

图字: 01-2013-2621号

内 容 简 介

.NET Gadgeteer是微软剑桥研究院发明的一款使用.NET Micro Framework 和Visual Studio/Visual Studio Express开发小型电子设备的开源工具包, 具有面向对象编程、与外部电子设备免焊组装、通过计算机辅助设计快速成型等优势, 是实现物联网应用的有效途径之一。

本书内容涵盖.NET Gadgeteer软件和硬件知识, 从设置开发环境、设计应用程序到调试技术, 并以大量示例项目讲解各种编程技术和.NET Gadgeteer的各个方面。

本书可以作为高等院校自动化、电子信息、物联网相关专业的教学用书, 也可作为嵌入式、物联网初学者的入门书。

图书在版编目(CIP)数据

玩转.NET Gadgeteer电子积木/(英) Simon Taylor著; 叶帆, 林子轩, 牛彦青译. —北京: 科学出版社, 2014.5

(物联网学堂)

书名原文: Microsoft .NET Gadgeteer: Electronics Projects for Hobbyists and Inventors

ISBN 978-7-03-039909-0

I. 玩… II. ①S…②叶…③林…④牛… III. 计算机网络—程序设计 IV. TP393

中国版本图书馆CIP数据核字(2014)第038593号

责任编辑: 叶 秋 杨 凯 / 责任制作: 魏 谨

责任印制: 赵德静 / 封面制作: 付永杰

北京东方科龙图文有限公司 制作

<http://www.okbook.com.cn>

科学出版社 出版

北京东黄城根北街16号

邮政编码: 100717

<http://www.sciencep.com>

北京源海印刷有限责任公司 印刷

科学出版社发行 各地新华书店经销

*

2014年5月第 一 版 开本: 787×960 1/16

2014年5月第一次印刷 印张: 16 1/4

印数: 1—4 000 字数: 280 000

定价: 48.00元

(如有印装质量问题, 我社负责调换)

Simon Taylor

Microsoft .NET Gadgeteer: Electronics Projects for Hobbyists and Inventors

0-07-179795-5

Copyright © 2013 by McGraw-Hill Education.

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including without limitation photocopying, recording, taping, or any database, information or retrieval system, without the prior written permission of the publisher.

This authorized Chinese translation edition is jointly published by McGraw-Hill Education (Asia) and China Science Publishing and Media Ltd. This edition is authorized for sale in the People's Republic of China only, excluding Hong Kong SAR, Macao SAR and Taiwan.

Copyright © 2014 by McGraw-Hill Education (Asia), a division of McGraw-Hill Education (Singapore) Pte. Ltd. and China Science Publishing and Media Ltd.

版权所有。未经出版人事先书面许可，对本出版物的任何部分不得以任何方式或途径复制或传播，包括但不限于复印、录制、录音，或通过任何数据库、信息或可检索的系统。

本授权中文简体字翻译版由麦格劳－希尔（亚洲）教育出版公司和中国科技出版传媒股份有限公司合作出版。此版本经授权仅限在中华人民共和国境内（不包括香港特别行政区、澳门特别行政区和台湾）销售。

版权 ©2014 由麦格劳－希尔（亚洲）教育出版公司与中国科技出版传媒股份有限公司所有。

本书封面贴有 McGraw-Hill Education 防伪标签，无标签者不得销售。
北京市版权局著作权合同登记号：01-2013-2621

致 谢

感谢微软.NET Micro Framework团队，尤其是Colin Miller、Lorenzo Tessoro、Zach Libby最近几年来对我的帮助。

感谢微软剑桥研究院.NET Gadgeteer团队成员Nicolas Villar、James Scott和Steven Johnson，他们对.NET Gadgeteer的高度热情同样也激励了我。

感谢McGraw-Hill Professional的专业编辑人员为本书出版所做的工作。

最后，特别感谢我的妻子Tsvetana，本书的出版离不开她的支持和鼓励。

推荐序

以前的嵌入式工作人员很少会去借助移动互联网的力量，而常规的开发人员或许又会觉着那些电子器件过于枯燥乏味。但是，随着移动互联网技术以及硬件技术的不断发展，这一切都在慢慢地变化，“物联网”“可穿戴式设备”“智能家居”等概念这几年突然间变得炙手可热。以前搞硬件的会发现，互联网及移动设备会为嵌入式设备带来无穷的魅力，而常规的开发人员也会惊呼：原来用代码来操作硬件并不是件很难的事儿。

.Net Micro Framework是微软所推出的面向广大.Net平台开发人员的一个框架，让你可以用熟悉的工具（Visual Studio）和开发语言（C#）来方便快捷地操控嵌入式硬件设备。2007年我第一次接触到.Net Micro Framework并写出我的第一个嵌入式“hello world”程序，用C#控制一块Digi开发板上的LED闪烁着莫尔斯电码“HelloMorse Code. Hello .NET Micro Framework.”时，你无法体会到我当时激动的心情。原来不需要晦涩的汇编，也不需要C语言，我也可以玩嵌入式开发！

随后，微软于2010年将.Net Micro Framework以Apache 2.0协议开源，将其通过CodePlex开源社区(<http://netmf.codeplex.com>)分发，从而使得更多的开发人员有所了解。

.Net Gadgeteer则是微软英国剑桥研究院基于.Net Micro Framework开发的一套开源的快速硬件原型平台，目前已经有许多第三方厂商生产/设计了自己的.Net Gadgeteer模块，包括国内著名的Seeed Studio。你可以像玩乐高积木一样把各种电子元器件连接起来，然后用上短短的两三行C#代码就可以控制这些小巧的部件，读取传感器数值、通过摄像头拍照或访问网络资源。

叶帆先生是我认识多年的朋友，他在微软.Net Micro Framework的相关部门工作了多年，对此平台有着丰富的经验，而且由他自行研发的基于.Net Micro Framework的紫藤开发板和物联网智能网关在业界也都赫赫有名。相信由叶帆先生来执笔翻译此书，将会为国内的广大.Net开发人员推开嵌入式开发这扇大门。

动起手来，尝试属于.Net开发人员的硬件乐趣吧！

张 欣

微软最有价值专家（MVP）

译者序

说起.NET Gadgeteer,不得不先说一下.NET Micro Framework。虽然.NET Micro Framework已经有十几年的发展历史了,但是在全球范围内,.NET Micro Framework的知名度远远低于它的近亲.NET Framework和.NET Compact Framework,其原因值得探究。

.NET Micro Framework仅从名字上理解,就是一个框架,和.NET Framework及.NET Compact Framework应该没有什么大的不同。但是,.NET Micro Framework有自己的特色,即自启动功能,也就是不需要操作系统也能运行。这个特色将.NET Micro Framework逐渐演化为一个操作系统的角色。恰是这一点,在.NET Micro Framework早期发展过程中,越来越显得没有优势和特色。

早期的.NET Micro Framework并没有直接对第三方开放,而是作为微软.NET全战略的一环,以.NET Micro Framework为基础推出一系列产品。目前,可穿戴设备炙手可热,如智能手表。其实以.NET Micro Framework为核心的第一代产品就是智能手表,早在2003年拉斯维加斯Comdex贸易展上,比尔·盖茨就曾亲自戴着智能手表进行过推广。这项以.NET Micro Framework为核心发展起来的技术叫MSN Direct,除了智能手表产品外,还有可以预报天气的咖啡壶、GPS导航器等产品。

以.NET Micro Framework为核心的第二代产品是SideShow,曾以笔记本第二屏、智能遥控器和智能键盘的面目出现,华硕、三星的一些笔记本就包含这样的SideShow显示屏。

但是,无论是MSN Direct,还是SideShow,其产品并不成功,这和微软早期强制推广.NET战略相关。因为基于.NET技术开发此类产品,虽然开发比较快,但是对硬件资源要求比较高,其.NET托管代码相比原生C++代码速度要慢许多。所以此类产品一旦批量生产,性价比肯定比较低。最初微软也打算全部用.NET C#语言开发操作系统(如Longhorn系统),后来事实证明这种做法是行不通的。

2009年初,微软开始调整.NET Micro Framework发展战略,首先以Apache 2.0许可的授权方式完全开源.NET Micro Framework,并基于.NET Micro Framework推出第三代产品Netduino。微软这个思路其实是仿照Arduino产品而做的,从产品命名到实际硬件接口,都是学习Arduino。我个人认为这个思路是对的,至少充分发挥了.NET Micro

Framework的优势——小巧、开发迅速，并且采用强大的Visual Studio进行程序开发和在线调试，让所有的.NET程序员很容易进入嵌入式领域进行相关开发。不得不说，在软硬件开发结合越来越紧密的时代，这对.NET程序员是一个福音。

而其后推出的基于.NET Micro Framework技术的.NET Gadgeteer产品更是充分发挥了.NET Micro Framework优势，在Netduino产品的基础上更上一层楼，青出于蓝而胜于蓝，完全演化成具有微软自己特色的产品。特别是微软定义的20种.NET Gadgeteer Socket类型，应该是微软为工控领域制订OPC技术标准以来，最重要的一个接口标准。

更具特色的是拖拽式可视化编程。在DOS时代开发一个界面程序，可以说是摸着石头过河，边写代码，边运行测试，以确认相关界面的位置是否合适。Windows平台下最初的Visual Basic等可视化编程工具，让我们的界面开发进入一个新时代，每个功能模块都被封装为一个控件，通过可视化拖拽式设计界面，让我们的界面开发提升到一个新的水平。同理，我们以软件界面开发的眼光审视目前的硬件开发，其实开发模式仍处在DOS时代，每外接一个功能模块，都需要我们的程序员匠心独运，小心翼翼分配相关的针脚，并用心开发相关驱动，最终通过应用程序调用底层接口，进行数据交互。

而.NET Gadgeteer的开发环境，让我们摒弃了这些琐碎和繁杂的步骤，视每个外接模块为一个控件，通过拖拽方式，让核心主板和这些模块相连，自动完成接口初始化、模块初始化等工作。留给我们做的就是一些业务逻辑的实施和完善，这是硬件开发领域的面向对象编程，是一个具有里程碑意义的硬件开发变革。

有了这些特色，.NET Gadgeteer已经和Netduino、Arduino等相关DIY产品不同，跳出了仅仅是学习硬件、电子产品小制作的范畴。再结合目前发展得如火如荼的3D打印技术，.NET Gadgeteer已经成为快速制作最终产品的最好选择（特别是小批量、个性化产品）。在物联网发展迅猛的今天，.NET Gadgeteer更是可以大显身手，可以方便地接入各种传感器模块，并把相关数据上传到云端。

本书共13章，由我和两位技术网友共同翻译完成。其中，牛彦青翻译了第1、2、5、6章及附录；林子轩翻译了第7、8、9、10章；我翻译了第3、4、11、12、13章，并对全书统稿。

初次翻译此类书籍，并且时间仓促有限，错误恐在所难免，敬请读者朋友和专家指正。

叶 帆

2013年9月12日于北京

序

Frederick Brooks，是软件工程学名著《人月神话》(*The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley Professional)的作者。他在书中描述了编程的喜悦：“一种创建事物的纯粹快乐”和“一种可触及的令人愉悦的工作”，“它打印结果，绘制图片，发出声音，移动机械臂”。几乎和我聊过天的每个程序员都记得他们早期的实验，如使用发光二极管和伺服系统，即使他们有一段时间不从事创建设备的工作。

所幸我有过编程教学经验，曾体会到在互动过程中创造新东西，与其他事物、与周围环境协调是一件很令人着迷的事。也有机会经历设备连接时发生爆炸的情况，从而促进了新一代的设备和技术的发展。.NET Gadgeteer同样也经历了这些过程。

做这类项目面临的挑战是，最初阶段的学习曲线总是陡峭的。在过去，想要完成一个电子项目，哪怕是简单地开启LED的操作，你都需要知道从哪里得到及如何选择兼容电子零件，如何加载兼容的开发工具，学习一种新的语言或新的桌面语言，掌握一种制作技能。如模板的调试和焊接，至少应学习GPIO级别的嵌入式接口知识，了解如何将你的代码部署到设备，然后祈求老天保佑。如果它没有按照你预期的结果执行，就必须弄清楚如何调试你所连接的设备（布线、逻辑及任何其他方面）。

.NET Gadgeteer改变了这一模式，通过.NET Gadgeteer，在你创建简单的原型之前，不需要学习太多的硬件知识。现在，你可以很快地创建激动人心的东西。如果你需要，可以继续深入研究。这一方式的优势是相当明显的。几乎每天，都有人在博客上发布，采用Gadgeteer技术，以很快的速度创造的一些很酷的东西。如第一个标准示例，我们在.NET MF/Gadgeteer中编写摄像头应用程序仅用了4行代码，而且仅仅是将模块连接起来就可以了。这种简单性是很重要的，因为这意味着完成一件很酷的作品会很迅速。

然而，一旦你创建了这个简单的相机和伺服系统，之后会怎样呢？当你按下按钮时，快速四线相机将会拍摄照片，并在显示器上显示拍摄的照片。之后，图片可能会存储到SD卡上，而且可以将SD卡上的图片滚动播放，可以删除与你最初设想不一致的图片。突然间，你开始想编写更多行代码，如文件的输入/输出、触摸屏界面操作及其他。

这就是我喜欢本书的原因，它不仅仅能实现一些简单的项目（一些你可以在互联网上找到的），而且在你完成前一个项目之后，如果有需要还可以获得更多的信息，来创建完全属于你自己的项目。例如，在你不知道什么是I²C的时候，也可以创建很多很酷的东西，但在某些时候，你可能会用到一个特殊的传感器，而只能I²C访问。最近这些天，我已经很少创建项目（不包括扩展模块）了，一个原因在于MCU本身具有某些功能。除非始终没有我所想要的模块，或者已有的模块不适用。本书提供一个快捷的.NET Gadgeteer入门指南，也仅提供一个继续学习的背景，我们要了解的东西还很多。

本书主题涉及Visual Studio的配置和使用，包括在工具集中与其他程序相比有何特点；当设备运行的时候是如何管理电源的；事件驱动设计模式是什么，为什么它是Gadgeteer应用程序的核心模式。这只是几个例子，实际上它涉及的领域相当多。如果你从网上复制过一些有趣的应用程序，而现在想加以扩展或创建自己的应用程序，本书将会为你提供搭建小型设备所需的几乎任何知识。

Colin Miller

微软.NET MF 产品经理

前 言

嵌入式设备是硬件和软件的组合。在硬件方面，需要有特定的传感器和控制器接口与实际应用相连。软件定义了设备的行为及传感器和控制器的反馈方式。

微软 .NET Gadgeteer 定义了硬件和固件标准，制造商可以开发符合标准的传感器模块，以使其能够与处理器主板兼容。这样就简化了嵌入式设备的第一个需求——硬件。你可以选择适合项目的传感器和接口，只需将其插入处理器主板。此标准通过处理器主板定义所需的接口，并且允许使用一系列来自不同制造商的处理器主板。.NET Gadgeteer 还定义了传感器和主板之间的固件接口，使传感器可以使用处理器主板提供的底层驱动。

第二个需求是实现应用程序各种功能的软件。微软 .NET Micro Framework 提供编程接口运行时操作系统（基于桌面 .NET Framework），还提供了完整的开发和调试工具，如 Visual Studio Express。应用程序可采用高级语言（C# 或 Visual Basic）编写。

结合兼容 Gadgeteer 的硬件和 .NET Micro Framework 系统，可以通过非常简单的方式组合成复杂的嵌入式系统。

本书目的在于探讨 .NET Gadgeteer 中的各种元素是如何工作的，你将获得创建自己的项目的相关知识。我们不仅仅介绍如何在项目中使用各种 Gadgeteer 传感器，还深入探讨 .NET Gadgeteer 的原理及编程的各个方面。理解了 .NET Gadgeteer 如何工作及如何与 .NET Micro Framework 交互，开发无错的复杂应用程序将会更简单。

本书内容涵盖了使用 .NET Gadgeteer 的各个方面，从设置开发环境、设计应用程序到调试技术。我们会通过示例项目讲解各种编程技术和 .NET Gadgeteer 的各个方面。

学完本书后，你将收获搭建小型嵌入式设备的知识和信心。

目 录

第 1 章 .NET Gadgeteer概述

1.1 .NET Gadgeteer基本组成	1
1.1.1 硬件接口	1
1.1.2 固件	2
1.2 Micro Framework和Gadgeteer简介	2
1.2.1 TinyCLR简介	3
1.2.2 基类层	4
1.3 Gadgeteer架构	4
1.3.1 硬件接口	4
1.3.2 固件接口	6
1.4 创建Gadgeteer应用程序	7
1.4.1 串行摄像头模块	7
1.4.2 应用程序设计器	8

第 2 章 软件开发环境

2.1 安装Visual C# 2010 Express	13
2.2 安装.NET Micro Framework	17
2.3 安装Gadgeteer Core SDK	21
2.4 Gadgeteer文档	24
2.5 安装Gadgeteer Mainboard和Modules SDK	25
2.6 小 结	36

第 3 章 Gadgeteer Socket、主板和模块

3.1 Gadgeteer Socket	38
----------------------------	----

3.2	主 板	41
3.3	模块和接口	43
3.3.1	Module基类	44
3.3.2	DaisyLinkModule基类	45
3.3.3	DisplayModule基类	46
3.3.4	NetworkModule基类	48
3.4	Gadgeteer 应用程序	48
3.4.1	Program基类	49
3.4.2	应用程序	50
3.5	Gadgeteer 接口、实用功能和服务	52
3.5.1	接 口	52
3.5.2	实用功能	52
3.5.3	服 务	54

第 4 章 Gadgeteer的API接口

4.1	模拟输入/输出	55
4.1.1	AnalogInput类	55
4.1.2	AnalogOutput类	56
4.2	数字输入、输出和输入/输出	57
4.2.1	DigitalInput接口	57
4.2.2	DigitalOutput接口	57
4.2.3	DigitalIO接口	58
4.3	InterruptInput类	58
4.4	PWMOutput类	59
4.5	I2CBus类	59
4.6	Serial类	60
4.7	SPI类	61

第 5 章 Gadgeteer主板和模块

5.1	Gadgeteer主板	63
5.1.1	GHI Electronics	63

5.1.2	Mountaineer Group	65
5.1.3	Love Electronics	68
5.1.4	Sytech Designs Ltd.	69
5.2	Gadgeteer 模块	71
5.2.1	以太网、WiFi和SD卡	71
5.2.2	图形显示器	71
5.2.3	I ² C和SPI模块	71
5.2.4	串行模块	72

第 6 章 部署和调试

6.1	TinyCLR和TinyBooter	74
6.2	使用MFDeploy	75
6.2.1	MFDeploy主界面	75
6.2.2	MFDeploy功能	77
6.3	用Visual Studio部署和调试	84
6.3.1	编译项目	85
6.3.2	设置断点	86
6.3.3	立即执行	89
6.3.4	单步执行代码和移动执行点	89
6.3.5	Visual Studio的更多特性	90

第 7 章 编写Gadgeteer应用程序

7.1	过程式和事件驱动式应用程序	91
7.1.1	基本设计流程：过程式与事件驱动式的对比	92
7.1.2	电池省电设计	92
7.2	Gadgeteer应用程序流程	94
7.2.1	Gadgeteer应用模板	95
7.2.2	应用程序线程	97
7.2.3	类与项目代码文件	99
7.2.4	使用过程式代码	102
7.3	小 结	106

第8章 数据输入/输出项目

8.1 在Visual Studio Express里创建空方案	107
8.2 SPI显示器模块：使用项目资源文件	111
8.3 I ² C加速度计与数据处理线程	116
8.4 Gadgeteer DaisyLink	123
8.5 集合多个模块的项目	126
8.5.1 创建项目	128
8.5.2 JoyInput类及其事件	129
8.5.3 DemoApp类	133
8.5.4 Gadgeteer Program.cs	138
8.6 小 结	138

第9章 串行通信项目

9.1 使用Serial2USB模块建立串行通信项目	139
9.1.1 创建新项目	140
9.1.2 启动并调试应用程序	146
9.1.3 启动终端应用程序	147
9.1.4 变更串行端口的物理设置	150
9.2 串口信息数据处理	153

第10章 SD卡与文件处理

10.1 挂载和卸载可移动媒体	157
10.2 GHI主板	158
10.3 Sytech NANO主板	160
10.4 目录与文件处理	161
10.4.1 使用StorageDevice类	162
10.4.2 目 录	162
10.4.3 文 件	163
10.5 保存与恢复设置数据项目	168
10.5.1 添加类到项目	168
10.5.2 Program.cs文件	174
10.5.3 Micro Framework 扩展弱引用	177

10.6 文本与CSV文件项目	177
10.6.1 简易文本记录器项目	177
10.6.2 CSV文件项目	182
10.7 小 结	189

第11章 以太网和Web设备项目

11.1 网络socket	192
11.1.1 设备网络设置	193
11.1.2 TCP/IP服务项目	196
11.2 Web设备	213
11.2.1 Web服务器	213
11.2.2 Web客户端	218
11.3 Micro Framework网络支持	220
11.4 小 结	220

第12章 设计Gadgeteer模块和主板

12.1 模 块	223
12.1.1 简易的定制原型模块	223
12.1.2 使用模块项目模板	224
12.1.3 GadgeteerHardware.XML	227
12.1.4 MSI的生成	229
12.2 主 板	232

第13章 将Gadgeteer原型转化成产品

13.1 使用现有模块还是自行设计	233
13.2 包装你的原型	235

附 录 Gadgeteer与Micro Framework 4.2

第 1 章

.NET Gadgeteer概述

.NET Gadgeteer是由微软剑桥研究院开发的。它最初仅用于研究院内部，旨在简化嵌入式设备的开发，建立一套硬件传感器和.NET Micro Framework系统连接的标准接口。此项目应用的广泛性很快显现，因而研究院决定将此技术公开，使之成为开源项目。



本书中将微软 .NET Gadgeteer（此注册商标归微软所有）简称为“Gadgeteer”。

1.1 .NET Gadgeteer基本组成

Gadgeteer技术将物理硬件接口和固件框架结合起来作为一个标准机制（或称为工具包），其目的在于将硬件传感器和其他外围设备（或称为模块）与处理器板（或称为主板）连接起来。我们可以通过Gadgeteer技术将不同制造商生产的模块与主板相连，以满足用户的各种应用需要。为避免与Micro Framework 运行时引起可能的混淆，在此我们以Gadgeteer Framework表示Gadgeteer内核。

1.1.1 硬件接口

通常情况下，Gadgeteer主板是黑色的，而供电模块设计成红色，以便于区分。Gadgeteer的物理硬件接口控制主板和模块之间的连接，定义连接器类型（10针，1.27mm）和针脚功能（哪些针脚用来供电，哪些针脚用做数据传输）。物理连接采用2×5针的IDC（绝缘位移连接器）和带状电缆。硬件电路板也进行了定义，其中包括安装孔的位置及间距、电路板的尺寸等。