



Leseprobe

Stephan Hüwe

Raspberry Pi für Windows 10 IoT Core

Der praktische Einstieg für Anwender und Entwickler

ISBN (Buch): 978-3-446-44719-6

ISBN (E-Book): 978-3-446-44809-4

Weitere Informationen oder Bestellungen unter

<http://www.hanser-fachbuch.de/978-3-446-44719-6>

sowie im Buchhandel.

Inhaltsverzeichnis

1	Einführung	1
1.1	Für wen ist dieses Buch interessant?	1
1.2	Wie ist dieses Buch aufgebaut?	2
1.3	Wichtige Hinweise	3
1.4	Material zum Buch	4
2	Schnelleinstieg in die Welt des Raspberry Pi	5
2.1	Ursprung	5
2.2	Varianten	6
2.3	Der Weg zu Windows 10	8
2.4	Aufbau	8
2.5	Schnittstellen	9
2.5.1	GPIO	10
2.5.2	SPI	11
2.5.3	I ² C	11
2.5.4	UART	12
2.6	Prototyping und Testaufbauten	13
2.6.1	Breadboarding	13
2.6.2	Softwareunterstützung	14
2.6.2.1	Fritzing	14
2.6.2.2	Virtual Breadboard	16
2.7	Elektrotechnische Grundlagen	17
2.7.1	Vorsichtsmaßnahmen im Umgang mit Spannung	17
2.7.2	Statische Aufladung vermeiden	17
2.7.3	Ohmsches Gesetz	18
2.7.4	Energieversorgung am Raspberry Pi	18
2.7.4.1	Stromstärke und Wahl des Netzteils	19
2.7.4.2	Unterbrechungsfreie Stromversorgung	20

2.7.4.3	Mobile Stromversorgung	20
2.7.4.4	Energieversorgung und GPIO	21
2.7.5	Stromversorgung von Bauteilen (z.B. Motoren)	22
2.7.6	LEDs	23
2.7.7	Widerstände	25
2.7.8	Weitere Bauteile	26
2.8	Benötigte Ausrüstung	27
2.8.1	Raspberry Pi-Ausrüstung	27
2.8.2	Offizielles Zubehör	28
2.8.2.1	Offizieller WLAN-Adapter	28
2.8.2.2	Raspberry Pi-Touchscreen-Display	28
2.8.3	Allgemeine Elektronik	30
2.8.4	Was brauche ich sonst noch?	30
3	Internet of Things mit Windows 10 IoT Core	31
3.1	Chancen für Windows 10 IoT Core	31
3.2	Internet of Things (IoT)	34
3.2.1	Ursprung und Idee	34
3.2.2	IoT als wachsender Markt	35
3.3	Beschaffung und Wartung	36
3.4	Sicherheit	37
3.5	Rechtliche Themen	38
4	Windows 10 auf dem Raspberry Pi	41
4.1	Eine letzte Windows-Version für alle	41
4.2	Bezugsvarianten und Updates	42
4.3	Lizenzierung	43
4.4	Installation	44
4.4.1	Voraussetzung	44
4.4.2	Unterstützte Schnittstellen und Geräte	45
4.4.3	Download	46
4.4.4	Installation der Windows 10 IoT Core-Tools	46
4.4.5	Hinweise zur Netzwerkkumgebung	48
4.4.5.1	Verbindung in das lokale Netzwerk	49
4.4.5.2	Direktverbindung zu Ihrem PC	49
4.4.6	Raspberry Pi für den ersten Start vorbereiten	49
4.4.7	Bootvorgang und erster Start	50
4.4.8	Administration des Raspberry Pi	52

4.5	Inbetriebnahme und Administration	52
4.5.1	Default-App	53
4.5.2	Raspberry Pi im Netzwerk finden	53
4.5.3	Verbindung über FTP	53
4.5.4	Verbindung über SSH	54
4.5.5	Verbindung über PowerShell	56
4.5.6	Allgemeine Kommandos	58
4.5.6.1	Lokale Benutzer anlegen	58
4.5.6.2	Benutzer zu Gruppen zuweisen	58
4.5.6.3	Passwort setzen	59
4.5.6.4	Gerätenamen abrufen und setzen	59
4.5.6.5	Netzwerkkonfiguration	59
4.5.6.6	Kopierwerkzeuge	59
4.5.6.7	Prozessverwaltung	59
4.5.6.8	Administration der Startup-App	60
4.5.6.9	Boot-Option festlegen	61
4.5.6.10	Geplante Aufgaben	61
4.5.6.11	Gerätetreiber	61
4.5.6.12	Zugriff auf die Registry	61
4.5.6.13	Dienste	61
4.5.6.14	Bootkonfiguration	61
4.5.6.15	Gerät herunterfahren oder neu starten	62
4.5.6.16	Bildschirmauflösung ändern	62
4.5.7	Weboberfläche	62
4.5.7.1	Grundlegendes	62
4.5.7.2	Administrationsbereiche	63
4.5.8	Bereich Home	63
4.5.9	Bereich Apps	64
4.5.10	Bereich Processes	65
4.5.11	Bereich Performance	65
4.5.12	Bereich Debugging	66
4.5.13	Bereich Realtime Event Tracking	67
4.5.14	Bereich Performance Tracing	68
4.5.15	Bereich Devices	69
4.5.16	Bereich Bluetooth	69
4.5.17	Bereich Networking	69
4.5.18	Bereich Windows Update	70

5	Entwicklung mit dem Raspberry Pi	71
5.1	Vorbereitung	71
5.2	Installation und Einrichtung	72
5.3	Windows IoT Core-Projektvorlagen	73
5.4	Das Konzept der Universal Apps	75
5.5	Beispielanwendung: Hello Pi	77
5.5.1	Benötigte Bauteile	78
5.5.2	Hardwareaufbau	78
5.5.3	Projekt in Visual Studio anlegen	79
5.5.4	Programmaufbau	81
5.5.4.1	Hinterlegen des Begrüßungstexts und Start-Buttons	82
5.5.4.2	Timer-Komponente	83
5.5.4.3	GPIO	83
5.5.5	Das gesamte Programm	84
5.5.6	Deployment und Debugging	86
5.5.6.1	Deployment und Test über Visual Studio	86
5.5.6.2	App-Pakete für den Store oder die lokale Verwendung	88
6	Projekte mit dem Raspberry Pi	95
6.1	Grundinformationen zum Source Code	95
6.2	Remote-Lichtschalter	96
6.2.1	Benötigte Komponenten	97
6.2.2	Hardwareaufbau	97
6.2.3	Softwareaufbau	97
6.2.4	Projektstruktur und Source Code	98
6.2.4.1	Besondere Projekteigenschaften	98
6.2.4.2	Modifizierung des Projekts	99
6.2.4.3	Hauptanwendung – HttpServer	100
6.2.5	Code auf GitHub	103
6.2.6	Demo	104
6.2.7	Ausblick	105
6.2.7.1	Verschönerung der Weboberfläche/Webservice	105
6.2.7.2	Schaltung einer „echten“ Lampe statt LED	105
6.3	Begrüßungsscreen für Besucher	108
6.3.1	Benötigte Komponenten	109
6.3.2	Hardwareaufbau	109
6.3.3	Softwareaufbau	109
6.3.4	Projektstruktur und Source Code	110
6.3.4.1	Format der Quelldaten	110

6.3.4.2	ViewModel	111
6.3.4.3	Hauptanwendung – Code Behind	112
6.3.4.4	Hauptanwendung – XAML	114
6.3.5	Code auf GitHub	116
6.3.6	Demo	116
6.3.7	Ausblick	116
6.3.7.1	Darstellung von Geschäftszahlen	116
6.3.7.2	Unser eigener TV-Sender	117
6.3.7.3	Wochenplaner für die Familie	117
6.4	Temperatursensor mit SPI	117
6.4.1	Benötigte Komponenten	117
6.4.2	Hardwareaufbau	118
6.4.2.1	Temperatursensor TMP36GT9Z	118
6.4.2.2	A/D-Wandler MCP 3002	119
6.4.2.3	Aufbau der Schaltung	120
6.4.3	Softwareaufbau	122
6.4.4	Projektstruktur und Source Code	123
6.4.4.1	Hauptanwendung – Code Behind	123
6.4.4.2	Hauptanwendung – XAML	125
6.4.5	Code auf GitHub	126
6.4.6	Demo	126
6.4.7	Ausblick	126
6.5	Kamera-Projekt (Mobile und IoT)	128
6.5.1	Benötigte Komponenten	128
6.5.2	Hardwareaufbau	128
6.5.3	Softwareaufbau	129
6.5.4	Projektstruktur und Source Code	129
6.5.4.1	Hauptanwendung – Code Behind	129
6.5.4.2	Hauptanwendung – XAML	132
6.5.5	Code auf GitHub	133
6.5.6	Demo	134
6.5.7	Ausblick	135
6.6	Mobile Datenerfassung mit dem Raspberry Pi	135
6.6.1	Benötigte Komponenten	136
6.6.2	Hardwareaufbau	136
6.6.2.1	Stromversorgung	136
6.6.2.2	Benutzerinteraktion und Bildschirmausgaben	140
6.7	Raspberry Pi und Cloud	141
6.7.1	Benötigte Komponenten	143
6.7.2	Hardwareaufbau	144

6.7.3	Softwareaufbau	144
6.7.4	Einrichtung der Cloud	144
6.7.4.1	Registrierung des Raspberry Pi am IoT Hub	147
6.7.5	Projektstruktur und Source Code	150
6.7.5.1	Projekt zum Nachrichtenversand – MessageSender	151
6.7.5.2	Projekt zur Nachrichtenverarbeitung – MessageProcessor	153
6.7.6	Code auf GitHub	155
6.7.7	Demo	155
6.7.8	Ausblick	155
6.8	Weitere Projektideen	157
6.8.1	Kommunikation über Bluetooth	157
6.8.2	Sprachausgabe	159
6.8.3	Bildschirmausgabe	161
6.8.4	Motoren	164
6.8.4.1	Gleichstrommotoren	165
6.8.4.2	Schrittmotoren	165
6.8.4.3	Servo-Motoren	166
6.9	Exkurs: Arduino Wiring/Sketch	167
6.9.1	Der Arduino	167
6.9.2	Crashkurs Arduino Sketch	169
6.9.3	Arduino Wiring-Apps mit Visual Studio	171
6.9.4	Umstellung des Controller-Treibers	174
	Stichwortverzeichnis	177

1

Einführung

Der Raspberry Pi (auch RPi, RasPi oder einfach nur Pi genannt) hat die Computerwelt zweifellos umgekrempelt. Es handelt sich dabei um einen kreditkartengroßen, aber vollwertigen Computer. Er ist kostengünstig in der Anschaffung und benötigt nur eine 5V-Versorgungsspannung.

Der RasPi ist die perfekte Plattform zur Realisierung eigener (hardwarenaher) Projekte. Lange Zeit war nur Linux für den Raspberry Pi verfügbar. Das machte den Raspberry vor allem für Windows- bzw. .NET-Entwickler nicht sonderlich interessant. Man musste seine Projekte entweder mit einer anderen Programmiersprache (wie z.B. Python) umsetzen oder auf Mono zurückgreifen. Darüber hinaus musste man auf Visual Studio verzichten und beispielsweise auf MonoDevelop als IDE ausweichen. In der Summe ist dies jedoch alles sehr mühsam.

Die Kehrtwende erfolgte im März 2015, als Microsoft das sogenannte Windows 10 IoT Core (IoT = Internet of Things) für den Raspberry Pi ankündigte¹. Somit war klar, dass auch das .NET-Framework unterstützt werden würde und man wie gewohnt mit Visual Studio arbeiten kann.

Dieses Buch greift den Paradigmenwechsel auf und führt in die neue Welt rund um Windows 10, den Raspberry Pi und das Internet der Dinge ein.

■ 1.1 Für wen ist dieses Buch interessant?

Dieses Buch richtet sich an alle, die schon lange in die Welt des Raspberry Pi eintauchen wollten, aber durch Linux davon abgeschreckt wurden. Es bietet einen Einstieg mit gewohnten Werkzeugen aus der Windows-Welt. Es wendet sich damit an all jene, die ihre

¹ <https://blogs.windows.com/windowsexperience/2015/03/18/windows-10-iot-powering-the-internet-of-things>

Projekte lieber mit Windows und Visual Studio umsetzen möchten als beispielsweise mit Python unter Linux. Dies spart zusätzliche Einarbeitungszeit und nimmt viele Hürden.

Windows 10 IoT Core ist bestimmungsgemäß kein Desktop-Betriebssystem. Es lassen sich daher keine Anwendungen, wie z. B. ein Browser, installieren. Jedoch dient es als sichere Betriebssystemplattform für die Realisierung eigener Projekte.

Hinsichtlich der Entwicklung sollte man bereits über Grundkenntnisse in einer .NET-Programmiersprache verfügen. Was den Raspberry Pi und die damit verbundene Elektronik betrifft, kann man durchaus ein Neuling sein. Dennoch ist ein gewisses Verständnis von Physik und Computerhardware sicher von Vorteil.

Neben den Hobby-Programmierern und -Entwicklern richtet sich dieses Buch auch an all jene, die den Raspberry Pi nicht nur im privaten, sondern auch im professionellen Bereich einsetzen möchten. Auch wenn Sie sich selbst nicht mit der Entwicklung beschäftigen möchten, so bekommen Sie zumindest eine Idee davon, wie der Raspberry das Unternehmensumfeld prägen könnte. Sie erfahren, welche Probleme Sie mit ihm lösen oder welche Geschäftsfelder sich mit ihm eröffnen könnten. Auf die möglichen Risiken wird ebenfalls hingewiesen.

Die enthaltenen Projektideen liefern Ihnen erste Impulse für eigene Projekte. Mithilfe der erlernten Grundlagen werden Sie auch in der Lage sein, Linux-Projekte auf Windows IoT Core zu portieren.

■ 1.2 Wie ist dieses Buch aufgebaut?

Dieses Buch ist in sechs Kapitel gegliedert. Nach diesem Einführungskapitel wird in Kapitel 2 der Einstieg in die Welt des Raspberry Pi vorbereitet. Dabei wird die Evolution des Raspberry aufgezeigt, es werden Grundlagen der Elektrotechnik vermittelt und Sie erhalten eine Kaufempfehlung für eine Grundausstattung. Denn viele von Ihnen haben sicherlich noch gar keine Erfahrungen mit dem Einplatinencomputer gemacht und benötigen deshalb erst einmal ein paar grundlegende Informationen über den RasPi.

Kapitel 3 beschreibt perspektivisch, welchen Einfluss das Internet der Dinge und damit auch der Raspberry Pi in Zukunft haben werden, und wie Microsoft diese Entwicklung mit Windows 10 IoT Core und der Cloud-Plattform Azure IoT-Suite vorantreibt. Diese Informationen sind sowohl für den Entwickler („Wie kann ich Projekte im privaten oder professionellen Umfeld umsetzen?“), aber auch für Unternehmen („Wie müssen wir uns ausrichten, um neue Märkte zu eröffnen?“) interessant.

Kapitel 4 beschäftigt sich intensiv mit Windows 10, insbesondere mit der IoT Core-Variante. Hier erfahren Sie alles von der Bezugsquelle über die Installation und Inbetriebnahme bis hin zur Verwendung.

Kapitel 5 erklärt, wie Sie Software für den Raspberry entwickeln können – und das natürlich unter Verwendung des Dreigespanns Raspberry Pi, Windows 10 und Visual Studio.

Kapitel 6 enthält eine Vielzahl von Beispielprojekten. Diese haben zwei Aufgaben: Anhand der Beispiele erlernen Sie wichtige Grundlagen und sammeln eigene Erfahrungen. Darüber hinaus sollen diese Beispiele selbstverständlich auch das Sprungbrett für die Umsetzung eigener Ideen sein.

■ 1.3 Wichtige Hinweise

Neben der Programmierung hat dieses Buch logischerweise auch einen deutlichen Hardwarebezug. Von daher gibt es auch viele Projekte, die mit elektrischen Schaltungen und damit einer Stromquelle zu tun haben. Das mag für den reinen Softwareentwickler vielleicht Neuland sein.



Deshalb gilt der Grundsatz: Der leichtsinnige Umgang mit Strom ist immer gefährlich. Die Netzspannung mit 220 V ist generell tabu. Aber auch niedrigere Spannungen können gefährlich sein.

Von daher der dringende Rat: Sollte Ihnen etwas unklar sein, machen Sie nicht weiter, sondern suchen sich den Rat eines Fachkundigen. Überprüfen Sie vor jedem Testlauf gewissenhaft den Aufbau auf mögliche Gefahren.

Windows 10 IoT Core ist noch relativ neu am Markt. Während der Entstehung des Buches war es noch nicht fertig entwickelt. Aber auch mit Erscheinen dieses Buches wird es vermutlich noch weiter einen steten Wandel durchlaufen.

Von daher kann es gut sein, dass sich einige der hier ausgeführten Sachverhalte in der Zwischenzeit geändert haben. Dies habe ich beim Schreiben vor allem hinsichtlich der Hardwareunterstützung und möglichen Bugs gemerkt. Sofern möglich, habe ich erwähnt, mit welcher Version ich getestet habe.

Darüber hinaus habe ich auch immer die URLs angegeben, von denen der aktuelle Stand abgerufen werden kann. Da Microsoft jedoch dafür bekannt ist, die eigenen Webseiten ständig umzubauen, kann es sein, dass ein Link ins Leere führt. In diesem Fall empfehle ich Ihnen, gezielt nach der URL zu suchen. Oft findet sich in Foren von Microsoft ein Hinweis auf die aktuelle Webadresse.

Da der automatische Anpassung auch Grenzen gesetzt sind, steht Ihnen die Möglichkeit zur Verfügung, gezielt gegen definierte Display-Größen zu entwickeln. Dabei werden die verfügbaren Auflösungen mitberücksichtigt (Bild 5.6).

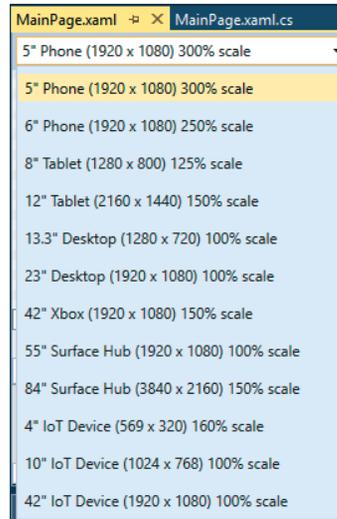


Bild 5.6 Auswahl der gewünschten Auflösung der Zielplattform

■ 5.5 Beispielanwendung: Hello Pi

Unser erstes Beispielprojekt mit dem Namen „Hello Pi“ führt in die Softwareentwicklung für den Raspberry Pi in Kombination mit Windows 10 und Visual Studio 2015 ein.

Das Projekt soll zwei Aufgaben erledigen:

- Ausgabe des Textes „Hello Pi“ über HDMI am Monitor: Hiermit wird der grundlegende Aufbau der Universal Apps gezeigt.
- Blinken einer LED in einem zeitlichen Rhythmus: So erlernen Sie exemplarisch die Ansteuerung der Hardware (in diesem Fall der GPIO-Pins).

Das „Hello Pi“-Beispiel basiert auf dem Einsteiger-Beispiel „Blinky“ von Microsoft².

² <https://ms-iot.github.io/content/en-US/win10/samples/Blinky.htm>

5.5.1 Benötigte Bauteile

Zur Umsetzung dieses Beispielprojekts benötigen Sie:

- ein Breadboard
- einige Steckbrücken
- eine LED
- einen 220-Ohm-Widerstand

Darüber hinaus sollte der Raspberry über einen Netzwerkzugriff verfügen. Damit Sie das Programm steuern können, sind zusätzlich eine USB-Maus und ein angeschlossener Monitor am HDMI-Ausgang hilfreich.

5.5.2 Hardwareaufbau

Der Hardwareaufbau ist einfach:

1. Verbinden Sie das kürzere Beinchen (Kathode, Minus) der LED mit GPIO 5 (das ist Pin 29).
2. Verbinden Sie das längere Beinchen mit dem Widerstand.
3. Verbinden Sie das andere Beinchen des Widerstands mit einem der 3,3 V-Pins auf dem Raspberry.



HINWEIS: Die Polarität der LED ist wichtig, damit das Beispiel funktioniert. Diese Art der Schaltung nennt man „Low-aktiv“. Dies bedeutet, wenn ein Low-Signal (0) anliegt, wird die LED leuchten.

Die LED sollte zunächst nicht leuchten. Einen schematischen Aufbau des Projekts finden Sie in Bild 5.7.

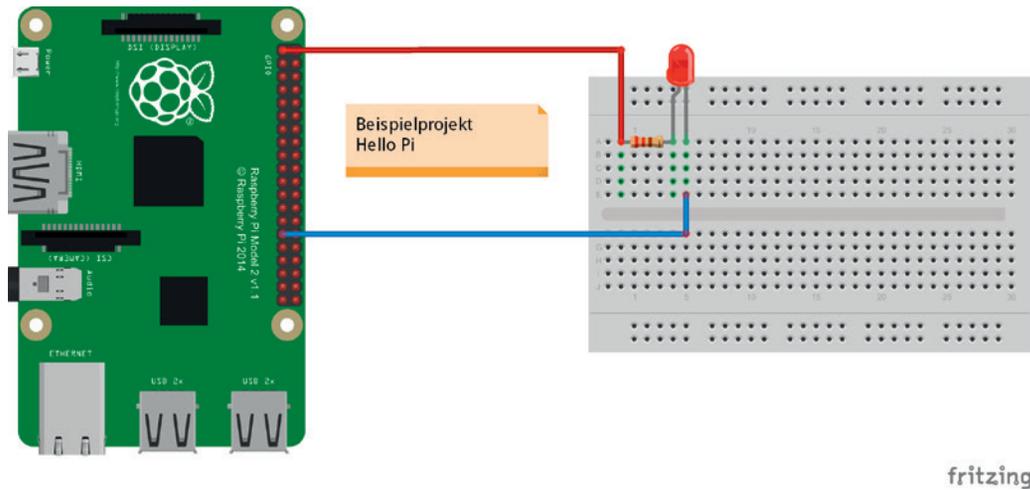


Bild 5.7 Schaltungsaufbau des Beispielprojekts „Hello Pi“

Bild 5.7 zeigt, wie die Schaltung auf einem Breadboard aussehen könnte.

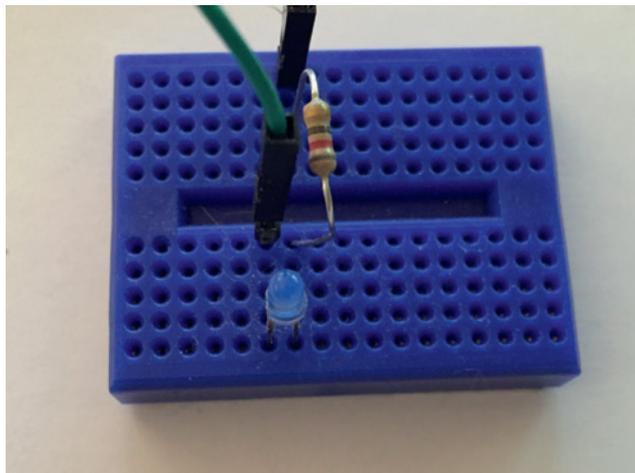


Bild 5.8 Breadboard-Schaltung der LED

5.5.3 Projekt in Visual Studio anlegen

Legen Sie zunächst ein neues Projekt in Visual Studio an. Hierzu wählen wir zunächst den Projekttyp *Blank App (Universal Windows)* aus (Bild 5.9). Achten Sie auf den Namen und das gewünschte Projektverzeichnis. Die Anbindung einer Quellcode-Verwaltung ist selbstverständlich auch bei Universal Apps möglich.

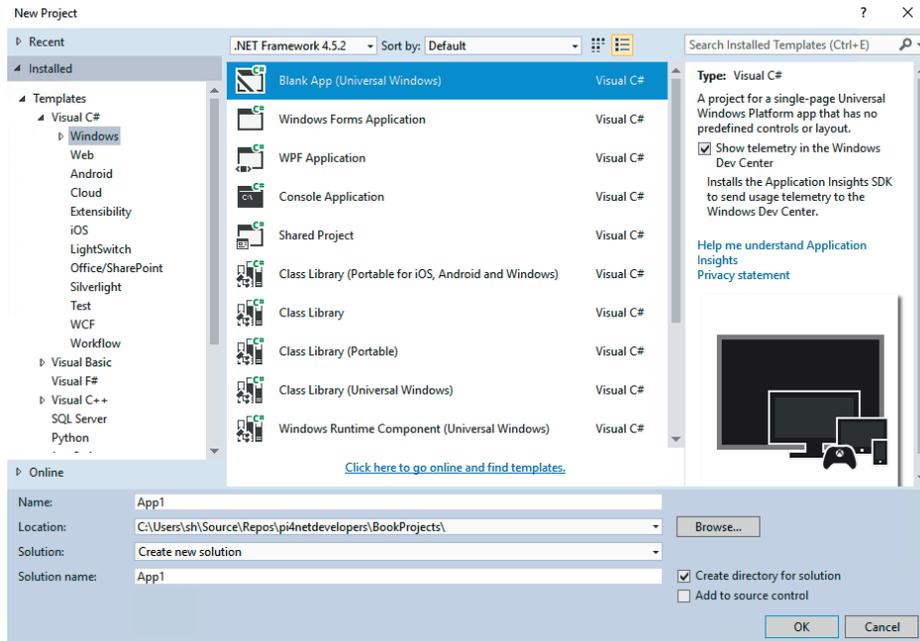


Bild 5.9 Neues Projekt in Visual Studio anlegen

Nachdem das Projekt angelegt wurde, müssen Sie zunächst eine neue Referenz hinzufügen (Bild 5.10) und dort *Windows IoT Extensions for the UWP* auswählen. Sie finden die Referenz unter:

Universal Windows->Extensions->Windows IoT Extensions for the UWP

Diese Referenz ist notwendig, damit wir die GPIO-Pins vom Code aus ansteuern können.

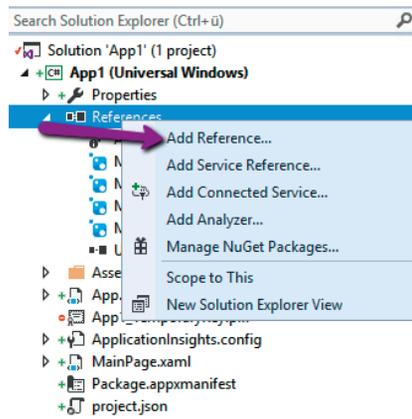


Bild 5.10 Hinzufügen einer neuen Referenz

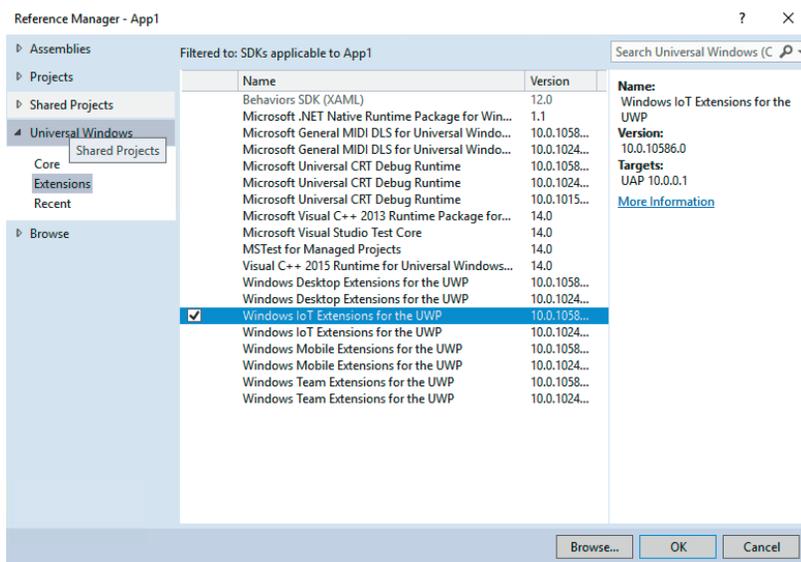


Bild 5.11 Referenz „Windows IoT Extension for the UPW hinzufügen“ (SDK)



HINWEIS: Sofern Ihnen mehrere Varianten der Referenz zur Verfügung stehen, achten Sie darauf, die richtige Version einzubinden. Die richtige Version entspricht der Build-Version von Windows 10, die auf dem Raspberry installiert wurde. Im Zweifel verwenden Sie die neuste verfügbare Version.

5.5.4 Programmaufbau

Auch wenn Sie bereits Erfahrungen in der Softwareentwicklung besitzen, wird das erste Projekt noch Schritt für Schritt erklärt. Bei den späteren Projekten in Kapitel 6 wird dann vor allem auf die spezifischen Besonderheiten eingegangen.

Ein Grundprojekt einer Universal App beinhaltet vier Dateien, die für uns zunächst wichtig sind:

- *App.xaml* und *App.xaml.cs*
- *MainPage.xaml* und *MainPage.xaml.cs*

Es handelt sich dabei um zwei XAML-Seiten mit jeweils der zugehörigen Code-Behind-Datei. Die **App** ist hierbei für die generelle Anwendung verantwortlich; während die **MainPage** das erste Fenster der Anwendung darstellt.

Die Beispiele in diesem Buch kommen fast immer mit einer einzelnen Seite (Page) aus. Es handelt sich somit um Single-Page-Anwendungen. Der größte Teil des Codes wird daher zunächst in der MainPage abgelegt werden.

5.5.4.1 Hinterlegen des Begrüßungstexts und Start-Buttons

Bevor wir uns an die Funktionen der Hardware machen, legen wir den Begrüßungstext an. Wechseln Sie hierzu in die Datei *MainPage.xaml* und ändern den Bereich von Grid wie folgt ab:

```
<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">
  <StackPanel HorizontalAlignment="Center" VerticalAlignment="Center">
    <TextBox x:Name="HelloPi" Text="Mein erstes Raspberry-Projekt!"
  Margin="10" IsReadOnly="True"/>
    <Button x:Name="Start" Content="Start" Margin="10"
  HorizontalAlignment="Center" Click="Start_Click"/>
  </StackPanel>
</Grid>
```

Damit wurde die Anwendung um einen Textbereich und einen Button erweitert. Damit die Anordnung auch optisch ansprechend ist, wurde ein Stackpanel eingebaut. Der Button dient uns später dazu, die Anwendung zu steuern.

Bild 5.12 zeigt, wie die Ausgabe der Anwendung über die HDMI-Schnittstelle am Bildschirm aussehen würde.

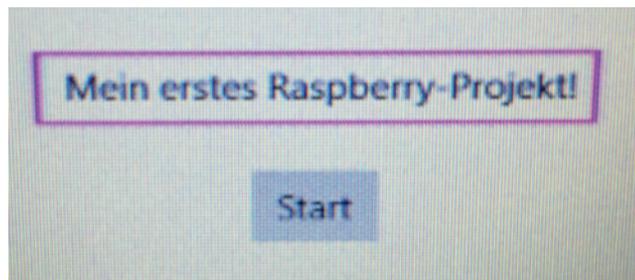


Bild 5.12 Ausgabe der Anwendung über HDMI



TIPP: Sie können das Programm natürlich so umbauen, dass keine Benutzerinteraktion erforderlich ist. In diesem Fall können Sie auch auf den Bildschirm verzichten und das Blinken der LED direkt verfolgen.

5.5.4.2 Timer-Komponente

Damit im Intervall die LED an- und ausgeschaltet werden kann, benötigt man einen Timer. Die Timer sind analog zu Standard-Windows-Projekten zu implementieren.



HINWEIS: Zu beachten ist jedoch, dass die Genauigkeit der Timer nicht sonderlich hoch ist, sie befindet sich in jedem Fall im Millisekunden-Bereich. Logiken, die eine höhere Genauigkeit erfordern, wie z. B. das Initialisieren eines Chips durch Wechsel-signale im Nano-Sekundenbereich, können hiermit nicht abgebildet werden.

Wir verwenden in unserem Beispiel den `DispatcherTimer`³, der in einem Intervall von 500 Millisekunden das Event `Timer_Tick` feuert.

Wenn das Event gefeuert wird, schalten wir die LED an oder aus. Dieser Code wird in der `MainPage.cs` hinterlegt.

Listing 5.3 Verwendung des Windows-Timers

```
public MainPage()
{
    this.InitializeComponent()
    // Timer initialisieren und starten

    timer = new DispatcherTimer();
    timer.Interval = TimeSpan.FromMilliseconds(500);
    timer.Tick += Timer_Tick;
    timer.Start();

    // . . .
}

private void Timer_Tick(object sender, object e)
{
    // . . .
    // Weitere Logik
}
```

5.5.4.3 GPIO

Im nächsten Schritt benötigen wir die Steuerung der GPIO-Pins, damit wir die LED hardwareseitig an- oder ausschalten können. Hierzu muss zunächst das Using in der `MainPage.cs` eingebunden werden:

```
using Windows.Devices.Gpio;
```

³ [https://msdn.microsoft.com/de-de/library/system.windows.threading.dispatchertimer\(v=vs.110\).aspx](https://msdn.microsoft.com/de-de/library/system.windows.threading.dispatchertimer(v=vs.110).aspx)

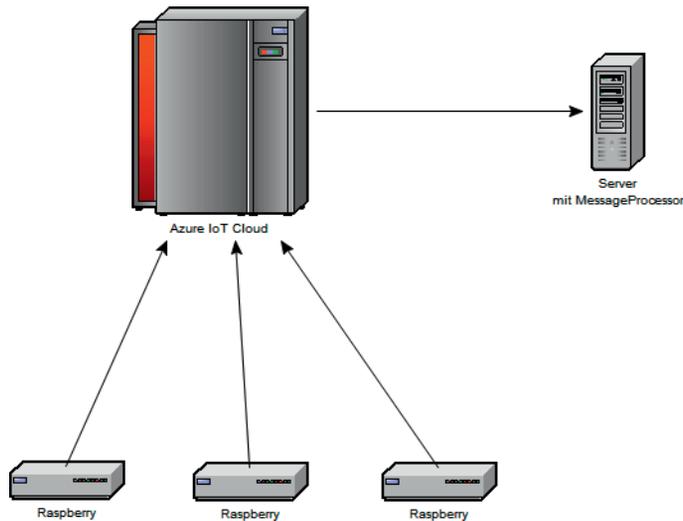


Bild 6.31 Beispielszenario zur Verbindung zwischen Raspberry und Cloud

Die Abrechnung erfolgt in diesem Fall über die Anzahl der versendeten Nachrichten. Microsoft kommt auch hier den Entwicklern entgegen und bietet ein kostenloses Abo an, das für kleinere und mittlere Projekte geeignet ist. Zum Zeitpunkt der Drucklegung war das kostenlose Angebot auf 500 Geräte und 8000 Nachrichten pro Tag limitiert.

Neben den Angeboten zum Nachrichtenaustausch wird Microsoft hier künftig weitere Dienste anbieten, wie z. B. Streaming-Daten oder maschinelles Lernen zur besseren Analyse von Daten. Es ist davon auszugehen, dass das Angebot künftig noch deutlich erweitert wird. Einsteigerprojekte sind unter folgenden Links bei Microsoft erhältlich und dienen als Vorlage für die genannten Beispiele:

- <https://blogs.windows.com/buildingapps/2015/12/09/windows-iot-core-and-azure-iot-hub-putting-the-i-in-iot>
- <https://azure.microsoft.com/de-de/documentation/articles/iot-hub-csharp-csharp-getstarted>

6.7.1 Benötigte Komponenten

Für dieses Projekt wird benötigt:

- Raspberry Pi mit Windows 10 auf SD-Karte und Netzteil
- Netzwerkverbindung über Kabel oder WLAN
- Ein Microsoft Azure-Konto. Sollten Sie noch kein Konto besitzen, so können Sie ein Testkonto⁵ einrichten.

⁵ <https://azure.microsoft.com/de-de/pricing/free-trial>

6.7.2 Hardwareaufbau

Im Projekt wird keine weitere Hardware – neben der Grundausstattung – benötigt. Natürlich kann das Projekt hinsichtlich Hardware beliebig erweitert werden, um Daten auszuwerten (z. B. Temperatursensor) oder Aktionen über die Cloud zu steuern (z. B. Lichtschalter wird von einem anderen Sensor betätigt, der die Helligkeit im Raum misst).

6.7.3 Softwareaufbau

Der Softwareaufbau besteht in diesem Fall aus zwei Blöcken. Einerseits muss die grundlegende Cloud-Infrastruktur im Microsoft Azure-Portal eingerichtet werden. Auf der anderen Seite benötigt man zwei Softwareprojekte, eines für die Raspberries, die die Nachrichten versenden. Das zweite Projekt soll auf einem Server laufen, der die Nachrichten zentral empfangen soll.

6.7.4 Einrichtung der Cloud

Zunächst müssen wir bei Microsoft einen Azure IoT Hub erstellen. Mithilfe des Hubs (sinngemäß „zentrale Stelle“) soll eine sichere Kommunikation zwischen den IoT-Geräten oder einem Backend-Dienst ermöglicht werden.

Als Erstes müssen Sie sich im Azure-Portal⁶ anmelden und dort einen neuen IoT Hub anlegen. Wechseln Sie dazu zunächst auf *Neu*, dann auf *Internet der Dinge* und dann auf *Azure IoT Hub* (Bild 6.32). Im nächsten Fenster muss das Projekt noch weiter eingerichtet werden (Bild 6.32). Hierzu muss zunächst ein Name vergeben werden. Ist dieser gültig und frei, wird ein grünes Häkchen angezeigt.

Bei *Tarif und Skalierung* können Sie den kostenfreien Tarif *F1* wählen. Dieser genügt in jedem Fall für kleine Projekte.

Als Nächstes muss noch eine Ressourcengruppe erstellt und zugewiesen werden. Die Gruppen werden in der Regel dazu verwendet, Ressourcen einer Anwendung (z. B. Datenbank-Server, IoT Hub, virtueller Server) logisch zu bündeln⁷.

Im letzten Schritt muss noch ein Standort zugewiesen werden. Hier empfiehlt es sich, einen Standort zu wählen, der sich geografisch in der Nähe befindet. So können beispielsweise Latenzen niedrig gehalten werden.

⁶ <https://portal.azure.com>

⁷ <https://azure.microsoft.com/de-de/documentation/articles/resource-group-portal>

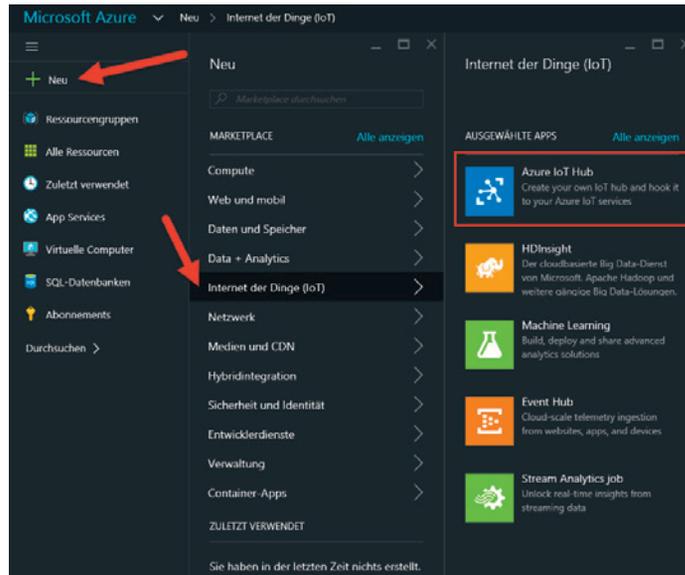


Bild 6.32 Anlegen eines neuen Azure IoT Hub

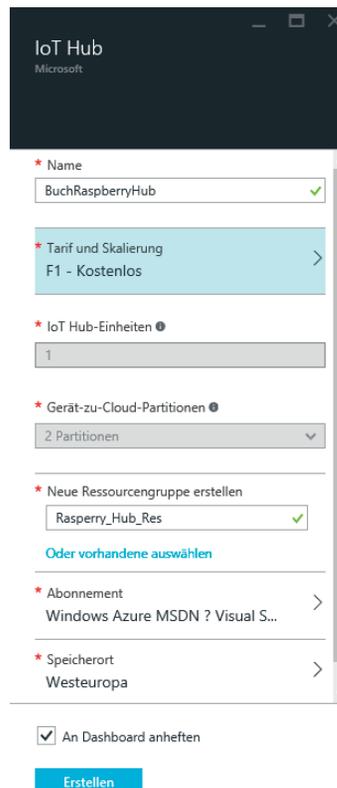


Bild 6.33 Detail-Parameter für den IoT Hub

Stichwortverzeichnis

Symbole

3D-Drucker 167
433-MHz-Sender 105
.NET-Framework 95

A

Abhängigkeiten 92
Akku 135
Analog-Digital-Wandler 117
Applikationen
– anzeigen 60
– entfernen 60
– hinzufügen 60
– Startup-App festlegen 60
Apps 64
AppX 88
Arduino 12, 167
Arduino Sketch 169
Arduino Wiring API 167
ARM 7
Ausgangsspannung 139
Auslastung 65
AVR-Mikrocontroller 167
Azure 32
Azure IoT Hub 142

B

Batterie 135
Benutzer
– anlegen 58
– Passwort ändern 59
– zu Gruppen hinzufügen 58

Betriebsmodus
– Headed 53
– Headless 53
Bildschirmauflösung 62
Bluetooth 69, 157
Bootkonfiguration 61
Bootstrap 105
Breadboard 13
– Virtual Breadboard 16
Breakpoint 88

C

C64 6
CadSoft EAGLE 14
Camera Connector Interface 8
Chip Select 121
CISC 7
Clock 120
Cloud-to-Device 142
Code-Behind 81
Com-Port 12
ConnectionString 149
Controller-Treiber 174

D

DataBinding 76, 109
Datenblatt 106
Datenschutz 39
DC-Motor 165
Debugging 66, 86
Default-App 51
Default Controller Driver
174

Deployment 86
– Authentifizierungsarten 87
DeviceFamily 132
DeviceId 147
Device-to-Cloud 142
Digital Signage 108
Direct Memory Mapped
Driver 174
DISM 47
DispatcherTimer 83
Display Connector 161
Display Serial Interface 8
Domäne 34
Drehmoment 166
Drehzahl 165
DSI 28
DynDNS 142

E

Elektromagnet 166
eneloop 21, 136
Entwicklermodus 72
Error-Report 66

F

Farbcodierung 25
Fast-Ring 42
FileZilla 53
Fritzing 14, 107
Funksteckdose 17, 105

G

GATT 159
 geplante Aufgaben 61
 Geräte-Manager 69
 Gerätetreiber 61
 Gleichstrommotor 165
 GPIO 9f., 83
 Grid 114

H

Hardwaresensor 117
 Hardwareunterstützung 45
 Headless App 98
 Heimautomatisierung 32
 herunterfahren 62
 HID 159
 Home 63
 HTML 96
 http 109

I

I²C 11
 Industrie 4.0 37
 Internet der Dinge 34
 Internet of Things 34
 IoT 31
 iothub-Explorer 149
 ISO-Datei 46

K

Klinkenstecker 159
 Kohleschicht 25
 Kopierwerkzeuge 59

L

LCD 136
 LED 23
 Lego 30
 Leiterplatte 15
 Lightning 170
 Lochrasterplatine 13

M

Magnetschalter 107
 Maker 31
 Manifest 98
 MCP3002 117
 MediaElement 161
 Mehrzeilen-LCD 163
 Memory-Dump 66
 MicroSD 27
 Microsoft Store 86
 Micro-USB 136
 MISO 11
 MIT-Lizenz 96
 MOSI 11
 Motor 164
 Motortreiber 164
 Multimeter 30
 Multitouch 163
 MVVM 76

N

Nachrichten 143
 netduino 30
 Netzwerkkonfiguration 59
 Netzwerkverwaltung 69
 Neustart 62
 Node.js 147

O

Ohmsches Gesetz 18
 On-Screen-Keyboard 140
 Oszilloskop 30

P

Part 107
 Partition 154
 Performance-Tracing 68
 portieren 2
 Powerbank 20, 139
 PowerShell 56
 Prozesse 65
 Prozessverwaltung 59
 Pulsweitenmodulation 165
 putty 54
 Python 2

R

Raspberry
 – Raspberry Pi 2 6
 – Raspberry Pi 3 6
 – Raspberry Pi Zero 6
 Raspberry Foundation 6
 Referenztemperatur 118
 Registry 61
 Relais 96, 106
 REST 105
 Return-on-Investment 35
 RFCOMM 159
 RISC 7

S

Schrittmotor 166
 SCL 12
 SCLK 11
 SDK 157
 SDL 12
 serielle Datenübertragung 159
 Servo-Motor 166
 Shield 161
 Signal
 – High 84
 – Low 84, 103
 Skalierung 142
 Sketch 169
 Slave Select 121
 Slow-Ring 42
 Socket 103
 Spannung 18
 Spannungseinbruch 140
 Spannungsregler 137
 SPI 11, 117, 123
 SPP 159
 Spracherkennung 141
 SSH 54
 stabilisiertes Netzteil 19
 Startup-App 53
 Steckboard 13
 Steckbrücke 30
 Steckerleiste 9
 Step-up-/Step-down 137

Step-up/Step-down Converter 21
Stifteingabe 140
Stimmpaket 161
Stromnetz 17
StromPi 20
Stromstärke 18
Stromversorgung 18
System-on-Chip 37
Systemzeit 155

T

Taktsignal 120
Task-Manager 65
Temperatursensor 117
TextBlock 122
Timer 83
Timer-Genauigkeit 83
TMP36GT9Z 117
Toleranzring 26
Touch-Oberfläche 140
Touchscreen 28, 135, 161
Tracing 67
Treiberunterstützung 157

U

UART 12
Universal Windows Apps 75
unterbrechungsfreie Strom-
versorgung 20
Updates 63
USB-Kartenleser 28
USB-Maus 164
USV 20
UWP 109

V

Verbindungszeichenfolge 146
ViewModel 111
Visual Studio
– Projektvorlagen 73
– Version 72
Vorwiderstand 97

W

Weboberfläche 62
Webserver 98

Widerstand 18, 25, 78
Windows 10 IoT Core 31
Windows Insider 42
Windows IoT Core Watcher
47
Windows IoT Lightning
167
Windows SDK 72
Windows-Update 70
Winkelposition 166
WinPRT 75
WinRT 75
WiringPi 10
WLAN-Verbindung 69
WPF 76
WS2801 11

X

XAML 76, 109
XML 109

Z

Zertifikat 90