WILHELM BÜCHNER HOCHSCHULE

Studienheft

AET01

Technik in der App-Entwicklung Teil 1

AET01

Technik in der App-Entwicklung Teil 1

Dr. Thomas Kalbe

Werden Personenbezeichnungen aus Gründen der besseren Lesbarkeit nur in der männlichen oder weiblichen Form verwendet, so schließt dies das jeweils andere Geschlecht mit ein.

Falls wir in unseren Studienheften auf Seiten im Internet verweisen, haben wir diese nach sorgfältigen Erwägungen ausgewählt. Auf die zukünftige Gestaltung und den Inhalt der Seiten haben wir jedoch keinen Einfluss. Wir distanzieren uns daher ausdrücklich von diesen Seiten, soweit darin rechtswidrige, insbesondere jugendgefährdende oder verfassungsfeindliche Inhalte zutage treten sollten.

Technik in der App-Entwicklung
Teil 1

Inhaltsverzeichnis

0319K06

wbh WILHELM BÜCHNER HOCHSCHULE

Vorwort

Mobilgeräte haben sich in einer fast beispiellosen Geschwindigkeit zu einem nicht mehr wegzudenkenden Alltagsgegenstand entwickelt. Heute verfügt ein großer Anteil der Weltbevölkerung über ein leistungsfähiges Smartphone, ein Tablet oder beides. Welche Entwicklungen hierzu führten und welche Besonderheiten im Hardwareaufbau von Mobilgeräten zu beachten sind, ist Thema des ersten Kapitels dieses Studienhefts.

Ein nicht zu unterschätzender Faktor für den Erfolg der Mobilgeräte sind die speziell zugeschnittenen Softwareplattformen mit einer fast unüberschaubaren Vielzahl von Anwendungen, den „Apps“. Hier haben sich vor allem das iOS-Betriebssystem von Apple sowie das unter der Führung von Google entwickelte Betriebssystem Android durchgesetzt. Im zweiten Kapitel lernen Sie den Aufbau dieser Plattformen kennen und Sie erstellen jeweils ein einfaches „Hello World“-Projekt für iOS und Android.

Ein wesentliches Merkmal von Mobilgeräten sind die vielzähligen Möglichkeiten, mit drahtlosen Netzwerken zu kommunizieren. Kapitel 3 führt Sie zunächst in die physikalischen Grundlagen digitaler Datenübertragung mittels elektromagnetischer Wellen ein. Danach lernen Sie die Organisation der Netzwerkkommunikation in einem Schichtenmodell am Beispiel des OSI-Referenzmodells kennen. Wichtig für das Verständnis moderner Netzwerkarchitekturen sind Kenntnisse über Verfahren, die mehreren Kommunikationsteilnehmern die gemeinsame Nutzung eines Übertragungskanals ermöglichen. Hierzu stellen wir Ihnen die Kanalzugriffsverfahren Frequency Division Multiple Access, ALOHA, Time Division Multiple Access und Code Division Multiple Access gegenüber.

Auf Basis der in Kapitel 3 behandelten Grundlagen der drahtlosen Netzwerkübertragung vermittelt Ihnen das vierte Kapitel den Aufbau und die Funktionsweise digitaler Mobilfunknetze, beginnend bei GSM, dem ersten digitalen Mobilfunknetz, über UMTS bis zum heutigen Mobilfunkstandard der vierten Generation, LTE Advanced. Neben Mobilfunknetzen spielen weitere Funknetze mit zum Teil speziellen Anwendungsgebieten eine Rolle. Als Beispiele für typische Funknetzwerke dieser Art befassen wir uns mit WLAN, Bluetooth und NFC.

Dieses Studienheft hat den Anspruch, ein möglichst breites Wissen zu vermitteln und ausgewählte Themen gegebenenfalls zu vertiefen. Grundkenntnisse der Informatik sowie der Physik auf dem Niveau der Oberstufe sind zum Verständnis der Lerninhalte hilfreich. Ein umfangreiches Sachwortverzeichnis soll Ihnen zudem als Ausgangspunkt für eigene Recherchen dienen.

Beim Bearbeiten dieses Studienhefts wünschen wir Ihnen viel Spaß und Erfolg!

Ihre Studienleitung

1 Übersicht und Aufbau mobiler Endgeräte

Wenn wir heute von mobilen Endgeräten sprechen, meinen wir meist Smartphones oder Tablets. Am Ende dieses Kapitels kennen Sie die wesentlichen Merkmale dieser mobilen Endgeräte und wissen, welche Entwicklungen zu der heute sehr großen Verbreitung mobiler Endgeräte führten.

Sie kennen den Hardwareaufbau von Mobilgeräten, bestehend aus CPU, GPU, Speicher, Display, Netzwerkmodulen, Kameras, Sensoren und Energieversorgung und wissen, welche Besonderheiten die Komponenten von Mobilgeräten aufweisen.

1.1 Merkmale mobiler Endgeräte

Der Begriff „mobiles Endgerät“ oder einfach „mobiles Gerät“ bzw. „Mobilgerät“ bezeichnet eine Hardware-Plattform, die nicht an einen Standort gebunden ist. Im Gegensatz zu einem stationären Desktop PC oder einer Workstation besitzt ein mobiles Gerät eine eigene Stromversorgung und integriert die notwendige Peripherie, um das Gerät sinnvoll zu nutzen. Dazu gehört ein Bildschirm zur Ausgabe sowie eine Eingabemöglichkeit, beispielsweise über Hardware-Tasten oder einen berührungsempfindlichen Bildschirm.

Wenn wir heute von einem mobilen Gerät sprechen, meinen wir meist ein Smartphone oder ein Tablet. Ein **Smartphone** ist ein mobiles Endgerät, das nicht nur zum Telefonieren genutzt werden kann. Aktuelle Plattformen ermöglichen eine Vielzahl weiterer Anwendungen, beispielsweise für die Verarbeitung von Dokumenten, für das Abrufen von Internetseiten oder für die Satelliten-Navigation. Zahlreiche Multimedia-Anwendungen wie Spiele, Musik- und Video-Player sowie Bildverarbeitungsprogramme stehen ebenfalls zur Verfügung.

Ein **Tablet** besitzt einen etwas größeren Bildschirm als ein Smartphone. Den meisten Tablet-Geräten fehlt die Telefoniefunktion. Auch weitere Peripherie-Bausteine, etwa zum Satellitenempfang, fehlen bei Tablets häufig. Auf beiden Geräte-Gattungen finden wir heute jedoch die gleichen Software-Plattformen, sodass Smartphone-Anwendungen mit vergleichsweise wenig Anpassungen auch auf Tablets ausgeführt werden können.

Wir werden deshalb im weiteren Verlauf meist von Smartphones sprechen. Wenn Sie Anwendungen für Tablets planen, beachten Sie bitte die genannten Einschränkungen und Besonderheiten dieser Geräte-Gattung.

Die Merkmale mobiler Endgeräte können wie folgt zusammengefasst werden:

- handliche Größe
- geringes Gewicht (0,1 kg–2,0 kg)
- eigene Stromversorgung
- berührungsempfindlicher Bildschirm (Bildschirmdiagonale ca. 4 bis 6 Zoll[1] bei Smartphones, 7 bis 11 Zoll bei Tablets)
- maßgeschneiderte Softwareplattform
- Multimedia-Unterstützung (Kamera, Sound, Grafik, Spiele)

1. 1 Zoll = 2,54 cm.

- Unterstützung diverser Funknetzwerk-Standards
- Geolokalisierung über Satellitensignale (GPS)
- Sensorik (Beschleunigung, Neigung, Kompass etc.)

Auf einige dieser Merkmale werden wir in diesem und den kommenden Heften noch näher eingehen. Zuvor befassen wir uns kurz mit der geschichtlichen Herkunft von mobilen Geräten.

Übung 1.1:

Weshalb fehlen Tablets die Telefoniefunktion und weitere Bausteine? Welche Vorteile können Tablets stattdessen gegenüber Smartphones ausspielen?

1.2 Geschichte mobiler Endgeräte

Die Wurzeln heutiger Smartphones lassen sich auf die Ideen des Forschers Alan Kay zurückführen. Alan Kay entwickelte in den Siebzigerjahren am Forschungsinstitut XEROX Parc in Palo Alto, Kalifornien, das Konzept für einen tragbaren Computer mit integriertem Bildschirm in Notizbuchgröße, das „Dynabook“.[2]

Das Dynabook sollte speziell auf mobile Anwendungen zugeschnittene Bedienkonzepte ermöglichen und zu einem möglichst günstigen Preis (ca. $ 500) angeboten werden. Die Vision Alan Kays ließ sich mit damaliger Technik nicht vollständig realisieren. Einige der Ideen zum Dynabook sind jedoch in einen der ersten Personal Computer eingeflossen, den Palo Alto. Dessen grafische Benutzeroberfläche diente später als Vorbild für das Apple-Macintosh-Betriebssystem und Microsoft Windows.

In den frühen Achtzigerjahren kamen die ersten tragbaren Computer in den Handel, wie beispielsweise der Osborne 1.[3] Diese Computer besaßen jedoch nur wenig Ähnlichkeit mit heutigen Smartphones oder Alan Kays Vision eines „tragbaren Computers in Notizbuchgröße“. Stattdessen handelte es sich um schwere Geräte ohne eigene Stromversorgung und ohne speziell auf Mobilanwendungen zugeschnittene Bedienkonzepte oder Betriebssysteme.

Mitte der Neunzigerjahre kamen Geräte in den Handel, die schon eher an die heutigen Tablets erinnern. Diese Geräte wie das Apple Newton, der Palm Pilot oder auch der Pocket PC von Microsoft[4], wurden als **PDA** (Personal Digital Assistant) bezeichnet.

Die PDAs waren mit einem berührungsempfindlichen Flüssigkristall-Bildschirm (LCD: Liquid Crystal Display) ausgestattet, der über einen Stift bedient wurde. Die Einsatzmöglichkeiten der PDAs waren aufgrund der damaligen Hardwarebeschränkungen, einem geringen Softwareangebot und vor allem fehlender Netzwerkinfrastruktur stark eingeschränkt. Die Verkaufszahlen von PDAs waren dementsprechend niedrig.

2. https://de.wikipedia.org/wiki/Dynabook
3. https://de.wikipedia.org/wiki/Osborne_1
4. Newton: https://de.wikipedia.org/wiki/Newton_(PDA),
   Palm Pilot: https://de.wikipedia.org/wiki/Palm_Pilot,
   Pocket PC: https://de.wikipedia.org/wiki/Microsoft_Pocket_PC

Mit der Digitalisierung und dem Ausbau der Mobilfunknetze in den Neunzigerjahren wurde das mobile Telefonieren, eine bis dahin schwerfällige und teure Technik, für ein breites Publikum interessant und bezahlbar (nach dem statistischen Bundesamt überstieg die Anzahl der Mobiltelefonnummern die Anzahl der Festnetzanschlüsse in Deutschland erstmals im Jahr 2013). Die ersten Mobiltelefone eigneten sich jedoch vorwiegend zum Telefonieren und besaßen nur wenige einfache, fest installierte Programme wie z.B. Adressbücher oder Taschenrechner.

Wenige Jahre nach Beginn der Mobilfunkära kamen bereits Geräte mit einer komplexeren Hardwareausstattung und leistungsfähigeren Betriebssystemen auf den Markt. Als Beispiele seien **Symbian OS** (Hersteller u.a. Nokia, ca. 2002) und **Windows Mobile** (2003) genannt. Diese Systeme konnten sich aufgrund mangelnder Bedienbarkeit, Anpassung an mobile Gegebenheiten und einem geringen Softwareangebot jedoch nicht durchsetzen.

Im Januar 2007 stellte Apple die erste Version des **iPhone** vor. Dieses Gerät war mit einem **Multitouch**-Display ausgestattet und ließ sich über Mehrfingergesten intuitiv bedienen. Mit **iOS** entwickelte Apple ein leistungsfähiges und speziell auf Mobilanwendungen zugeschnittenes Betriebssystem, wofür in relativ kurzer Zeit ein großes Softwareangebot zur Verfügung stand.

Ende 2008 folgten die ersten Mobilgeräte mit dem Betriebssystem **Android** von Google mit einer sehr ähnlichen Hardware und ähnlichen Bedienkonzepten. Im Gegensatz zu iOS, das sich auf Geräte der Firma Apple beschränkt, steht Android auch anderen Herstellern zur Verfügung.

Die beiden Systeme iOS und Android sind bis heute ein großer kommerzieller Erfolg. Schätzungen zufolge wurden bis Ende 2014 etwa drei Milliarden Android-Geräte und über eine Milliarde Geräte mit iOS verkauft. Wenn wir heute von „mobilen Geräten“ sprechen, meinen wir deshalb in erster Linie Geräte mit iOS oder Android. Andere Systeme sind entweder vom Markt verschwunden oder konnten sich dagegen bis heute noch nicht wirklich erfolgreich durchsetzen (vgl. auch Abschnitt 2.1).

Übung 1.2:

Welche Entwicklungen führten zu den heutigen Smartphones?

1.3 Hardware und Aufbau mobiler Endgeräte

Die Hardware eines modernen mobilen Endgeräts besteht üblicherweise aus den folgenden Komponenten:

- Hauptprozessor (CPU)
- Grafikprozessor (GPU)
- Haupt- und Festspeicher
- Multitouch-Display
- Kommunikations- und Netzwerkmodule
- Kameras
- Sensoren (z.B. Beschleunigung, Neigung, Kompass)

- Energieversorgung und Energiemanagement

Damit diese Komponenten in ein kleines Gehäuse passen, müssen sie sehr kompakt sein. Der Hauptprozessor wird daher meist mit anderen Komponenten (z.B. GPU, Hauptspeicher, Sensordatenverarbeitung) zu einem Modul, dem **SOC** (System on Chip), zusammengefasst.

Bevor wir uns mit den einzelnen Komponenten näher befassen, betrachten wir noch kurz die Entwicklung der Mobilgeräte-Hardware am Beispiel des iPhone (vgl. Tab. 1.1). Die erste iPhone-Generation erschien 2007 und war mit einem 32 Bit Einzelkern-Prozessor mit 412 MHz Taktfrequenz ausgestattet. Das 2017 erschienene iPhone X besitzt einen 64 Bit Sechskernprozessor mit bis zu 2,39 GHz Taktfrequenz. Die Displayauflösung hat sich von 320 x 480 Bildpunkte auf 1125 x 2436 Bildpunkte fast vervierfacht, die Auflösung der Kamera von 2 Megapixel auf 12 Megapixel versechsfacht. Seit dem iPhone 4 gehört ein GPS-Empfänger zur Standardaustattung.

Tab. 1.1: iPhone-Generationen im Vergleich

Jahr	Modell	Hauptprozessor	Display	Kamera	GPS
2007	iPhone	32 Bit 412 MHz Single Core	320 x 480	2 MP	Nein
2008	iPhone 3g	32 Bit 412 MHz Single Core	320 x 480	2 MP	Nein
2010	iPhone 4	32 Bit 800 MHz Single Core	640 x 960	5 MP	Ja
2012	iPhone 5	32 Bit 1.3 GHz Dual Core	640 x 1136	8 MP	Ja
2014	iPhone 6	64 Bit 1.4 GHz Dual Core	750 x 1334	8 MP	Ja
2015	iPhone 6s	64 Bit 2.0 GHz Dual Core	750 x 1334	12 MP	Ja
2016	iPhone 7	64 Bit 2.34 GHz Quad Core	750 x 1334	12 MP	Ja
2017	iPhone X	64 Bit 2.39 GHz Hexa Core	1125 x 2436	12 MP	Ja

Die Leistungsfähigkeit der Mobilgeräte mit einem Android-Betriebssystem nimmt ebenfalls stetig zu. Das aktuelle Top-Modell von Samsung beispielsweise, das „Galaxy S9 “, besitzt einen leistungsfähigen Prozessor mit acht Rechenkernen, einen Bildschirm mit einer Auflösung von 1440 x 2960 Bildpunkten und eine Kamera mit 12 Megapixel Auflösung.

1.3.1 Hauptprozessor (CPU)

Moderne Mobilgeräte verfügen üblicherweise über eine **CPU** (Central Processing Unit, Hauptprozessor) mit einer RISC-Architektur. Die Abkürzung **RISC** bedeutet „Reduced Instruction Set Computer“ und grenzt diese Prozessorarchitektur gegen die in Desktop-PCs üblicherweise verwendeten Prozessoren mit **CISC**-Architektur (Complex Instruction Set Computer) ab.

RISC-Prozessoren besitzen eine große Anzahl von **Registern** (schnelle Zwischenspeicher auf dem Chip), um langsame Hauptspeicherzugriffe zu vermeiden. Der Befehlssatz eines RISC-Prozessors besteht aus vergleichsweise einfachen Befehlen, die einheitlich strukturiert sind und die gleiche Laufzeit aufweisen. Dadurch kann der Befehlsdurchsatz

des Prozessors einfacher durch Techniken wie **Pipelining** optimiert werden. Der Energieverbrauch lässt sich ebenfalls leichter vorhersagen und kontrollieren. Komplexe Operationen, für die ein CISC-Prozessor möglicherweise eigene Befehle besitzt, müssen in einem RISC-Prozessor allerdings durch eine Reihe einfacher Befehle nachgebildet werden.

Die meisten Mobilgeräte sind mit ARM-Prozessoren ausgestattet. **ARM** (Advanced RISC Machines) ist ursprünglich eine Entwicklung der britischen Firma Acorn und wird derzeit von der Firma ARM Limited weiterentwickelt und an verschiedene Hersteller wie Apple, Samsung, HTC oder NVidia lizenziert. Die erste Prozessorgeneration mit ARM-Architektur (ARMv1), ein stromsparender 32 Bit Einzelkern-Prozessor mit 8 MHz Takt, erschien 1985. Die derzeit aktuelle Version 8 der ARM-Architektur (ARMv8) spezifiziert einen 64 Bit Prozessorkern mit bis zu 3 GHz Takt.

Heutige Mobilgeräte besitzen mehrere ARMv8-Kerne integriert auf einem System on Chip. Aktuelle Beispiele sind die SOCs „A11“ (Apple, sechs Kerne, davon zwei auf Hochleistung und vier auf Energieeffizienz ausgerichtet) und „Exynos 9“ (Samsung, bis zu acht Kerne).

1.3.2 Grafikprozessor (GPU)

Moderne Mobilgeräte besitzen eine **GPU** (Graphics Processing Unit, Grafikprozessor), um die CPU bei grafiklastigen Aufgaben zu entlasten. Neben 3D-Beschleunigung in Spielen und Decodierung von komprimierten Videoformaten sorgt die GPU für einen flotten Aufbau der Benutzeroberfläche der Anwendungen. Die GPU eines Mobilgeräts ist häufig auf dem SOC integriert.

Die **Fill Rate** (deutsch: Füll-Rate) ist ein Maß für die Leistungsfähigkeit einer GPU und beschreibt, wie viele Bildschirm-Pixel pro Sekunde eingefärbt werden können. Mit zunehmender Display-Auflösung der Mobilgeräte (vgl. Tab. 1.1) muss auch die Fill Rate der GPU entsprechend ansteigen. Moderne Mobilgeräte-GPUs verfügen über eine Fill Rate zwischen 3,5 Milliarden Pixel (iPhone 6) und über 7 Milliarden Pixel (iPad Air 2) pro Sekunde.

Im Gegensatz zu einem Desktop-Computer muss eine GPU für Mobilgeräte einen besonders niedrigen Energiebedarf aufweisen. Die Architektur dieser GPUs unterscheidet sich daher meist wesentlich von den Desktop-GPUs.

Einen großen Marktanteil unter den Mobilgeräte-GPUs besitzen derzeit die Adreno-Architektur der Firma Qualcomm sowie die PowerVR-GPU der Firma Imagination. Mit dem A11 Bionic-SOC setzt Apple 2017 erstmals auf eine selbstentwickelte GPU. Neben Grafik-Anwendungen ist diese GPU auch für KI-Aufgaben wie Sprach- oder Bilderkennung optimiert.

Übung 1.3:

Wie hoch muss die Fill Rate einer mobilen GPU mindestens sein, um eine Anwendungsoberfläche auf einem Display mit 750 mal 1334 Bildpunkten mit einer Bildwiederholfrequenz von 60 Hz (60 Bilder pro Sekunde) anzuzeigen?

Übung 1.4:

Weshalb ist die Fill Rate eines Tablet-Computers oft höher als bei einem Smartphone?

1.3.3 Haupt- und Festspeicher

Moderne Mobilgeräte verfügen über einen Hauptspeicher mit einer Kapazität zwischen 512 MByte und 6 GByte.[5] Das iPhone X beispielsweise besitzt einen 3 GByte großen Hauptspeicher. Hierbei handelt es sich üblicherweise um **SDRAM** (Synchronous Dynamic Random Access Memory) im Standard **DDR4** (Double Data Rate 4). Der gleiche Speicher-Typ wird auch in Desktop-Computern eingesetzt, allerdings ist der Speicher auf Mobilgeräten häufig auf dem SOC integriert.

Zusätzlich zum flüchtigen Hauptspeicher verfügen Mobilgeräte noch über einen persistenten (d.h. nicht flüchtigen) Speicher, meist vom Typ **NAND-Flash**, mit Kapazitäten zwischen 2 und 256 GByte auf Smartphones und bis zu 1 TByte auf Tablets. Ein Flash-Speicher besteht aus einer Kontrolllogik und den Speicherzellen. Die Speicherzellen sind ähnlich wie MOSFET-Transistoren aufgebaut, besitzen jedoch ein zusätzliches Gate (als „Gate“ wird die Steuerleitung eines Transistors bezeichnet), das **Floating Gate**. Durch eine spezielle elektrische Isolierung kann eine elektrische Ladung im Floating Gate permanent gespeichert werden.

NAND-Flash im derzeit üblichen Standard **EMMC** 5.1 (Embedded Multimedia Card) erzielt eine Datentransferrate von 250 MByte/s lesend bzw. 125 MByte/s schreibend und ist um etwa eine Größenordnung langsamer als DDR4-SDRAM. Die Entwicklung von Flash-Speichern schreitet allerdings rasant voran. Der in Samsungs Smartphone Galaxy S6 von 2015 eingesetzte Flash-Speicher im Standard **UFS** 2.0 (Universal Flash Storage) erzielt bereits Transferraten von über 1 GByte/s. Der für 2018 angekündigte Nachfolger UFS 3.0 soll die doppelte Datenrate von UFS 2.0 erzielen.

Einige Android-Geräte sind mit einem zusätzlichen Kartenslot für SD- oder Micro-SD-Karten ausgestattet, manche Android-Tablets besitzen sogar einen USB-Anschluss.

Übung 1.5:

Was ist ein Floating Gate?

1.3.4 Touch-Displays

Mobilgeräte werden üblicherweise hauptsächlich über ein berührungsempfindliches Display (auch „Touchscreen“ oder „Touchpad“) bedient. Hierbei unterscheiden wir zwei Techniken, die **resistiven Displays** und die **kapazitiven Displays**.

Resistive Displays reagieren auf mechanischen Druck („resistiv“: auf Druck reagierend), der durch einen Finger oder einen beliebigen anderen, genügend festen Gegenstand hergestellt werden kann. Für präzisere Eingaben wird ein spezieller kugelschreiberartiger Stift (auch „Stylus“ genannt) verwendet.

5. 1 MByte = 2^{20} Byte, 1 Gbyte = 2^{30} Byte.

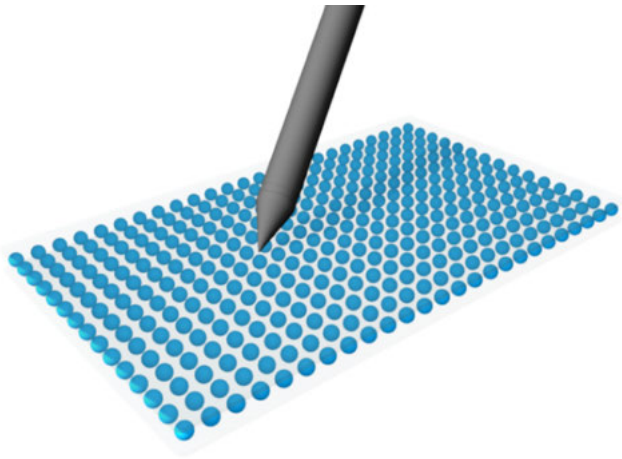


Abb. 1.1: Resistiver Touchscreen mit Spacer Dots (blau) und Stylus (grau)

Eine schematische Darstellung eines resistiven Displays zeigt Abb. 1.1. Das Display besteht aus zwei leitfähigen Schichten, die durch kleine Abstandshalter („Spacer Dots“) voneinander getrennt sind. Die obere Schicht besteht aus einem dehnbaren Polyester-Material, die untere Schicht ist auf eine stabile Grundplatte aufgetragen.

Um die *x*-Koordinate der Druckstelle zu bestimmen, wird am linken Rand der oberen Schicht eine elektrische Spannung angelegt, die nach rechts hin abnimmt. An der Druckstelle berühren sich die beiden Schichten. Durch Messung der sich daraus ergebenden Spannung an der unteren Schicht kann die *x*-Koordinate berechnet werden. Die Bestimmung der *y*-Koordinate erfolgt mit einer kurzen Verzögerung von einigen Millisekunden. Hierzu wird eine Spannung am oberen Rand der oberen Schicht angelegt, die nach unten hin abnimmt.

Resistive Displays besitzen einige Nachteile. So können sich nach einiger Zeit durch die mechanische Belastung Risse in der Polyesterschicht bilden. Wischgesten, etwa zum Scrollen eines größeren Display-Bereichs, sind mit einem resistiven Display kaum zu realisieren. Sie können auch nur mit einem Finger bedient werden, da ein gleichzeitiger Druck mit zwei Fingern nur die Kontaktfläche der Schichten vergrößert.

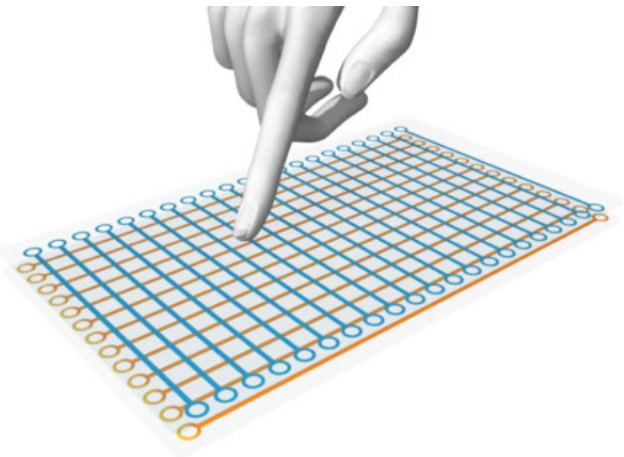


Abb. 1.2: Kapazitiver Touchscreen mit Zeilenelektroden (blau) und Spaltenelektroden (orange)

Aus diesen Gründen werden heute vor allem kapazitive Touchscreens eingesetzt. Eine schematische Darstellung zeigt Abb. 1.2. Zwischen zwei Glasplatten befindet sich ein Gitternetz aus Elektroden. Die senkrechte Elektrodenschicht ist von der waagerechten Elektrodenschicht durch eine isolierende Schicht (in Abb. 1.2 nicht eingezeichnet) abgetrennt. Die isolierende Schicht zwischen den Elektroden dient als **Dielektrikum**. Somit besitzt jeder Kreuzungspunkt die elektrischen Eigenschaften eines Kondensators.

Durch die elektrische Leitfähigkeit der menschlichen Haut fließt bei Berührung der Display-Oberfläche eine elektrische Ladung ab, wodurch sich die Kapazität des Kondensators am Kreuzungspunkt ändert. Eine Steuerungselektronik misst permanent die Kapazität jedes einzelnen Kondensators und kann dadurch ermitteln, ob eine Berührung am jeweiligen Kreuzungspunkt vorliegt. Kapazitive Displays sind somit in der Lage, gleichzeitige Berührungen mit mehr als einem Finger zu erkennen.

Neuartige Displays verwenden zusätzliche Komponenten, um die Stärke einer Berührung auszuwerten. Apple nennt diese erstmals im iPhone 6s verbaute Technologie „3D Touch“.

Übung 1.6:

Weshalb können Sie ein kapazitives Display nicht bedienen, wenn Sie einen Handschuh tragen?

Übung 1.7:

Warum funktioniert Scrolling mit einem kapazitiven Display besser als mit einem resistiven Display?

1.3.5 Kommunikations- und Netzwerkmodule

Ein wesentliches Merkmal moderner Mobilgeräte sind die umfangreichen Möglichkeiten, mit Mobilfunk- und Drahtlosnetzen in Verbindung zu treten. Wir werden die hierbei eingesetzten Techniken und Netzwerkstandards in den späteren Kapiteln dieses Studienhefts noch genauer betrachten und uns an dieser Stelle auf eine kurze Übersicht beschränken.

Zu den unterstützten Mobilfunkstandards zählen heute üblicherweise **GSM** (Global System for Mobile Communications), **GPRS** (General Packet Radio Service), **EDGE** (Enhanced Data Rates for GSM Evolution), **UMTS** (Universal Mobile Telecommunications System), **HSPA** (High Speed Packet Access) und **LTE** (Long Term Evolution) bzw. **LTE Advanced** (vgl. Tab. 1.2).

GSM war der erste digitale Funknetzstandard und löste 1992 das analoge C-Netz ab. GSM wird somit als Mobilfunknetz der 2. Generation, kurz **2G**, bezeichnet. Die GSM-Erweiterungen GPRS und EDGE werden als Zwischenstufe zur dritten Generation unter **2,5G** zusammengefasst. UMTS ist ein Mobilfunknetz der dritten Generation (**3G**) und erlangte in Deutschland einen hohen Bekanntheitsgrad aufgrund der Versteigerung verfügbarer Frequenzen durch die Bundesnetzagentur im Jahr 2000. HSPA ist eine Erweiterung von UMTS (**3,5G**). Der derzeit schnellste Mobilfunkstandard ist LTE (**3,9G**) bzw. LTE Advanced (**4G**). Der Ausbau des LTE-Netzes begann in Deutschland im Jahr 2010. Der folgende Standard **5G** ist 2018 bereits im Testbetrieb und soll 2020 weltweit verfügbar sein.

Tab. 1.2: Übersicht aktueller Mobilfunkstandards

Standard	Generation	Datenrate[a)]
GSM	2G	9,6 kbit/s
GPRS	2,5G	53,6 kbit/s
EDGE	2,5G	236,8 kbit/s
UMTS	3G	384 kbit/s
HSPA	3,5G	42 Mbit/s
LTE	3,9G	300 Mbit/s
LTE Advanced	4G	1000 Mbit/s
5G	5G	3500 Mbit/s

a) 1 kbit: 10^3 Bit, 1 Mbit: 10^6 Bit

Jeder Mobilfunkstandard verwendet ein eigenes **Protokoll**, das detailliert beschreibt, in welcher Art und Weise die Daten übertragen werden. Der **Baseband-Prozessor** ist auf einem Smartphone für die Umsetzung dieser Protokolle zuständig. Moderne 4G-Baseband-Prozessoren unterstützen üblicherweise auch die älteren Mobilfunkstandards. Zum Senden und Empfangen von Signalen in Form elektromagnetischer Wellen (vgl. Abschnitt 3.1) dient ein **Transceiver**-Baustein. Das Wort „Transceiver“ setzt sich zusammen aus „Transmitter“ (Sender) und „Receiver“ (Empfänger).

Die Mobiltelefonie ist auf weitere Infrastruktur angewiesen. Dazu gehören Mikrofon und Lautsprecher, Analog-Digital- und Digital-Analog-Wandler zur Umwandlung analoger (Ton-)Signale in digitale Daten und zurück, sowie hinreichende Prozessorkapazität, um Audiodaten zu komprimieren und zu dekomprimieren. Diese Komponenten werden nicht nur exklusiv für die Mobiltelefonie genutzt, sondern beispielsweise auch für Multimedia-Anwendungen oder zur Spracherkennung.

Neben den Mobilfunknetzen spielen weitere Drahtlosnetze eine Rolle. Der Empfang von Satellitensignalen zur Ortsbestimmung mittels **GPS** (Global Positional System) gehört heute zur Standardausstattung jedes Smartphones. Mobilgeräte können im drahtlosen Netzwerkstandard **IEEE 802.11**, auch **WLAN** (Wireless Local Area Network) oder **WiFi** genannt, kommunizieren. Hinzu kommt die Unterstützung diverser Standards zum kontaktlosen Datenaustausch über kurze Distanzen (wenige Zentimeter bis einige Meter) wie **Bluetooth** und **NFC** (Near Field Communication). Auch für diese Art von Drahtlosnetzen finden sich in Mobilgeräten häufig Module, die gleich mehrere Netzwerkstandards unterstützen.

1.3.6 Kameras

Moderne Mobilgeräte verfügen heute in der Regel über mindestens eine Kamera für Bild- und Videoaufnahmen. Häufig findet sich eine zweite Kamera mit einer etwas geringeren Auflösung auf der Vorderseite für Videotelefonieanwendungen und „Selfies“.

Bei einer Digitalkamera wird das einfallende Licht über eine Linse auf einen Bildsensor gebündelt. Der Bildsensor besteht aus einem dicht gepackten Gitter aus Fotodioden. Eine **Fotodiode** ist ein Halbleiterelement, das bei Lichteinstrahlung Ladung abgibt. Die abgegebene Ladung wird in einem Kondensator gesammelt und somit kann die Intensität des einfallenden Lichts gemessen werden.

Bei digitalen Bildsensoren unterscheiden wir zwei Schaltungsprinzipien: **CCD** (Charge Coupled Device) und **APS** (Active Pixel Sensor). CCD-Sensoren lesen die Ladungen der einzelnen Bildpunkte seriell aus, während ein APS die Ladungen aufgrund einer komplexeren Schaltung parallel ermittelt und somit höhere Bildraten als ein CCD-Sensor erzielt.

Die Hersteller werben meist mit einer hohen Auflösung der Kameras in **Megapixel** (Millionen Bildpunkte). Diese Zahl dient jedoch nur eingeschränkt als Maß für die Bildqualität einer Kamera, denn bei gleichbleibender Sensorgröße müssen die Fotodioden für eine höhere Auflösung dichter gepackt werden. Kleinere Fotodioden sind jedoch schneller gesättigt (d. h. sie geben keine weitere Ladungen mehr ab), was den Dynamikumfang der Diode verkleinert.

Ein weiterer nicht zu unterschätzender Faktor bei der Bildqualität ist die verwendete Optik. Der begrenzte Platz auf einem Mobilgerät erlaubt nur sehr einfache und kleine Objektive.

1.3.7 Sensoren

Moderne Mobilgeräte besitzen eine Vielzahl von Sensoren, die interessante Anwendungen ermöglichen. Beschleunigungssensoren gehören inzwischen zur Standardausstattung jedes Mobilgeräts. Ein solcher Sensor wird als **Accelerometer** bezeichnet und ermittelt die lineare Beschleunigung eines Körpers aufgrund der Trägheit von Massen. In dem Sensorgehäuse ist eine Testmasse an einer Feder aufgehängt. Wirkt eine Beschleunigung auf den Sensor, so setzt die Testmasse aufgrund ihres Trägheitswiderstands eine Trägheitskraft entgegen, deren Größe über die Federlänge gemessen werden kann.

Die Einheit der Beschleunigung ist Meter pro Sekunde zum Quadrat (m/s^2) und wird oft in Vielfachen der Erdbeschleunigung g angegeben. Ein g entspricht etwa $9{,}81\frac{m}{s^2}$.

Die in Mobilgeräten eingebauten Sensoren verwenden anstelle einer Feder häufig ein **piezoelektrisches Material**. Ein solches Material besteht aus speziellen Kristallen oder Keramiken, die eine einwirkende mechanische Kraft in elektrische Spannung umwandeln. Das Grundprinzip eines piezoelektrischen Accelerometers zeigt Abb.1.3. In Abhängigkeit der Richtung einer auf den Sensor einwirkenden Beschleunigung wird das Piezoelement entweder gestreckt oder gestaucht, was eine Spannungsänderung zur Folge hat.

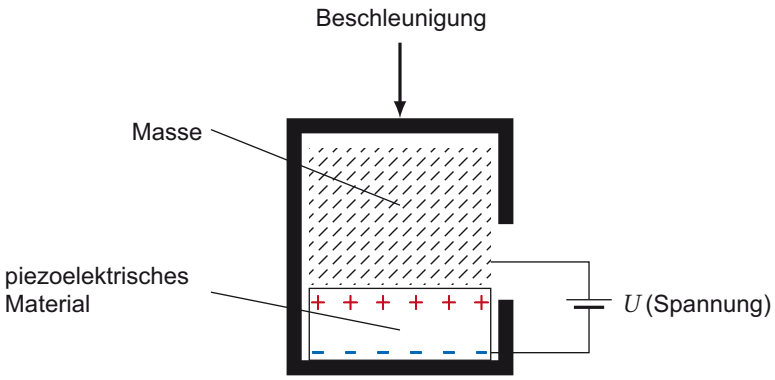


Abb. 1.3: Piezoelektrischer Beschleunigungssensor

Ein einzelner Accelerometer misst die Beschleunigung entlang einer Raumachse. Mobilgeräte enthalten daher drei solcher Sensorelemente (meist in einem Bauteil integriert), um die Beschleunigung entlang aller drei Raumachsen zu ermitteln.

Beschleunigungssensoren können beispielsweise zur Steuerung von Anwendungen eingesetzt werden: Ein Klopfen oder Schütteln des Geräts löst eine Aktion aus, wie z. B. das Weiterschalten eines Lieds. Sie eignen sich auch als Bewegungs-Tracker: Eine nach „oben“ wirkende Beschleunigung wird als Schritt interpretiert.

Ein sehr typischer Anwendungsfall von Beschleunigungssensoren ist die **Lageerkennung**: Hierzu wird ein 3D-Koordinatensystem festgelegt, das der Bildschirmhorizontalen die *x*-Achse und der Bildschirmvertikalen die *y*-Achse zuweist. Die *z*-Achse steht senkrecht auf dem Display (vgl. Abb. 1.4). Dieses Koordinatensystem stimmt mit der Ausrichtung der drei Beschleunigungssensoren überein.

Liegt das Smartphone flach auf einem Tisch, wirkt eine Beschleunigung von 1 *g* in *z*-Richtung. Die Beschleunigungskomponenten in *x*- und *y*-Richtung sind null. Entsprechend lässt sich feststellen, ob das Gerät hochkant oder quer gehalten wird, und das Gerät kann auf das jeweilige Bildschirmformat umschalten.

In gewissen Grenzen können die Beschleunigungssensoren auch zur genaueren Orientierungsbestimmung des Geräts im Raum genutzt werden. Hierzu werden anhand der drei Beschleunigungskomponenten die Drehwinkel an den drei Achsen über einfache Trigonometrie bestimmt. Diese Drehwinkel werden auch mit den in der Luftfahrt üblichen Begriffen **Pitch**, **Roll** und **Yaw** bezeichnet (vgl. Abb. 1.4).

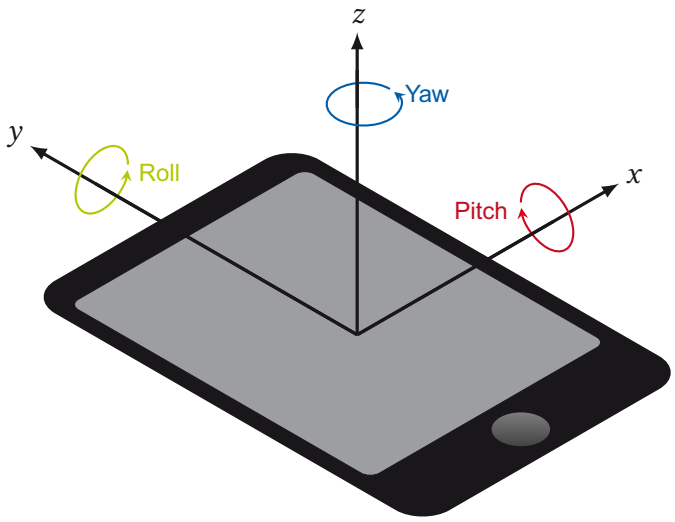


Abb. 1.4: 3D-Koordinatensystem und Drehwinkel Pitch, Roll und Yaw

Es ist jedoch nicht möglich, allein mit Beschleunigungssensoren alle drei Raumwinkel robust zu bestimmen. Liegt das Gerät beispielsweise flach auf einem Tisch (Pitch und Roll sind null), ist der Winkel um die *z*-Achse (Yaw) unbekannt. Ein weiteres Problem ist, dass die Lagebestimmung nur dann funktioniert, wenn das Gerät weitgehend in Ruhelage ist. Falls eine weitere Beschleunigung auf das Gerät einwirkt, kann diese nicht mehr von der Erdbeschleunigung unterschieden werden. Bei einem fallenden Gerät messen alle drei Sensoren zudem einen Betrag von null.

Übung 1.8:

Angenommen, Sie halten das Mobilgerät hochkant vor Ihrer Brust. Welcher Winkel kann in diesem Fall nicht bestimmt werden?

Daher sind in vielen Mobilgeräten inzwischen zusätzliche Bewegungssensoren, die Gyroskope, eingebaut (das erste Smartphone mit einem 3-Achsen Gyroskop war das iPhone 4). Ein **Gyroskop** dient zur Messung der Geschwindigkeit einer Drehung um eine bestimmte Achse. Diese Geschwindigkeit wird als **Winkelgeschwindigkeit** bezeichnet und üblicherweise in Radiant pro Sekunde (rad/s) angegeben.[6]

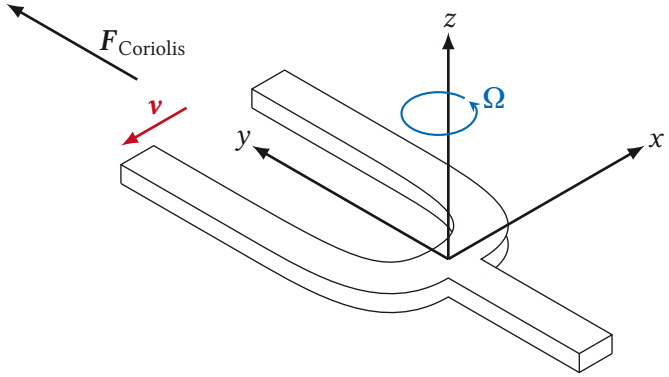


Abb. 1.5: Vibrationskreisel

6. 1 rad = 360° / 2· π

Das Grundprinzip eines Gyroskops basiert auf der Drehimpulserhaltung einer rotierenden Masse und ist bereits seit über 200 Jahren bekannt. Gyroskope in Form von Kreiseln werden beispielsweise zur Orientierungsbestimmung in Flugzeugen eingesetzt.

Anstelle von Kreiselgyroskopen mit rotierenden Scheiben sind die Gyroskope in Smartphones jedoch als **Vibrationskreisel** ausgelegt. Hierbei handelt es sich um eine schwingende Masse, vergleichbar mit einer Stimmgabel (vgl. Abb. 1.5). Wird diese Masse mit einer Winkelgeschwindigkeit Ω um die Achse senkrecht zur Schwingungsachse rotiert, entsteht aufgrund des **Coriolis**-Effekts eine zu diesen beiden Achsen senkrechte Kraft. Diese Kraft ist proportional zur Masse m, der Geschwindigkeit $\boldsymbol{v}$ der Schwingung sowie der Winkelgeschwindigkeit Ω und berechnet sich zu

$$F_{\text{Coriolis}} = -2\, m\Omega \times v$$

Vibrationskreisel werden heute in mikroskopisch kleinen Strukturen, den **MEMS** (Microelectromechanical Systems), günstig gefertigt und dienen beispielsweise zur Steuerung von Spielen oder zur Stabilisierung verwackelter Bildaufnahmen.

Gyroskope und Accelerometer messen unterschiedliche Größen: Ein Gyroskop misst eine Rotationsgeschwindigkeit, während ein Accelerometer die Beschleunigung in einer vorgegebenen Richtung bestimmt. Sie messen jedoch keine absoluten Positionen oder Winkel, sondern ihre Ableitungen bzw. Änderungsraten: Die Beschleunigung ist die zweite Ableitung der Position und die Winkelgeschwindigkeit ist die erste Ableitung des Winkels. Um zu einem Zeitpunkt die tatsächliche Lage im Raum zu ermitteln, müssen diese Werte ausgehend von einer gegebenen bekannten Position über die Zeit aufintegriert werden. Diese **Zeitintegration** ist jedoch insbesondere bei sehr schnellen Bewegungen sowie über einen längeren Zeitraum nur begrenzt genau.

Wenn ein Gyroskop nur Änderungsraten erkennen kann, wie ermitteln wir nun den (absoluten) Yaw-Winkel bei einem Mobilgerät, das flach auf einem Tisch liegt (vgl. Abb. 1.4)? Hierzu kann ein weiterer Sensor, der **Magnetometer**, eingesetzt werden. Dabei handelt es sich im Prinzip um einen Kompass, der die Richtung zum magnetischen Nordpol anzeigt. Ein Magnetometer ist jedoch aufgrund von Störeinflüssen, etwa durch elektrische Felder, ebenfalls nur begrenzt genau.

Neben den drei Sensortypen Accelerometer, Gyroskop und Magnetometer finden wir noch einige weitere Sensoren in Mobilgeräten, etwa Näherungs- und Umgebungslichtsensoren, Fingerabdrucksensoren, Barometer (Messung von Luftdruck) und Thermometer. Mehrere Sensoren können auf einem Baustein integriert sein. Die Digitalisierung, Auswertung und Aufbereitung der Sensordaten übernimmt ein spezieller Prozessor, der **DSP** (Digital Signal Processor). Ein Beispiel ist der „Motion Coprocessor“ M9 von Apple.

Übung 1.9:

Was unterscheidet ein Gyroskop von einem Accelerometer?

1.3.8 Energieversorgung und Energiemanagement

Mobilgeräte stellen sehr hohe Anforderungen an die Energieversorgung. Eine Smartphone-Batterie muss klein und leicht sein. Gleichzeitig muss sie aber auch eine Menge Verbraucher (CPU, GPU, Display, Netzwerkmodule, Sensoren etc.) über einen möglichst langen Zeitraum stabil mit Energie beliefern können.

Aus diesem Grund sind heute in Mobilgeräten vorwiegend **Lithiumionen-Akkus** auf Basis von Lithium-Verbindungen zu finden, da diese im Vergleich zu anderen Akku-Techniken wie Nickel-Metall-Hydrid (NiMH) oder Nickel-Cadmium (NiCd) eine höhere **Energiedichte** besitzen.

Die Energiedichte beschreibt, wie viel **Energie** in einem Akku im Verhältnis zu seiner Masse oder seinem Volumen gespeichert ist, und wird in Wattstunden pro Kilogramm (Wh/kg) oder Wattstunden pro Volumen (Wh/m^3) angegeben. Lithiumionen-Akkus besitzen im Vergleich zu NiMH- oder NiCd-Akkus eine etwa doppelt bis dreimal so große Energiedichte.

Die Maßeinheit für Energie ist die **Wattstunde** (Watt mal Stunde, kurz Wh). Sie beschreibt vereinfacht gesagt die Fähigkeit eines Systems, Arbeit zu verrichten, bspw. einen Motor anzutreiben oder die CPU mit Strom zu versorgen. Das Formelsymbol für Energie ist E (von engl. *energy*).

Die **Leistung** bezeichnet die in einer Zeitspanne von einem Akku gelieferte bzw. von einem Verbraucher konsumierte Energie. Ihre Einheit ist das Watt (W), ihr Formelsymbol ist P (von engl. *power*). Der Zusammenhang zwischen Energie und Leistung ist somit über die Formel „Leistung gleich Energie pro Zeit“ bzw. $P = E/t$ gegeben. Mit der Energie von einer Wattstunde kann beispielsweise ein Motor mit 1 Watt Leistung eine Stunde lang angetrieben werden.

Aus diesen Zusammenhängen wird ersichtlich, dass die Komponenten von Mobilgeräten eine möglichst geringe Leistungsaufnahme aufweisen müssen um lange Akkulaufzeiten zu gewährleisten. Die Leistung lässt sich auch als Produkt aus **Stromstärke** I (Einheit Ampere, kurz A) und Spannung U (Einheit Volt, kurz V) darstellen:

$$P = I \cdot U$$

Die Komponenten von Mobilgeräten müssen daher mit möglichst niedrigen Spannungen betrieben werden, um bei gleichbleibender Stromstärke den Leistungsbedarf zu reduzieren.

Die **Nennkapazität** eines Akkus wird in Amperestunden (Einheit Ah) angegeben. Sie ist ein Maß für die Ladungsmenge, die in einem Akku gespeichert ist. Um von der Nennkapazität eines Akkus auf die darin gespeicherte Energie zu schließen, muss der Spannungswert bekannt sein, mit dem der Akku betrieben wird.

Die Werte für Nennkapazität und Betriebsspannung der Akkus finden sich in den Datenblättern oder sind auf das Gehäuse aufgedruckt. Übliche Mobilgeräte-Akkus besitzen eine Nennkapazität von etwa 1500–3000 mAh und arbeiten mit Betriebsspannungen zwischen 3 und 4 Volt.

Übung 1.10:

Welche Nennkapazität benötigt ein Akku, um einen Motor mit 50 mW (1 mW = 1 Milli-Watt = 1/1000 Watt) Leistung zwei Stunden lang bei 3 Volt Spannung anzutreiben?

Um die Akkulaufzeiten zu optimieren, besitzen die meisten Mobilgeräte einen speziellen Hardwarebaustein, den **Power Management IC**. Der Power Management IC dient dazu, den Energiebedarf einzelner Komponenten zu erkennen und gegebenenfalls zu drosseln oder eine Komponente kurzfristig mit Energie zu versorgen.

Der Power Management IC reduziert beispielsweise den Energiebedarf eines Prozessors, indem die Taktfrequenz f oder die Betriebsspannung U gesenkt wird, denn die Leistungsaufnahme eines Prozessors ist proportional zu seiner Taktfrequenz und proportional zum Quadrat der Spannung:

$$P \sim U^2 \cdot f$$

Verdoppelt sich also die Taktfrequenz, verdoppelt sich auch die Leistung. Wird die Spannung verdoppelt, wird viermal so viel Leistung verbraucht.

Um effektiv zu arbeiten, muss der Power Management IC schnell reagieren, da sich der Energiebedarf eines Mobilgeräts sehr schnell ändern kann. Die Power Management ICs enthalten häufig auch eine spezielle Elektronik, um ein Überladen oder eine Tiefenentladung des Akkus zu verhindern und somit seine Lebensdauer zu verlängern.

Übung 1.11:

Weshalb kann die Akkulaufzeit eines Mobilgeräts nicht eindeutig vorhergesagt werden?

Zusammenfassung

Mobile Endgeräte sind von handlicher Größe, besitzen ein geringes Gewicht und eine eigene Stromversorgung. Wenn wir heute von mobilen Endgeräten sprechen, meinen wir in erster Linie Smartphones und Tablets. Diese Gerätegattung wird vorwiegend über einen berührungsempfindlichen Bildschirm bedient, besitzt eine maßgeschneiderte Softwareplattform, unterstützt diverse Multimedia-Anwendungen sowie Funknetzwerkstandards und verfügt über eine Vielzahl von Sensoren.

Die Idee einer maßgeschneiderten Plattform für Mobilanwendungen entstand bereits in den Siebzigerjahren. Erste Geräte kamen in den Achtziger- und Neunzigerjahren auf den Markt, konnten sich jedoch aus verschiedenen Gründen nicht durchsetzen. Erst der Ausbau der digitalen Funknetze Ende der Neunzigerjahre, die Multitouch-Steuerung des ersten iPhone, diverse Entwicklungen im Hardwarebereich sowie das große Softwareangebot führten zu der rasanten Verbreitung von Smartphones und Tablets.

Mobilgeräte besitzen spezielle Hardwarekomponenten, die vor allem platz- und energiesparend ausgelegt sein müssen. Häufig sind mehrere Komponenten auf einem SOC integriert. Neben CPU, GPU und Haupt- und Festspeicher gehört zu der typischen Hard-

wareausstattung eines Mobilgeräts auch ein Multitouch-Display, Kommunikations- und Netzwerkmodule, Kameras, Sensorik sowie eine Energieversorgung und ein ausgeklügeltes Energiemanagement.

Aufgaben zur Selbstüberprüfung

1.1 Wodurch unterscheidet sich ein RISC-Prozessor von einem CISC-Prozessor und weshalb enthalten Mobilgeräte üblicherweise einen RISC-Prozessor?

1.2 Welche Werte liefern die Accelerometer der drei Achsen, wenn das Gerät mit einem Pitch-Winkel von 30° gehalten wird?

1.3 Das Gyroskop aus Abb. 1.5 hat eine Masse von 5 g und misst bei einer Schwingungsgeschwindigkeit von 1 m/s eine Coriolis-Kraft von 0,01 kg · m/s^2. Ermitteln Sie die zugehörige Winkelgeschwindigkeit.

Angenommen, Sie messen diese konstante Winkelgeschwindigkeit über einen Zeitraum von drei Sekunden. Um welchen Winkel wurde das Gyroskop in der gegebenen Zeit gedreht?

1.4 Wie können Sensoren zum Akkusparen beitragen?

2 Software-Plattformen für mobile Endgeräte

Am Ende dieses Kapitels kennen Sie die Systemarchitektur der beiden wichtigsten Plattformen für mobile Anwendungen, iOS und Android. Sie sind in der Lage, eine einfache Anwendung für diese Plattformen mit dem jeweiligen Software Development Kit zu erstellen.

Cross-Platform Frameworks versprechen eine plattformübergreifende Entwicklung mobiler Anwendungen. Sie kennen die verschiedenen Kategorien dieser Frameworks und können ihre Vor- und Nachteile benennen.

2.1 Verbreitung von Software-Plattformen für mobile Endgeräte

Die derzeit verbreitetsten Systeme für mobile Endgeräte sind **iOS** von Apple und **Android** von Google. Wie der Graph in Abb. 2.1 zeigt, waren Anfang 2017 etwa 85 % aller mobilen Endgeräte mit einem Android-Betriebssystem ausgestattet, und ungefähr 14,7 % mit iOS.

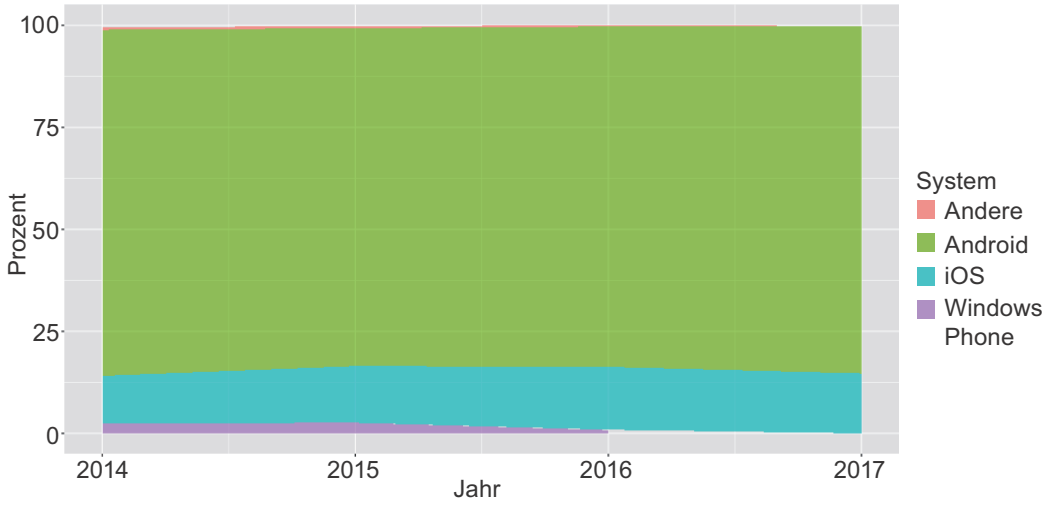


Abb. 2.1: Weltweite Marktverteilung mobiler Betriebssysteme in den Jahren 2014 bis 2017 (Quelle: http://www.idc.com/prodserv/smartphone-os-market-share.jsp)

Andere Plattformen wie **Blackberry OS** [7] oder **Windows Phone** spielen nur eine untergeordnete Rolle.

Im Folgenden werden wir uns mit den beiden wichtigsten Plattformen für mobile Anwendungen, iOS und Android, näher befassen. Dabei betrachten wir die Systemarchitektur dieser Plattformen und erstellen eine einfache „Hello World“-Anwendung. Als Grundlage der „Hello World“-Anwendung dient das **SDK** (Software Development Kit) der jeweiligen Plattform.

Im letzten Abschnitt dieses Kapitels widmen wir uns kurz der plattformübergreifenden Entwicklung mobiler Anwendung mithilfe von Cross-Platform Frameworks.

7. https://de.wikipedia.org/wiki/Blackberry_OS

2.2 iOS

Das dem iPhone zugrunde liegende Bedienkonzept ist bewusst sehr einfach gehalten. iOS-Geräte besitzen im Gegensatz zu Geräten früherer Generationen nur wenige Hardwarebuttons. Die Bedienung erfolgt fast ausschließlich über einen Multitouch-Bildschirm, der auch Mehrfingergesten erfasst.

Die Bildschirmoberfläche von iOS zeigt einen Home Screen, das **Springboard** (vgl. Abb. 2.2). Auf dem Springboard sind die Symbole („Icons“) der installierten Anwendungen in einem regelmäßigen Gitter angeordnet. Anwendungen werden über eine Fingerberührung auf ein Symbol gestartet. Mit einer Wischgeste kann auf weitere Bereiche des Home Screen umgeschaltet werden. Unten befindet sich das **Dock** mit vier weiteren Icons für die wichtigsten Anwendungen.

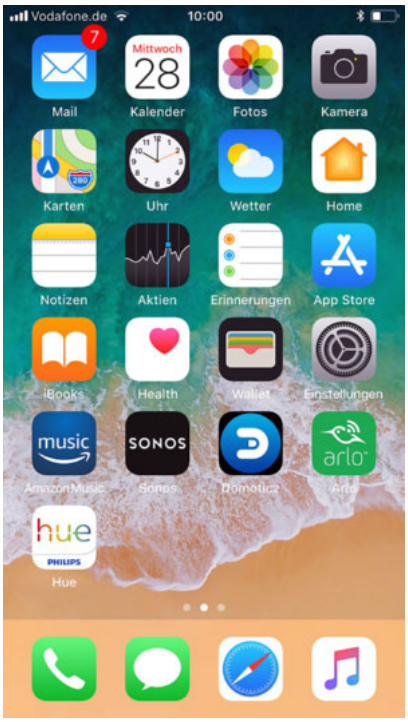


Abb. 2.2: iOS-Oberfläche mit Springboard und Dock (iOS 11)

Die erste Version des iPhone-Betriebssystems wurde Anfang 2007 zusammen mit dem iPhone vorgestellt und Mitte 2007 veröffentlicht. Im Frühjahr 2008 veröffentlichte Apple sodann ein SDK, das es auch Drittanbietern ermöglicht, native Anwendungen für das iPhone zu entwickeln und über den „App Store“ zu verbreiten. 2010 erfolgte mit Version 4.0 die Umbenennung des zu diesem Zeitpunkt „iPhone OS“ genannten Betriebssystems in iOS.

Seit dem Erscheinen des ersten iPhone wird iOS ständig aktualisiert. Jährlich erscheint eine neue Version mit zum Teil wesentlichen Überarbeitungen und Ergänzungen. Zusätzlich finden kleinere Versionssprünge statt, die vornehmlich Fehler beheben, die Sicherheit verbessern oder kleinere Ergänzugen bieten.

2.2.1 iOS-Systemarchitektur

Die Softwarearchitektur von iOS lässt sich in vier Schichten unterteilen, wobei eine Schicht einer höheren Stufe auf Funktionen der darunterliegenden Schichten zugreifen kann (vgl. Abb. 2.3). Der Abstraktionsgrad der Schichten nimmt nach oben hin zu.

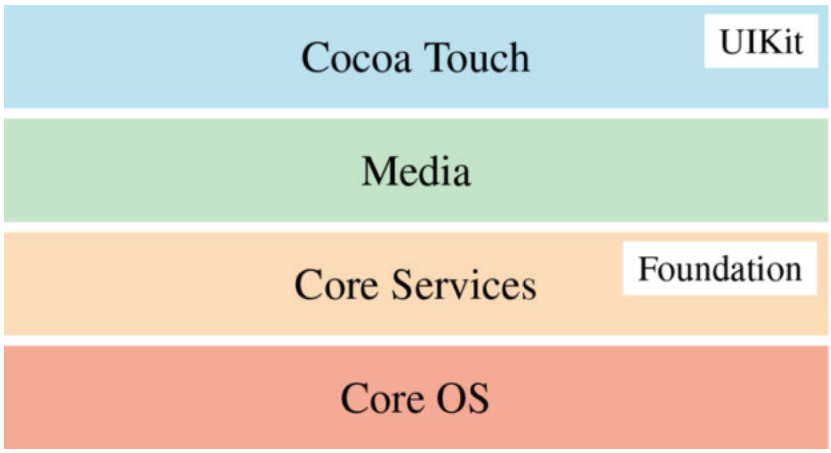


Abb. 2.3: iOS-System-Architektur

Die unterste Schicht **Core OS** enthält den Kern, die Hardware-Treiber und das low-level Unix-Interface des Betriebssystems. Der iOS-Kern verwaltet unter anderem den virtuellen Speicher, das Dateisystem, Netzwerk-Sockets, Threads (nebenläufige Programmfäden) und übernimmt das Energiemanagement. Der Zugriff auf den Betriebssystem-Kern und die Treiber ist geschützt und erfolgt über Funktionen des Betriebssystems.

Beim iOS-Kern handelt es sich um einen Abkömmling des an der Carnegie Mellon University in Pennsylvania entwickelten **Mach-Kernel**, der von der Firma NeXT und später von Apple weiterentwickelt und angepasst wurde. Das Unix-Interface basiert auf dem **BSD Unix**, einer Entwicklung der Universität von Kalifornien in Berkeley.

Neben dem Betriebssystem-Kern enthält die Core OS-Schicht eine Reihe von Software-Bibliotheken, die **Frameworks**. Zu den Frameworks der Core OS-Schicht zählen unter anderem:

Accelerate Framework:	digitale Signalverarbeitung, Lineare Algebra und Bildverarbeitung
Core Bluetooth Framework:	Verbindung zu Bluetooth-Geräten
Security Framework:	Zertifikate und Schlüsselverwaltung, Kryptografie
Local Authentication Framework:	Authentifizierung zum Schutz privater Daten

Die zweite Schicht **Core Services** enthält eine Reihe von System-Diensten, darunter Lokalisierungsdienste, Verwaltung von Cloud-Speichern, File-Sharing und Datenbankverwaltung. Die Funktionalitäten dieser Schicht lassen sich ebenfalls über Frameworks ansprechen. Die wichtigsten Frameworks der Core Services-Schicht sind **Foundation** und **Core Foundation**. Hier finden sich viele nützliche Datenstrukturen und Klassen, die die Programmierung einer Anwendung erleichtern. Daneben enthält die Core Services-Schicht viele weitere Frameworks, beispielsweise:

Core Data Framework:	Datenaufbereitung und Datenbankverwaltung (SQLite)
Core Location Framework:	Lokalisierung über GPS, Mobilfunk oder WLAN (WiFi)
Core Motion Framework:	Zugriff auf Accelerometer und Gyroskope
Local Telephony Framework:	Zugriff auf Telefoniefunktionen
EventKit Framework:	Zugriff auf Kalender und Termine
HealthKit Framework:	Fitness und Körperüberwachung
Social Framework:	Zugriff auf Twitter und Facebook
StoreKit Framework:	Zugriff auf App Store und „In-App Purchase“ (In-App Käufe)
WebKit Framework:	Darstellung von HTML-Seiten in einer iOS-Applikation

Die dritte Schicht **Media** fasst die Multimedia-Funktionalitäten zusammen und kann über eine Vielzahl von Frameworks angesprochen werden, etwa:

AVFoundation und AVKit Framework:	Videos und Musik abspielen, aufnehmen und verwalten
Core Audio:	Musik und Töne generieren, aufnehmen und mischen
Core Graphics Framework:	2D-Vektorgrafik
Core Image Framework:	Bild- und Videobearbeitung, Bildfilterung
GLKit, OpenGL ES und Metal Framework:	Low-level Schnittstellen für 3D-Grafik

In der obersten Schicht **Cocoa Touch** sind unter anderem die Funktionen zusammengefasst, die das Aussehen und das User Interface einer iOS-Anwendung definieren. Wichtigster Bestandteil von Cocoa Touch ist das **UIKit Framework**, das die Infrastruktur zur Gestaltung einer grafischen Benutzeroberfläche sowie für die Ereignisbehandlung bereitstellt. Daneben definiert die oberste Schicht ebenfalls weitere Frameworks, beispielsweise:

iAd Framework:	Banner-Werbung
MapKit Framework:	Zugriff auf den Kartendienst „Apple Maps“
Message UI:	SMS oder E-Mail versenden

Weitere Informationen zur iOS-Systemarchitektur und zu den Frameworks finden Sie in dem technischen Dokument „iOS Technology Overview“ (iOS, 2015).

Übung 2.1:

Mit welchem Framework kommt ein Anwender oder eine Anwenderin einer iOS-App direkt am meisten in Berührung? Mit welchen Frameworks haben Sie als Anwendungsprogrammiererin oder -programmierer am meisten zu tun?

Das Softwaresystem von iOS ist im Wesentlichen in den beiden Programmiersprachen C und Objective-C verfasst. Die in C geschriebenen Frameworks sind meist an dem Präfix „Core“ zu erkennen. **Objective-C** ist eine Erweiterung der Programmiersprache C um objektorientierte Funktionalitäten, wie Klassen und Vererbung, sowie weitere moderne Programmierkonzepte, etwa dynamische Typisierung und einen Garbage Collector. Ihre Wurzeln liegen neben C in der Sprache **Smalltalk**.

Objective-C ist die Sprache, die in der iOS- und macOS-Anwendungsentwicklung vorwiegend eingesetzt wird. Außerhalb von Apple ist diese Sprache weniger verbreitet.

Eine iOS-Anwendung ist in der Regel nach dem **MVC**-Entwurfsmuster (Model-View-Controller-Entwurfsmuster) strukturiert (vgl. Abb. 2.4). In diesem Modell sind die Benutzeroberfläche („View“), die eigentliche Programmlogik („Controller“) sowie das Datenhaltungsmodell („Model“) in eigene abgeschlossene Komponenten aufgeteilt, die miteinander kommunizieren.

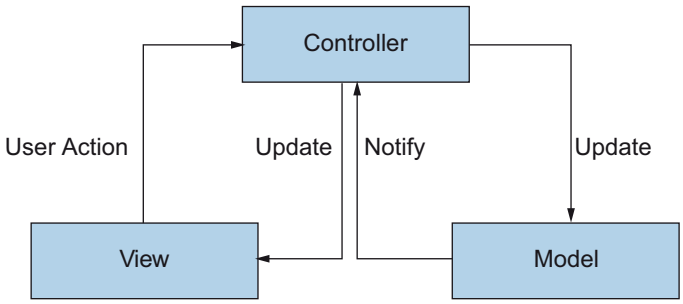


Abb. 2.4: Model-View-Controller-Entwurfsmuster

Die View-Komponenten reagieren auf eine Benutzerinteraktion („User Action“), beispielsweise ein Touch eines Buttons. Sie informieren den Controller über dieses Ereignis. Der Controller kann sodann die Model-Komponente informieren, beispielsweise um aktualisierte Daten in einer Datenbank zu speichern („Update“). Der Controller kann aber auch die View-Komponente informieren, falls die Benutzeroberfläche aktualisiert werden soll („Update“). Analog wird die Model-Komponente den Controller informieren, falls sich der Datenbestand geändert haben sollte („Notify“). Dies kann wiederum Änderungen der Darstellung notwendig machen, über die der Controller den View informiert.

Übung 2.2:

Welche Vorteile besitzt das Model-View-Controller-Entwurfsmuster?

2.2.2 Anwendungsentwicklung unter iOS

Die zur Programmierung einer iOS-Anwendung benötigten Programmbibliotheken (Frameworks) sind im iOS SDK enthalten, das Bestandteil der integrierten Entwicklungsumgebung **Xcode** ist. Xcode kann auf einem Rechner mit einem aktuellen macOS-Betriebssystem kostenlos über den App Store bezogen werden. Auf Rechnern mit anderen Betriebssystemen wie Linux oder Windows steht das iOS SDK nicht zur Verfügung.

Xcode ist eine leistungsfähige Entwicklungsumgebung, die neben C und Objective-C auch C++, Python und Ruby unterstützt. Seit 2015 ist zudem die von Apple neu entwickelte Sprache **Swift** (Apple, 2015 b) verfügbar. Swift ist eine leicht zu erlernende und leistungsfähige objektorientierte Sprache und soll die Programmierung von iOS-Anwendungen vereinfachen. Die Sprachen lassen sich in einer iOS-Anwendung nahezu beliebig mischen.

Hinweis:

> Sie können diesen Abschnitt nur dann praktisch nachvollziehen, wenn Sie über einen Rechner mit macOS verfügen.

Eine einfache „Hello World“-Anwendung erstellen Sie mit Xcode wie folgt: Im Startbildschirm wählen Sie *Create a new Xcode project*. Danach wählen Sie eine Vorlage („Template“) für eine iOS-Application aus, beispielsweise „Single View Application“. Geben Sie dann einen Namen für das Projekt an und wählen Sie eine der Sprachen Swift oder Objective-C aus.

In der Projektansicht links (dem „Navigator“) sehen Sie die Dateien des Projekts. Wichtig sind hier zunächst die Dateien „Main.storyboard“ und „ViewController.swift“ (Swift) bzw. „ViewController.h“ und „ViewController.m“ (Objective-C). Wenn Sie die Datei „Main.storyboard“ doppelt anklicken, öffnet sich eine weiße Oberfläche. Dies ist die Anwendungsoberfläche. Aus dem Bereich rechts unten (die „Object Library“) können Sie nun Steuerelemente wie Buttons, Labels, Slider, Switches usw. auf die Anwendungsoberfläche ziehen und darauf verteilen. Im Beispiel wurden ein Label und ein Button ausgewählt (vgl. Abb. 2.5).

Den Quellcode der Controller-Klasse in der Swift-Variante zeigt Code 2.1. Diese von Xcode generierte Klasse erbt von `UIViewController` und überschreibt zwei Ereignis-Methoden der Basisklasse, `viewDidLoad` und `didReceiveMemoryWarning`.

Um einen View auf der Anwendungsoberfläche mit dem Controller zu verknüpfen, ziehen Sie mit gedrückter mittlerer Maustaste eine Verbindung von dem View auf den Quellcode der Klasse `ViewController`. Unser Beispiel soll bei einem Klick auf den Button den Text im Label verändern. Zunächst benötigen wir also eine Referenz auf das Label. Verbinden Sie hierzu das Label mit dem ViewController (vgl. Abb. 2.6). Daraufhin öffnet sich ein kleines PopUp-Fenster. Wählen Sie *Connection: Outlet* und vergeben Sie einen Namen, damit Sie das Label im Code ansprechen können, z. B. „label“.

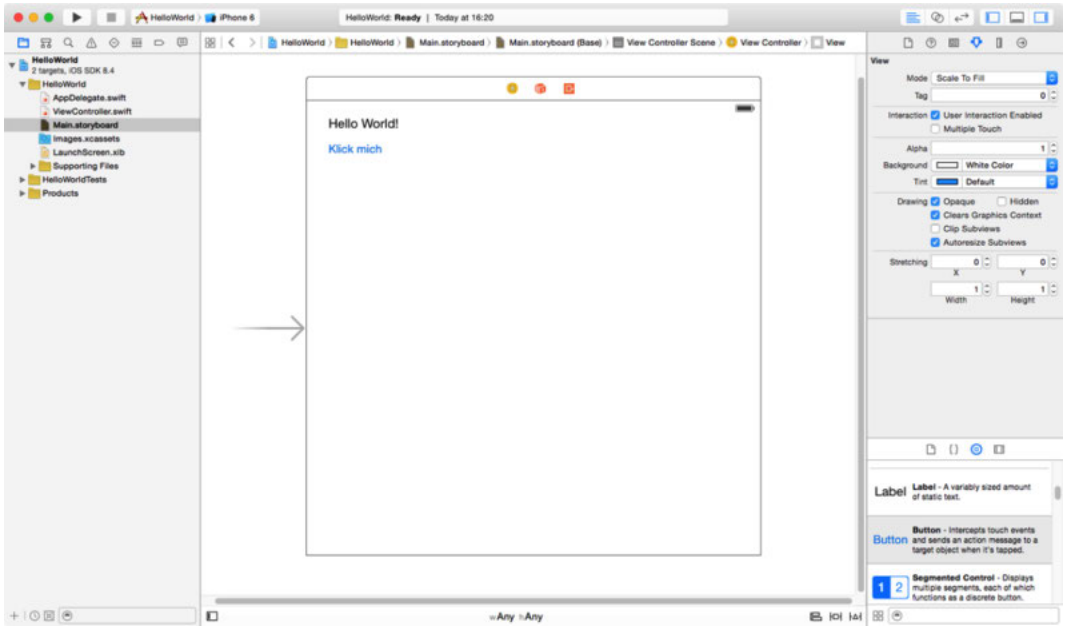


Abb. 2.5: Xcode: Single-View-Application-Projekt

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view,
        // typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

Code 2.1: Klasse ViewController

Ziehen Sie auch eine Verbindung von dem Button zum ViewController. Wählen Sie diesmal aber *Connection: Action*, da wir bei einem Klick auf den Button eine Aktion durchführen möchten. Vergeben Sie der Action einen Namen, beispielsweise „onButtonClick“.

Xcode fügt daraufhin eine Referenz auf das Label sowie die Funktion `onButtonClick` hinzu. Diese Funktion wird bei einem Touch auf den Button aufgerufen und hier können wir nun beispielsweise die `text`-Eigenschaft des Labels ändern. Code 2.2 zeigt die hierzu erforderlichen Ergänzungen in der Klasse `ViewController`.

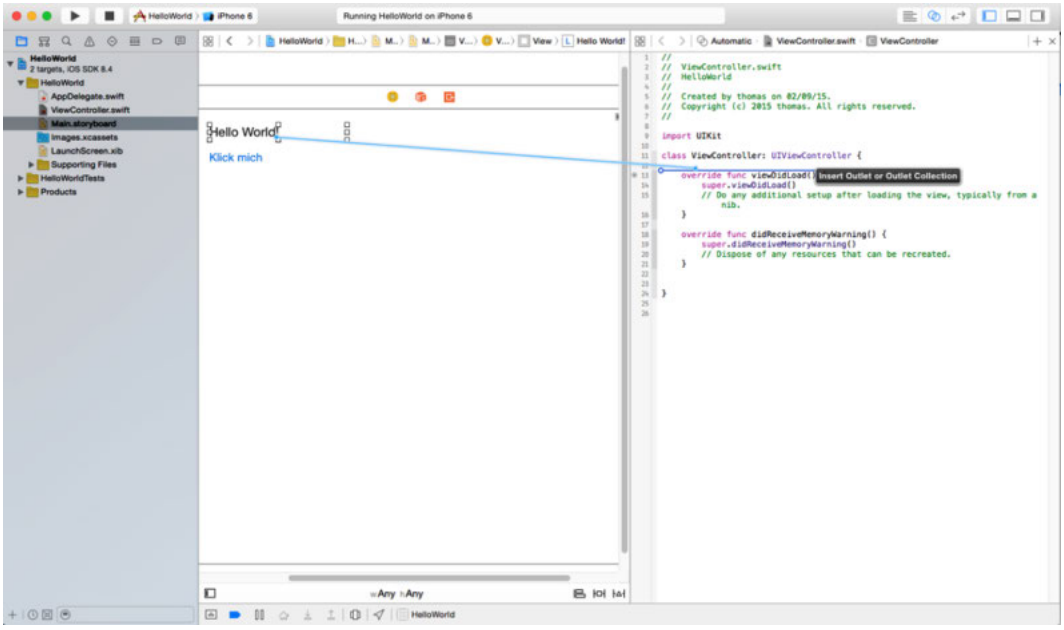


Abb. 2.6: Verknüpfung eines View mit dem Controller

```
class ViewController: UIViewController {

    @IBOutlet weak var label: UILabel!

    @IBAction func onButtonClick(sender: AnyObject) {
        label.text = "Button gedrückt"
    }

    // ...

}
```

Code 2.2: Ausschnitt der Klasse `ViewController` mit den Ergänzungen der „Hello World“-Anwendung

Nun können Sie die Anwendung testen. Apple setzt hier aber eine weitere Hürde: Um eigene Anwendungen auf einem iOS-Gerät zu installieren, benötigen Sie einen kostenpflichtigen Apple Developer Account. Die Kosten belaufen sich derzeit jährlich auf $ 100. Ohne diesen kostenpflichtigen Account können Sie die Anwendung nur in dem mit Xcode mitgelieferten Simulator laufen lassen, der jedoch für die ersten Gehversuche völlig ausreichend ist. Der Simulator startet automatisch bei einem Klick auf das Symbol mit dem schwarzen Pfeil in der oberen Symbolleiste von Xcode.

Nach einiger Zeit steht der Simulator bereit und zeigt die Anwendungsoberfläche (vgl. Abb. 2.7, links). Bei einem Klick auf den Button ändert sich der im Label dargestellte Text (vgl. Abb. 2.7, rechts).

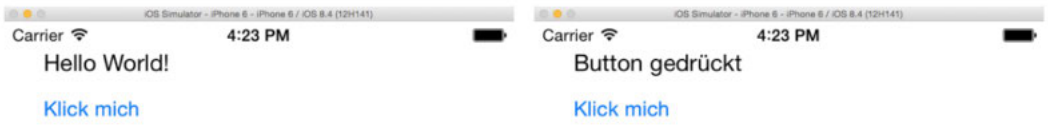


Abb. 2.7: „Hello World"-Anwendung im iPhone-6-Simulator
links: Startzustand
rechts: nach Klick auf den Button ändert sich der im Label dargestellte Text

Übung 2.3:

Welche Komponente ist in unserem Beispiel der Controller? Der View? Das Model?

2.3 Android

Die Oberfläche von Android zeigt ebenfalls einen Home Screen, den **Launcher** (vgl. Abb. 2.8). Neben Symbolen für den schnellen Zugriff auf eine Anwendung kann der Home Screen von Android auch **Widgets** aufnehmen. Das sind Programme, die ihre Informationen in einer kleinen Übersicht direkt auf dem Home Screen darstellen, wie zum Beispiel eine Uhr, ein Wetterdienst oder die gerade gespielte Musik. Der Home Screen kann wie unter iOS mehrere Bereiche umfassen, zwischen denen mit einer Wischgeste gewechselt wird. Wie unter iOS findet sich unten eine Leiste der wichtigsten Anwendungen. Welche Anwendungen in dieser Leiste angezeigt werden ist konfigurierbar.

Ältere Android-Geräte besaßen noch Hardwarebuttons zur Navigation in einer Anwendung. Neuere Android-Geräte besitzen außer einem An- und Ausknopf und einem Lautstärkeregler gar keine Hardwarebuttons mehr. Stattdessen wird am unteren Displayrand eine Symbolleiste, die **System Navigation**, mit drei Symbolen eingeblendet: Das Dreieck wechselt aus einem Anwendungsbildschirm zurück, beispielsweise auf den Home Screen. Der Kreis verschiebt eine laufende Anwendung in den Hintergrund. Über das Viereck kann eine pausierende Anwendung wieder in den Vordergrund geholt werden.

Android wurde Ende 2007 von der **Open Handset Alliance** unter der Federführung von Google angekündigt. Die Open Handset Alliance ist ein Zusammenschluss von derzeit über 80 Unternehmen, darunter Gerätehersteller wie HTC, LG und Samsung, sowie Mobilfunkanbieter wie Vodafone und T-Mobile. Erste Smartphones mit einem Android-Betriebssystem erschienen Ende 2008.

Android-Versionen mit wesentlichen Änderungen werden unter einem Codenamen zusammengefasst, der seit Android 1.5 nach einer Süßigkeit wie „Cupcake" oder „Donut" benannt wird.

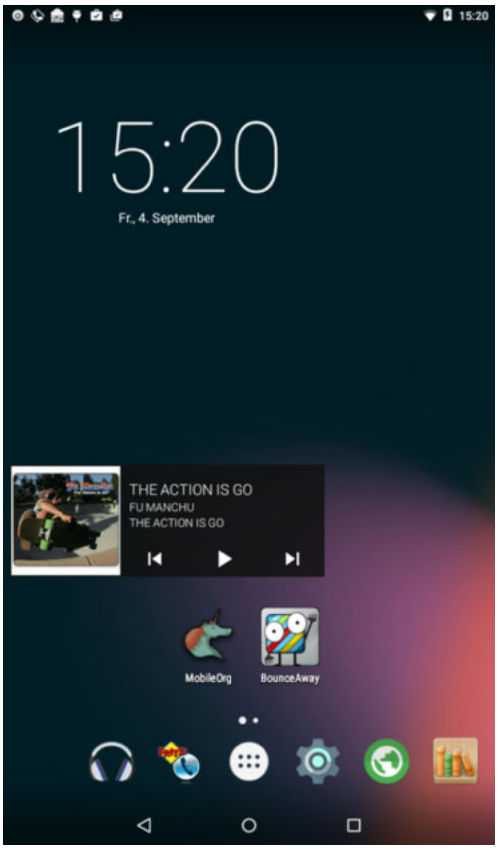


Abb. 2.8: Android Home Screen

Etwa jährlich erscheint eine aktualisierte Android-Version mit einem neuen Codenamen (vgl. Tab. 2.1).

Tab. 2.1: Android-Versionen

Version	Erscheinungsjahr	Codename
1.0-1.1	2008	Base
1.5	2009	Cupcake
1.6	2009	Donut
2.0-2.1	2009	Eclair
2.2	2010	Froyo
2.3	2010	Gingerbread
3.0-3.2	2011	Honeycomb
4.0	2011	Ice Cream Sandwich
4.1-4.3	2012	Jelly Bean
4.4	2013	KitKat
5.0-5.1	2014	Lollipop

Version	Erscheinungsjahr	Codename
6.0	2015	Marshmallow
7.0	2016	Nougat
8.0	2017	Oreo

2.3.1 Android-Systemarchitektur

Ähnlich wie iOS ist die Android System-Architektur in vier Schichten aufgeteilt (vgl. Abb. 2.9). Auf der untersten Ebene **Linux-Kernel** befindet sich der Betriebssystem-Kern mit den Hardware-Treibern. Android basiert auf einem Linux, das auf die speziellen Anforderungen eines Mobilgeräts, beispielsweise bezüglich des Energiemanagements, angepasst wurde.

Das **Hardware Abstraction Layer** (Hardware-Abstraktionsschicht) definiert ein Interface, über das Komponenten höherer Schichten die Hardware-Funktionen eines Android-Geräts nutzen können. Nur diese Schicht kann auf die Gerätetreiber der darunterliegenden Schicht direkt zugreifen. Das Hardware Abstraction Layer ermöglicht somit die Kompatibilität von Android mit einer Vielzahl von Geräten mit unterschiedlichen Hardware-Komponenten.

Auf der dritten Schicht befinden sich einige Bibliotheken („Libraries“), darunter SQLite (Datenbank), OpenGL ES (3D Grafik), WebKit (Webseitendarstellung) und SSL (Verschlüsselung). Weiterhin enthält diese Schicht die **Android Runtime** mit der **Dalvik Virtual Machine**, einer von Google speziell angepassten Laufzeitumgebung für Java-Anwendungen. Jede Android-Anwendung läuft in einem eigenen Prozess mit einer eigenen Instanz einer Dalvik Virtual Machine und kann nicht direkt auf die Hardware oder andere Prozesse zugreifen. Da ein Anwendungsprozess somit nicht die Stabilität oder die Sicherheit des Systems negativ beeinflussen kann, wird dieses Prinzip auch **Sandboxing** genannt. Mit Android 5.0 wurde die Dalvik Virtual Machine durch eine verbesserte Version ersetzt, die einfach als **ART** (Android Runtime) bezeichnet wird.

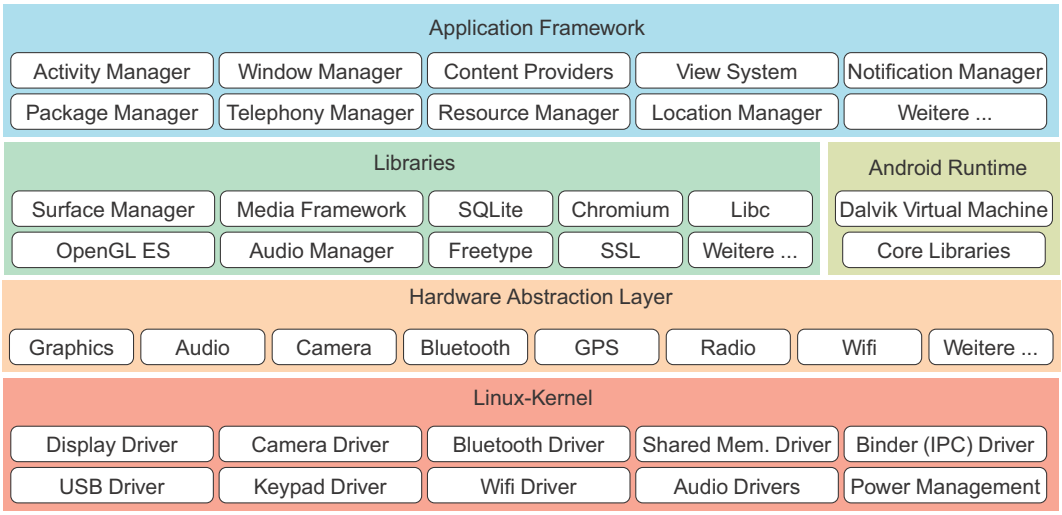


Abb. 2.9: Android-System-Architektur

Die **Core Libraries** enthalten Java-Bibliotheken zur Anwendungsentwicklung mit einer Teilmenge der Java Standard Edition („Java SE“) angereichert durch eine Vielzahl von Android-spezifischen Bibliotheken.

Die oberste Schicht **Application Framework** stellt die Infrastruktur für Android-Anwendung in Form spezieller Dienste zur Verfügung. Einige der wichtigsten Komponenten dieser Schicht sind:

Activity Manager:	steuert den Lebenszyklus einer Android-Anwendung
Content Provider:	erlaubt den Austausch von Daten zwischen Applikationen
View System:	definiert die Komponenten der grafischen Benutzeroberfläche
Telephony Manager:	ermöglicht den Zugriff auf Telefoniefunktion
Location Manager:	ermöglicht den Zugriff auf Lokalisierungsdienste

Bei einer **Activity** (Aktivität) handelt es sich um einen abgeschlossenen Teil einer Anwendung mit eigener Benutzeroberfläche. Eine Android-Anwendung besitzt in der Regel eine oder mehrere Activities. Eine Activity kann eine andere Activity starten, womit die neue Activity in den Vordergrund gerät. Der Start einer Activity kann entweder explizit durch Angabe eines Namens erfolgen oder auch implizit, indem nur die Anforderungen an die zu startende Activity definiert werden. Im zweiten Fall spricht man auch von einer **losen Kopplung** der Komponenten.

Die lose Kopplung von Activities funktioniert über Anwendungsgrenzen hinweg, sodass eine Anwendung auch die E-Mail-Anwendung, die Telefonieanwendung und andere Anwendungen starten kann, wenn sie die hierfür notwendigen Berechtigungen besitzt.

Übung 2.4:

Was müssen Gerätehersteller tun, damit Android auf ihren Geräten läuft?

2.3.2 Anwendungsentwicklung unter Android

Während der größte Teil der Android-Softwarearchitektur in den Programmiersprachen C und C++ geschrieben ist, erfolgt die Anwendungsentwicklung auf Android dagegen vorwiegend in der Programmiersprache Java. Die für die Android-Entwicklung benötigte Entwicklungsumgebung **Android Studio** mit dem Android-SDK kann von der Android-Developer-Webseite unter https://developer.android.com/sdk/index.html kostenlos für macOS, Linux und Windows heruntergeladen werden.

In Android Studio erstellen Sie über einen Projektassistent ein neues Projekt erstellt. Nach der Projekterstellung zeigt sich die Oberfläche von Android Studio wie in Abb. 2.10.

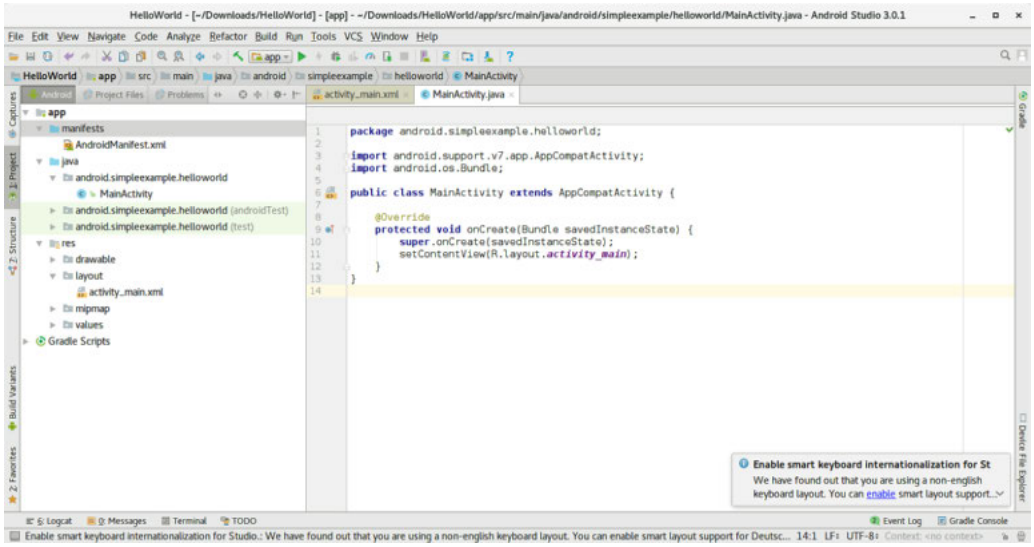


Abb. 2.10: Android Studio

Im Bereich *app* befindet sich das **App Manifest**. Dabei handelt es sich um eine Datei im XML-Format mit dem Namen „AndroidManifest.xml“. Das App Manifest ist die zentrale Konfigurationsdatei in einem Android-Projekt und legt unter anderem fest:

- den Namen des Projekts
- das Symbol (Icon), mit dem das Programm in der App-Übersicht angezeigt wird
- die Komponenten (Activities) der Anwendung
- das „Theme“ der Anwendung
- die Berechtigungen („permissions“) der Anwendung

Themes sorgen für eine einheitliche Gestaltung einer Android-Anwendung durch Festlegung von Farben, Textstilen, Abständen etc. Eine Android-Anwendung kann bestimmte Funktionen wie Lokalisierungsdienste, Telefonieanwendungen oder Internetverbindungen nur dann nutzen, wenn die hierzu erforderliche Berechtigung im App Manifest eingetragen ist.

Code 2.3 zeigt das vom Android Studio generierte App Manifest in einem einfachen „HelloWorld“-Projekt.

Im Wurzelelement `manifest` ist ein `application`-Element eingeschachtelt, das wiederum ein `activity`-Kindelement enthält. Das Element `intent-filter` (etwa: „Absichts-Filter“) spezifiziert, auf welche Anforderungen die Activity reagieren soll (vgl. „lose Kopplung“, Abschnitt 2.3.1). Die Action „MAIN“ kennzeichnet die Activity als Start-Activity, die Kategorie „LAUNCHER“ sorgt dafür, dass sie in die App-Übersicht aufgenommen wird.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest package="android.simpleexample.helloworld"
    xmlns:android="http://schemas.android.com/apk/res/android">

  <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
```

```
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
      android:supportsRtl="true"
      android:theme="@style/AppTheme">
      <activity android:name=".MainActivity">
        <intent-filter>
          <action android:name="android.intent.action.MAIN"/>

          <category
          android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
      </activity>
  </application>
</manifest>
```

Code 2.3: Android App Manifest (Datei „AndroidManifest.xml“)

Der Ordner „res“ enthält die Ressourcen eines Android-Projekts. Hier finden Sie in Unterordnern viele weitere XML-Dateien, die unter anderem das Layout der Benutzeroberfläche, Farben, Stile oder Texte (Strings) festlegen.

Code 2.4 zeigt das modifizierte Layout der Activity im „HelloWorld“-Projekt (Datei „res/layout/activity_main.xml“). Das Wurzelelement ist vom Typ „ConstraintLayout“ und enthält zwei Kindelemente: ein Textfeld („TextView“) und einen Button.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="android.simpleexample.helloworld.MainActivity">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        android:id = "@+id/textview_hello"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>

    <Button
        android:id="@+id/button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Button"
        android:onClick="onButtonClick"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@id/textview_hello"/>

</android.support.constraint.ConstraintLayout>
```

Code 2.4: Android-Layout-Definition (Datei „res/layout/activity_main.xml“)

Die verschiedenen Attribute beschreiben das Verhalten und das Aussehen eines Elements im Detail. Die Höhe des TextView und des Buttons wird in dem Beispiel durch ihren jeweiligen Inhalt bestimmt (`wrap_content`), während die Breite auf den Wert „0dp“ gesetzt ist. Das hat in einem ConstraintLayout den Effekt, dass die Elemente die gesamte Bildschirmbreite einnehmen. Das Layout selbst nimmt die gesamte Display-Oberfläche ein (`match_parent`). Der TextView besitzt eine ID, um ihn im Programmcode anzusprechen. Das `onClick`-Attribut des Buttons spezifiziert den Namen der EventHandling-Methode. Die Kindelemente des Layouts besitzen zudem ein `text`-Attribut.

Übung 2.5:

Wo finden Sie die Ressourcen in einem Android-Projekt?

Den Java-Quellcode für das einfache „HelloWorld“-Beispiel zeigt Code 2.5. Die Klasse `MainActivity` erbt von `AppCompatActivity` (eine abwärtskompatible Variante der Klasse `Activity`) und überschreibt die Lebenszyklusmethode `onCreate`. Diese Methode wird beim Start der Activity aufgerufen und lädt über den Aufruf der Methode `setContentView` das in der Datei „activity_main.xml“ spezifizierte Layout.

```
package android.simpleexample.helloworld;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
  }

  public void onButtonClick(View view)
  {
    TextView textview =
    (TextView)findViewById(R.id.textview_hello);
    textview.setText("Button geklickt");
  }
}
```

Code 2.5: Android-Activity-Quellcode (Datei „MainActivity.java“ im Ordner „app/java/android.simpleexample.helloworld“)

Der von Android Studio erzeugte Quellcode der `MainActivity` wurde um die Event-Handling-Methode `onButtonClick` erweitert. Diese Methode wird bei einer Berührung des Buttons aufgerufen und ändert den im TextView voreingestellten Text auf den String „Button geklickt“.

Nun kann das Projekt über den grünen Pfeil in der Werkzeugleiste von Android Studio auf einem Emulator oder angeschlossenen Realgerät ausgeführt werden.

Die fertige „Hello World“-Anwendung für Android zeigt Abb. 2.11. Bei einem Klick auf den Button wird der im TextView dargestellte Text entsprechend Code 2.5 abgeändert.

Abb. 2.11: „Hello World“-Anwendung unter Android

Seit 2017 ist es zudem möglich, Android-Anwendungen in der Programmiersprache **Kotlin** zu entwickeln (Google, 2017). Kotlin ist eine Entwicklung der Firma JetBrains und an JavaScript und andere moderne Sprachen angelehnt. Die Unterstützung von Kotlin kann bei Projekterstellung aktiviert werden (vgl. Abb. 2.12).

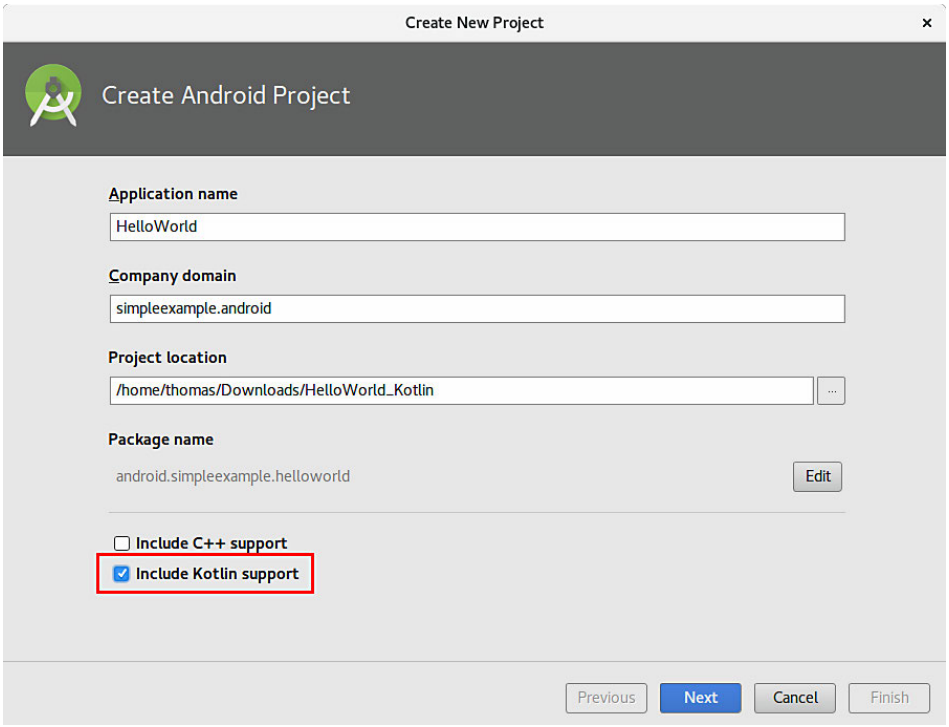


Abb. 2.12: Kotlin-Aktivierung in Android Studio

Das Beispiel aus Code 2.5 kann in Kotlin wie in Code 2.6 gezeigt umgesetzt werden.

```
package android.simpleexample.helloworld

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.TextView

class MainActivity : AppCompatActivity() {

    Override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    fun onButtonClick(view : View) {
        var textView =
        findViewById<TextView>(R.id.textview_hello)
        textView.setText("Button geklickt")
    }
}
```

Code 2.6: Das Beispiel aus Code 2.5 in Kotlin

Hinweis:

Neben Kotlin existieren noch weitere Programmiersprachen, die alternativ zu Java eingesetzt werden können. Hierzu zählen „Groovy“, „Scala“ und „Clojure“. Die Einbindung dieser Programmiersprachen ist allerdings etwas komplexer als bei Kotlin.

2.4 Plattformübergreifende Anwendungsentwicklung

Eine Anwendung, die Sie auf Basis des SDK einer Plattform entwickeln, ist auf diese spezielle Plattform zugeschnitten. Sie kann somit alle Funktionen der Plattform nutzen und kann für die Plattform optimiert werden. Eine Portierung der Anwendung auf eine andere Plattform ist jedoch aufwendig, da der Code angepasst und meist auch in eine andere Programmiersprache übersetzt werden muss.

Eine Alternative stellen die **Cross-Platform Frameworks** dar, die eine plattformübergreifende Anwendungsentwicklung versprechen. Der Code einer Anwendung muss mit einem Cross-Platform Framework nur einmal geschrieben werden und lässt sich auf verschiedenen Plattformen ausführen.

Die Cross-Platform Frameworks können in zwei Kategorien eingeteilt werden: Frameworks der ersten Kategorie generieren hybride Anwendungen, die der zweiten Kategorie native Anwendungen.

Eine **hybride Anwendung** basiert auf den Möglichkeiten zur Webseitendarstellung einer mobilen Plattform. Sie besteht aus einem mit dem SDK der jeweiligen Plattform erstellten nativen Teil. Dieser native Teil dient als **Container** für die eigentliche Anwendung, die mit plattformübergreifenden Webtechniken wie HTML5, CSS und JavaScript erstellt wird. Spezielle Schnittstellen ermöglichen den Zugriff auf Hardwarefunktionen wie Kamera oder Lokalisierungsdienste.

Beispiele für Frameworks dieser Kategorie sind PhoneGap[8], Mobile Angular UI[9] und Sencha Touch[10].

Übung 2.6:

Wie heißen die Browser Engines, die unter iOS und Android jeweils implementiert sind und in welchen Schichten sind sie jeweils zu finden?

Frameworks der zweiten Kategorie erzeugen aus einem Zwischencode ein vollständiges „natives" Projekt für die unterstützten Plattformen auf Basis der jeweiligen Plattform-SDK. Die Anwendung wird in einer Zwischensprache, beispielsweise JavaScript oder C#, verfasst und sodann von dem Framework in die native Sprache der Plattform übersetzt. Gegenüber hybriden Anwendungen hat dies den Vorteil, dass das Projekt mit dem nativen SDK weiter bearbeitet werden kann. Die Anwendung lässt sich sodann für eine Plattform verfeinern oder optimieren. Zudem entfällt die in einer hybriden Anwendung benötigte Interpretationsschicht zur Webseitendarstellung, wodurch die Hardware entlastet wird.

Beispiele für Frameworks der zweiten Kategorie sind Appcelerator[11] und Xamarin.[12]

Die Entscheidung für oder gegen ein Cross-Platform Framework hängt von verschiedenen Faktoren ab. Um nur eine Auswahl zu nennen:

- Unterstützung durch den Hersteller oder eine Community
- Kosten und Lizenzmodelle
- Einarbeitungsaufwand
- Leistungsfähigkeit
- Zielgruppe der Anwendung

Zusammenfassung

Die derzeit verbreitetsten Plattformen für mobile Anwendungen sind iOS und Android. Andere Systeme fristen derzeit allenfalls ein Nischendasein.

Die Systemarchitektur von iOS ist in vier Schichten aufgeteilt, wobei der Abstraktionsgrad nach oben hin zunimmt. Schichten höherer Stufen greifen in der Regel auf Funktionen unterer Schichten zu.

Die Anwendungsentwicklung für iOS-Geräte erfolgt vorwiegend in den Programmiersprachen Objective-C und Swift. Zusätzlich können auch weitere Sprachen wie C oder C++ eingesetzt werden. Das Software Development Kit (SDK) für iOS ist Bestandteil der integrierten Entwicklungsumgebung Xcode.

8. http://phonegap.com/
9. http://mobileangularui.com
10. https://www.sencha.com/
11. http://www.appcelerator.com/
12. https://xamarin.com

iOS-Anwendungen sind üblicherweise im MVC-Entwurfsmuster gestaltet, wobei eine Trennung zwischen dem Datenhaltungsmodell (Model), der Benutzeroberfläche (View) und der Programmlogik (Controller) stattfindet.

Die Android-Systemarchitektur ist ebenfalls in Schichten aufgeteilt. Die Android Runtime dient als Sandbox, in der jede Java-Anwendung von einer eigenen Dalvik Virtual Machine ausgeführt wird.

Für die Anwendungsentwicklung unter Android stellt Google die Entwicklungsumgebung Android Studio zur Verfügung. Android-Anwendungen bestehen aus einer Reihe lose gekoppelter Komponenten, die sich gegenseitig aufrufen und beeinflussen können. Die zentrale Komponente einer Android-Anwendung ist die Activity, ein abgeschlossener Teil einer Anwendung mit eigener Benutzeroberfläche.

Android-Projekte enthalten zudem eine Vielzahl von Konfigurationsdateien im XML-Format, die unter anderem das Layout einer Benutzeroberfläche, Farben, Stile und Texte beschreiben. Zentrale Konfigurationsdatei einer Android-An¬wen¬dung ist das Android Manifest. Hier werden neben diversen Projekteinstellungen auch die Berechtigungen der Anwendung spezifiziert.

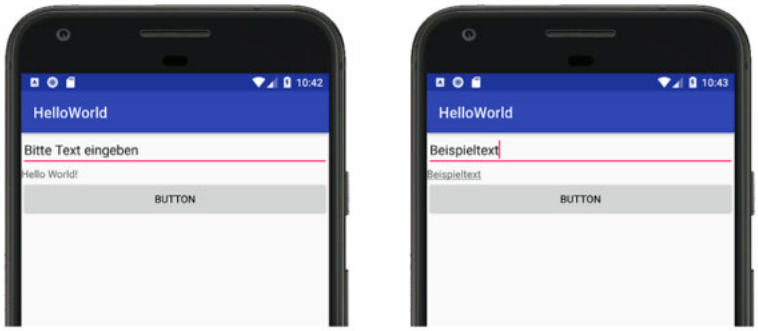
Cross-Platform Frameworks versprechen eine plattformübergreifende Anwendungsentwicklung für Mobilgeräte. Wir unterscheiden zwischen Frameworks, die eine hybride Anwendung generieren, und Frameworks, die ein natives Projekt der jeweiligen Plattform generieren.

Aufgaben zur Selbstüberprüfung

2.1 Geben Sie eine realistische Schätzung ab, welcher Anteil der Weltbevölkerung Anfang 2015 über ein Android- oder iOS-Gerät verfügte.

2.2 Beschreiben Sie, wie die lose Kopplung von Komponenten unter Android funktioniert.

2.3 Installieren Sie Android Studio und erweitern Sie das Beispielprogramm aus Abschnitt 2.3.2. um eine zusätzliche View-Komponente für die Eingabe von Text. Bei einem Klick auf den Button soll dieser Text in den TextView übernommen werden:



3 Datenübertragung in Funknetzwerken

Ein wesentliches Merkmal moderner Mobilgeräte sind die zahlreichen Möglichkeiten, mit verschiedenen Drahtlosnetzen zu kommunizieren.

Am Ende dieses Kapitels kennen Sie die physikalischen Grundlagen der drahtlosen Datenübertragung mittels elektromagnetischer Wellen. Sie sind mit dem OSI-Referenzmodell vertraut, das die Datenübertragung in digitalen Netzen anhand sieben Schichten auf verschiedenen Abstraktionsebenen beschreibt. Schließlich kennen Sie mit FDMA, ALOHA, TDMA und CDMA die vier wichtigsten Kanalzugriffsverfahren, die es mehreren Sendern ermöglichen, einen Übertragungskanal gemeinsam zu nutzen.

3.1 Physikalische Grundlagen

Mobilgeräte kommunizieren mit anderen Geräten in der Regel über drahtlose Netzwerke (Mobilfunknetze, GPS, WLAN etc., vgl. Abschnitt 1.3.5). Die Grundlage jeder drahtlosen Datenübertragung sind elektromagnetische Wellen, die von einem Sender über eine Antenne ausgestrahlt und von der Antenne eines Empfängers empfangen werden (vgl. Abb. 3.1).

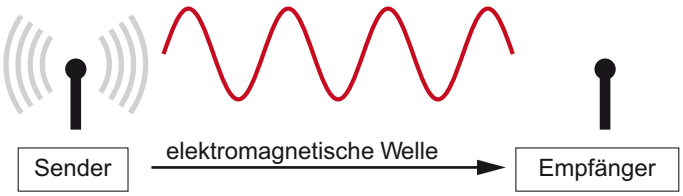


Abb. 3.1: Grundprinzip drahtloser Datenübertragung

Eine **elektromagnetische Welle** entsteht durch oszillierende (d.h. schwingende) Elektronen und kann mittels hochfrequenter Wechselströme erzeugt werden. Trifft die von einer Sender-Antenne abgestrahlte elektromagnetische Welle auf die Antenne eines Empfängers, bringt sie dort wiederum Elektronen zum Schwingen und erzeugt somit messbare Spannungsänderungen.

Eine Welle bezeichnet allgemein eine Veränderung bzw. Schwingung einer physikalischen Größe, die sich im Raum ausbreitet. Sie wird durch ihre Amplitude, Wellenlänge und Frequenz charakterisiert (vgl. Abb. 3.2). Die **Amplitude** einer Welle beschreibt die maximale Auslenkung ihrer Schwingung (bei elektromagnetischen Wellen bestimmt sie die Stärke des elektromagnetischen Felds).

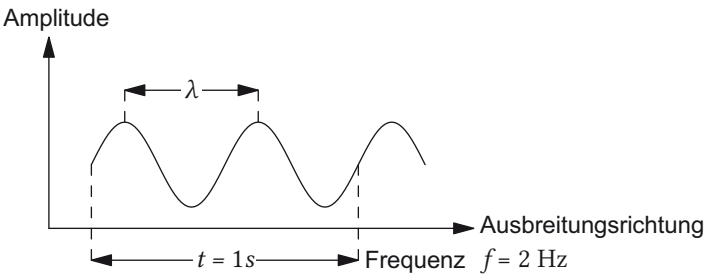


Abb. 3.2: Sinusförmige Welle mit Amplitude, Frequenz *f* und Wellenlänge *λ*

Die **Wellenlänge** λ einer periodischen (d.h. sich wiederholenden) Welle ist der Abstand zwischen zwei Punkten auf der Welle, die sich im gleichen Schwingungszustand befinden, beispielsweise der Abstand zwischen zwei Schwingungsspitzen oder zwei Schwingungstälern.

Die **Frequenz** f einer Welle ist der Kehrwert der Dauer einer Wellenperiode und wird nach dem deutschen Physiker Heinrich Hertz in der Einheit **Hertz** (Hz) angegeben, wobei 1 Hz = 1/s entspricht. Eine Welle mit der Frequenz 2 Hz durchläuft somit in der Dauer von einer Sekunde zwei Schwingungsspitzen.

Üblicherweise benötigt eine Welle ein Medium wie Wasser (Wasserwellen) oder Luft (Schallwellen), um sich auszubreiten. Elektromagnetische Wellen dagegen benötigen kein Medium und breiten sich auch im luftleeren Raum (Vakuum) aus. Feste Materie kann elektromagnetische Wellen abbremsen.

Das Produkt aus Wellenlänge λ und Frequenz f einer Welle bestimmt ihre **Ausbreitungsgeschwindigkeit** c:

$$c = \lambda \cdot f$$

Elektromagnetische Wellen breiten sich im Vakuum konstant mit Lichtgeschwindigkeit aus (d.h. c = 299 792 485 m/s, das entspricht etwa einer Milliarde Kilometer pro Stunde). Die Wellenlänge einer elektromagnetischen Welle bestimmt somit ihre Frequenz, und umgekehrt kann mit der Frequenz einer elektromagnetischen Welle ihre Wellenlänge berechnet werden.

Übung 3.1:

Wodurch unterscheiden sich elektromagnetische Wellen von mechanischen Wellen wie Schall- oder Wasserwellen?

Der gesamte Frequenzbereich elektromagnetischer Wellen wird als **elektromagnetisches Spektrum** bezeichnet (vgl. Abb. 3.3). Elektromagnetische Wellen besitzen abhängig von ihrer Wellenlänge bzw. Frequenz sehr unterschiedliche Eigenschaften. Am unteren Ende des Spektrums finden wir die sehr langwelligen Radiowellen mit Wellenlängen bis zu mehreren Kilometern. Radiowellen können sich über große Entfernungen ausbreiten, Gebäude durchdringen und sind für Lebewesen weitgehend unschädlich. Am oberen Ende des Spektrums befindet sich die hochfrequente und kurzwellige Gammastrahlung, die bei radioaktivem Zerfall entsteht. Dazwischen sind Mikrowellen, Infrarot-, Ultraviolett- (UV) und Röntgen-Strahlung. Sichtbares Licht besteht aus elektromagnetischen Wellen in dem schmalen Bereich zwischen Infrarot- und UV-Strahlung.

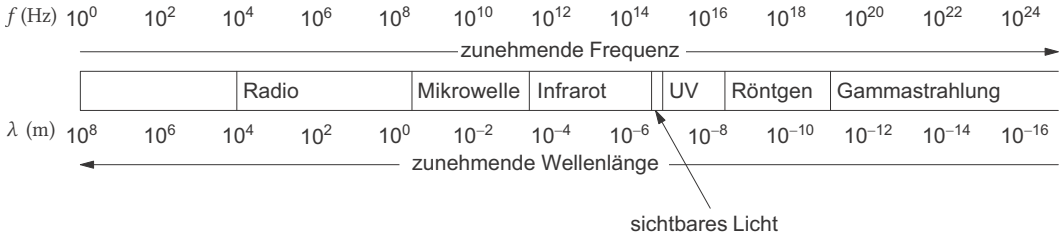


Abb. 3.3: Elektromagnetisches Spektrum

Drahtlose Netzwerkverbindungen nutzen die Frequenzen im Radio- und Mikrowellenbereich. Auch in diesen Frequenzbändern verhalten sich elektromagnetische Wellen stark unterschiedlich. Niederfrequente Radiowellen (bis etwa 10^6 Hz) folgen der Erdkrümmung und können in bis zu 1000 km Entfernung noch empfangen werden. Im Frequenzbereich zwischen 10^6 Hz bis etwa 10^8 Hz werden Radiowellen von der Ionosphäre reflektiert und können auf diese Weise große Distanzen überwinden. Radiowellen über 10^8 Hz und Mikrowellen breiten sich geradlinig aus und können so sehr gut auf einen Empfänger fokussiert werden. Wände und Gebäude durchdringen sie jedoch kaum.

Übung 3.2:

Welche Art der Netzwerkverbindung nutzt sichtbares Licht zur Datenübertragung?

Für die Datenübertragung wird das elektromagnetische Spektrum in **Frequenzbänder** aufgeteilt. Zum Beispiel nutzt der Mobilfunkstandard GSM-900 den Frequenzbereich von 890 bis 915 MHz (1 MHz = 10^6 Hz) für das Senden vom Mobilgerät zur Basisstation („Uplink“). Die Basisstationen senden im Frequenzbereich von 935-960 MHz („Downlink“). Diese Frequenzbänder werden häufig in weitere kleinere Bänder aufgeteilt, die jeweils den einzelnen Sendern oder Empfängern zugeteilt werden. Bei einem solchen Frequenzband spricht man auch von einem **Kanal**. GSM-900 teilt das genutzte Frequenzband beispielsweise in 124 Kanäle auf.

Die Differenz der oberen Frequenz und der unteren Frequenz eines Kanals ist die **Bandbreite**. Sie steht in einem engen Verhältnis zur theoretisch maximal möglichen Datenübertragungsrate: Je größer die Bandbreite, desto mehr Informationen können in einer Zeiteinheit übertragen werden.

Das **Trägersignal** des Kanals ist eine sinusförmige Schwingung mit einer Frequenz, die der Bandbreite des Kanals entspricht. Dieses Trägersignal wird zur Informationsübertragung genutzt, indem es durch eine **Modulation** geeignet verändert wird. Das einfachste Modulationsverfahren ist die Amplitudenmodulation. Dabei wird die Amplitudenstärke des Signals in diskrete Stufen unterteilt. Funknetzwerke nutzen üblicherweise effektivere Modulationsverfahren wie die Phasenmodulation oder die Frequenzmodulation.

Angenommen, mit jeder Halbschwingung des sinusförmigen Trägersignals wird 1 Bit übertragen. Die maximale Datenübertragungsrate entspricht in diesem Fall der doppelten Bandbreite *H*. Ein Signal mit 100 MHz Bandbreite könnte auf diese Weise maximal 200 Mbit/s (1 Mbit = 10^6) Bit) übertragen. Werden auf jeder Halbschwingung 2 Bit übertragen (etwa durch vier diskrete Amplitudenstufen bei einer Amplitudenmodulation), verdoppelt sich die Datenübertragungsrate. Wird für 1 Bit mehr als eine Halbschwingung benötigt, sinkt die Datenrate entsprechend.

Allgemein gilt bei einer Bandbreite H und einem Modulationsschema, das V verschiedene Zustände pro Halbschwingung codiert:

$$\text{maximale Datenübertragungsrate} = 2 \cdot H \log_2 V \text{ Bit/s}$$

Dieser fundamentale Zusammenhang zwischen Bandbreite und Datenübertragungsrate ist auch als **Nyquist-Theorem** (benannt nach dem schwedischen Physiker Harry Nyquist) bekannt. Die Formel gilt jedoch nur bei Idealbedingungen. In der Realität sind elektromagnetische Wellen Störeinflussen ausgesetzt, die sie abschwächen oder beim

Empfänger verändert auftreffen lassen. Bereits das atmosphärische Rauschen beeinträchtigt ein Signal, aber auch Signale anderer Sender auf den gleichen oder benachbarten Frequenzbändern wirken sich störend aus.

Die Signalstärke geteilt durch die Stärke des Störsignals bzw. Rauschens wird als Signal-Rausch-Verhältnis oder **Störabstand** (engl. signal-to-noise ratio) bezeichnet und in **Dezibel** (dB) angegeben. Der Störabstand wirkt sich ebenfalls auf die Datenübertragungsrate aus: Je größer der Störabstand, desto besser ist das Signal und es können mehr Daten pro Zeiteinheit übertragen werden (vgl. auch „Computer Networks“ (Tanenbaum; Wetherall, 2011)).

Übung 3.3:

Welche Bandbreite hat ein Downlink-Kanal im Mobilfunkstandard GSM-900? Wovon ist die Datenübertragungsrate ansonsten noch abhängig?

3.2 OSI-Referenzmodell

Die Datenübertragung in einem (drahtlosen) Netzwerk wird von einer Anwendung aufseiten des Senders initiiert. Die zu übermittelnden Daten werden sodann von der Sendeeinheit (bzw. dem Transceiver-Baustein, vgl. Abschnitt 1.3.5) in elektromagnetische Wellen umgesetzt, auf der Empfängerseite empfangen und dort schließlich von einer Anwendung verarbeitet.

Zwischen der Anwendung und dem Senden der Daten bzw. dem Empfang von Daten und deren Verarbeitung sind jedoch eine Reihe von weiteren Maßnahmen erforderlich, um eine reibungslose Kommunikation zu gewährleisten. Aufgrund von Störeinflussen (z.B. durch andere Sender oder atmosphärisches Rauschen) können bei der Datenübertragung Fehler oder Datenverluste auftreten. Dies muss erkannt und die Daten müssen sodann erneut übertragen werden. Teilen sich mehrere Sender einen Übertragungskanal, muss dies irgendwie koordiniert werden, damit sich die einzelnen Sender nicht gegenseitig stören (vgl. auch Abschnitt 3.3). Insbesondere drahtlose Kommunikation kann leicht mitgehört werden. Damit private Daten nicht in falsche Hände gelangen, müssen sie geeignet verschlüsselt werden.

Um von diesen und weiteren Details bei der Netzwerkübertragung zu abstrahieren, eignet sich ein Schichtenmodell. Jede Schicht kann auf die Funktionalität der darunterliegenden Schicht zugreifen und der Abstraktionsgrad nimmt nach oben hin zu.

Übung 3.4:

Wo ist Ihnen ein Schichtenmodell noch begegnet?

Ein Standardmodell für die Darstellung von Netzwerkkommunikation ist das **OSI-Referenzmodell** (Open Systems Interconnection). Dieses Modell wurde bereits in den Achtzigerjahren von der **ISO** (International Standards Organization) entworfen und besteht aus sieben Schichten (vgl. Abb. 3.4).

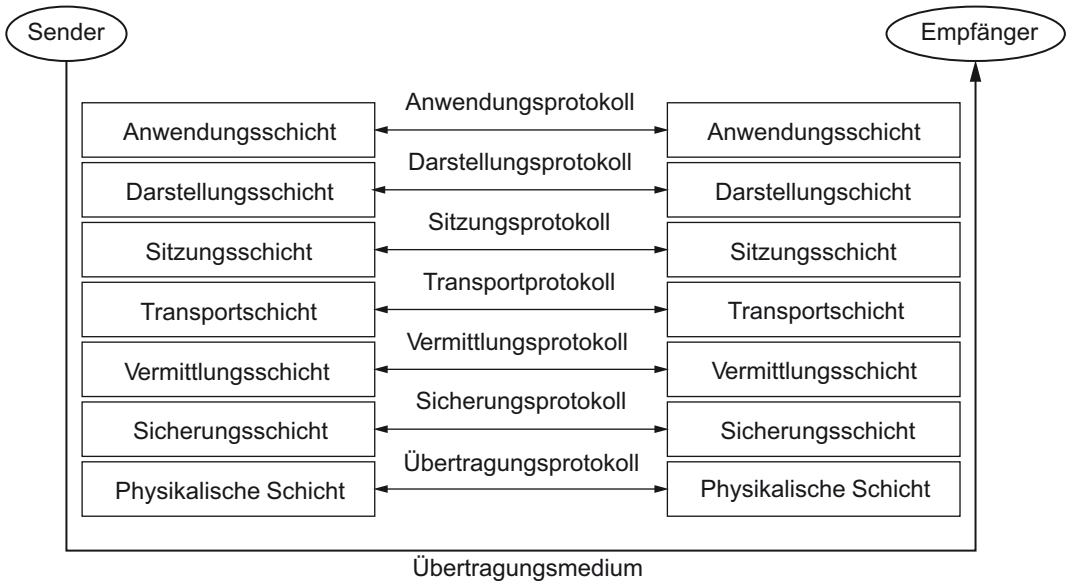


Abb. 3.4: Die sieben Schichten im OSI-Referenzmodell

Auf jeder der sieben Schichten existiert ein eigenes **Protokoll**, mit dem die Instanzen der jeweiligen Schicht auf Sender- und Empfängerseite kommunizieren. Die Details der Protokolle darunterliegender Schichten sind dabei für die Kommunikationspartner nicht von Interesse. Eine Anwendung muss sich somit beispielsweise nicht um die wiederholte Übertragung von verlorenen Datenpaketen kümmern, wenn diese Aufgabe bereits auf einem Protokoll einer darunterliegenden Schicht erledigt wird.

Die Aufgaben jeder Schicht im OSI-Referenzmodell können wie folgt zusammengefasst werden (vgl. auch „Computer Networks“ (Tanenbaum; Wetherall, 2011)):

Physikalische Schicht: auch „Bitübertragungsschicht“. Diese Schicht ist für die eigentliche Übertragung binärer Daten zuständig. Sie bestimmt, wie eine 0 oder eine 1 als Wellensignal codiert wird und sorgt für die Initiierung und Beendigung einer Verbindung.

Sicherungsschicht: Die Aufgabe dieser Schicht ist es, eine zuverlässige und fehlerfreie Übertragung zu gewährleisten. Hierzu werden größere Datenpakete in kleinere Pakete, die **Rahmen** oder **Frames**, aufgeteilt, eventuell mit einer Prüfsumme versehen und an die physikalische Schicht zur Übermittlung weitergeleitet. Der erfolgreiche Empfang eines Rahmens kann vom Empfänger durch einen speziellen **Acknowledgement Frame** bestätigt werden. Bei einem Fehlerfall (bspw. erkennbar durch einen Fehler in der Prüfsumme) kann der Empfänger die erneute Übertragung eines Rahmens anfordern. Die Sicherungsschicht muss auch den Fall eines kompletten Verlusts eines Rahmens erkennen und behandeln. Rahmen können beim Empfänger zudem in einer anderen Reihenfolge ankommen als sie gesendet wurden. Die Datenpakete müssen deshalb von der Sicherungsschicht des Empfängers in der korrekten Reihenfolge aus den einzelnen Frames zusammengesetzt werden.

Vermittlungsschicht: Bei **leitungsorientierten Diensten** (auch „verbindungsorientierte Dienste“) sorgt die Vermittlungsschicht für die Schaltung einer dauerhaften Verbindung, bspw. durch eine feste Zuordnung eines Übertragungskanals zwischen zwei Kommunikationspartnern. Die **paketorientierten Dienste** dagegen belegen einen Kanal nicht exklusiv, sondern nutzen ihn je nach Bedarf und Verfügbarkeit oder teilen die Kommunikation auf verschiedene Kanäle auf. Hier sorgt die Vermittlungsschicht für die Weiterleitung der Datenpakete an den Empfänger. Zu den Aufgaben der Vermittlungsschicht gehört zudem das **Routing**, d.h. die Wegesuche über verschiedene Netzwerkknoten zum Ziel.

Transportschicht: Die Transportschicht ist dafür zuständig, die von der Sitzungsschicht übergebenen Daten einer Anwendung beim Empfänger wieder an die gewünschte Anwendung auszuliefern. Auch die Transportschicht unterteilt einkommende Datenströme in kleinere Bestandteile, die **Segmente**. Die Transportschicht kann die Segmente verschiedener Anwendungen auf einem Übertragungskanal bündeln, indem sie nacheinander verschickt werden. Um den Durchsatz zu erhöhen, können aber auch Segmente einer Anwendung auf verschiedene Übertragungskanäle verteilt werden. Zu den Aufgaben der Transportschicht gehört zudem die **Flusskontrolle**. Sie dient zur Drosselung der Datenübertragungsrate, falls ein langsamer Empfänger die Daten nicht so schnell verarbeiten kann, wie sie geliefert werden.

Sitzungsschicht: Die Sitzungsschicht bietet den Anwendungen verschiedene weitergehende Dienste an, etwa um sich auf einem entfernten Rechner einzuloggen oder bei einem Dateitransfer nach einem Verbindungsabbruch die Übertragung an der entsprechenden Stelle wieder aufzunehmen.

Darstellungsschicht: Die Bedeutung einer Bitfolge hängt davon ab, wie das System sie interpretiert. Verschiedene Systeme repräsentieren Daten jedoch auf unterschiedliche Weise. Beispielsweise existieren verschiedene Standards zur Codierung von Zeichen wie UTF-8 oder ISO-8859. Um von diesen Unterschieden zu abstrahieren, werden die Daten von der Darstellungsschicht in ein einheitliches Format überführt. Kompression und Verschlüsselung finden ebenfalls häufig in dieser Schicht statt.

Anwendungsschicht: Anwendungen wie Webbrowser oder E-Mail-Programme greifen über die Anwendungsschicht auf die Funktionen darunterliegender Schichten zu. Die Aufgaben der Anwendungsschicht sind im OSI-Referenzmodell im Vergleich zu den anderen Schichten nicht sehr genau definiert.

Das OSI-Referenzmodell ist ein Modell, das in genau dieser Form in der Realität nicht existiert. Die meisten Implementierungen von Kommunikationsmodellen orientieren sich zwar am OSI-Modell, es können aber auch Schichten zu einer größeren Schicht zu-

sammengefasst sein oder gänzlich fehlen. Die Aufgaben der Schichten überschneiden sich in der Realität in einigen Fällen, oder mehrere Schichten übernehmen ähnliche Aufgaben (z.B. Fehlererkennung und Flusskontrolle).

Ein Beispiel für ein reales Modell ist **TCP/IP** (Transmission Control Protocol/Internet Protocol). Ein Vergleich der beiden Modelle zeigt Abb. 3.5. TCP/IP besteht aus vier Schichten, wobei die physikalische Schicht mit der Sicherungsschicht des OSI-Modells zu einer Schicht, der **Netzzugangsschicht**, zusammengefasst ist. Die Aufgaben der **Internetschicht** entsprechen im Wesentlichen der Vermittlungsschicht im OSI-Modell. Sitzungs- und Darstellungsschicht sind in TCP/IP konkret gar nicht vorhanden, ihre Funktionalität kann auch in der Anwendungsschicht nachgebildet sein.

Abb. 3.5: TCP/IP im Vergleich zum OSI-Referenzmodell

Dieser Abschnitt soll als kurzer Überblick über das OSI-Referenzmodell und die einzelnen Aufgaben der jeweiligen Schichten genügen. Nicht jeder Aspekt des OSI-Modells ist zudem im Kontext der Mobilgeräte von Bedeutung (z.B. Routing). Im nächsten Abschnitt werden wir uns mit Techniken befassen, die es ermöglichen, dass sich mehrere Kommunikationspartner einen Übertragungskanal teilen, um die verfügbare Bandbreite optimal auszunutzen. Kenntnisse über diese Kanalzugriffs- und Multiplexverfahren sind wesentlich zum Verständnis moderner Mobilfunknetze.

Übung 3.5:

Eine Netzwerkadresse teilt sich auf in die Angabe einer IP, gefolgt von einem Doppelpunkt und einer Portnummer, z.B. *192.168.1.123:8080*. Auf welchen Schichten des OSI-Modells bzw. des TCP/IP-Modells findet die Adressierung dabei jeweils statt?

3.3 Kanalzugriffs- und Multiplexverfahren

Allen Arten von (digitalen) Funknetzwerken steht nur eine begrenzte Bandbreite zur Verfügung, denn aus naheliegenden Gründen kann nicht der gesamte Frequenzbereich des elektromagnetischen Spektrums genutzt werden:

- Hochfrequente Strahlung überwindet nur vergleichsweise kurze Distanzen, wird von Gebäuden gehindert und kann sogar schädlich für Lebewesen sein (Gammastrahlung).
- Einige Frequenzbänder sind für wichtige Übertragungen wie Polizei- oder Militärfunk reserviert.
- Andere Netzwerke sowie Übertragungen von anderen Kommunikationspartnern im eigenen Netzwerk dürfen nicht gestört werden.

Damit sich Sender auf verschiedenen Netzen nicht stören, sind die nutzbaren Bandbreiten in den meisten Ländern streng reguliert. Somit besteht das Interesse der Netzbetreiber, die zugewiesene Bandbreite optimal auszunutzen, was bedeutet, so vielen Kommunikationspartnern wie möglich die möglichst größte Bandbreite bzw. Datenübertragungsrate zur Verfügung zu stellen.

Die in diesem Abschnitt besprochenen Techniken befassen sich also mit dem Problem, dass mehrere Sender einen Frequenzbereich teilen und die Frage „Wer erhält wann Zugriff auf das Übetragungsmedium?“ auf unterschiedliche Weise beantworten.

3.3.1 Frequency Division Multiplexing und Multiple Access

FDM (Frequency Division Multiplexing) ist ein bereits aus der Analogtechnik bekanntes Verfahren und unterteilt das verfügbare Frequenzband in kleinere, nicht überlappende Bereiche, die jeweils von einem Sender exklusiv genutzt werden können. Radio und (analoge) Telefonnetze beispielsweise basieren auf dieser Technik. Auch (digitale) Satellitenübertragungen und Mobilfunknetze nutzen FDM.

Das Grundprinzip von FDM zeigt Abb. 3.6: Das Signal der Sender wird bandbreitenbegrenzt. Das bedeutet, die Frequenzen unterhalb und oberhalb bestimmter Limits werden abgeschnitten. Analoge Telefonnetze zur Übertragung menschlicher Sprache begrenzen die Bandbreite beispielsweise auf den Bereich zwischen 300 und 3100 Hz.

Die Frequenz eines Senders wird sodann auf einen bestimmten Bereich angehoben, z. B. auf das Frequenzband von 60 kHz bis 64 kHz für den Sender 1 in Abb. 3.6. Wenn alle Sender auf unterschiedlichen Frequenzbändern senden, stören sie sich gegenseitig nicht. Es kann allerdings zu „Übersprechen“ kommen, etwa bei unzureichender Bandbreitenbegrenzung oder wenn die Frequenzbänder zu dicht beieinander liegen.

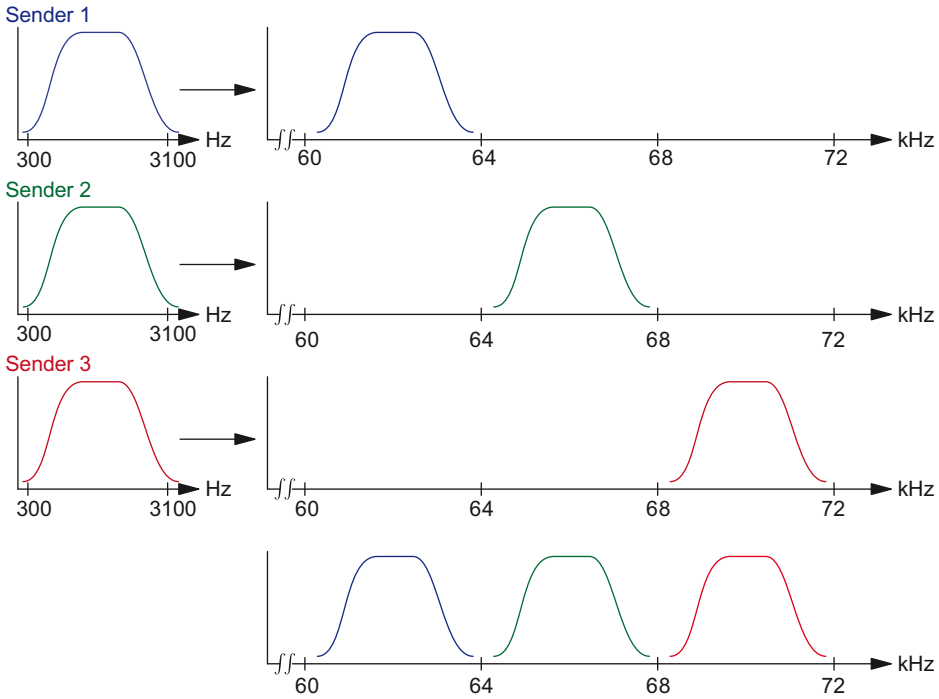


Abb. 3.6: Frequency Division Multiplexing (nach Tanenbaum und Wetherall, 2011)

FDMA (Frequency Division Multiple Access) ist das auf FDM basierende Kanalzugriffsverfahren (vgl. Abb. 3.7). Jedem Sender wird dabei ein (zumindest zeitweise) exklusiver Frequenzbereich zugeteilt, der unabhängig von anderen Sendern genutzt werden kann.

Die Anzahl der Sender, die mit FDMA gleichzeitig senden können, ist durch die Anzahl der Funkkanäle begrenzt. Jedem Sender steht dabei nur eine bestimmte Bandbreite zur Verfügung. Eine kurzfristig freiwerdende Bandbreite (etwa weil ein Sender zu einer bestimmten Zeit nicht sendet) kann von anderen Sendern nicht ohne Weiteres genutzt werden.

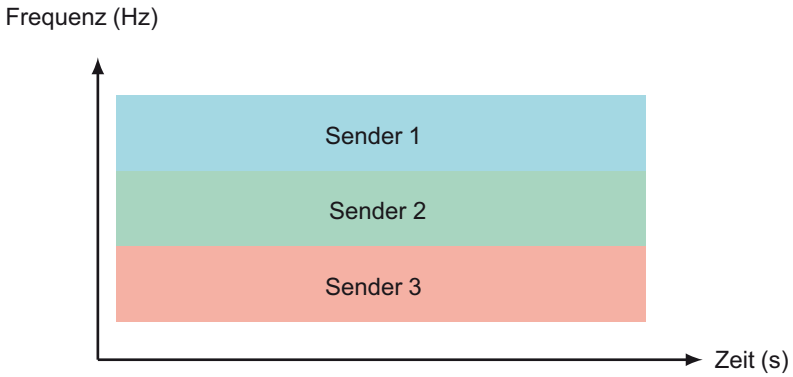


Abb. 3.7: Frequency Division Multiple Access

3.3.2 ALOHA

Im Gegensatz zu Radiosendern oder TV-Übertragungen über Satellitensignale zeichnen sich digitale Computernetze häufig dadurch aus, dass nicht zu jedem Zeitpunkt gesendet wird. Kurze Übertragungsspitzen (etwa wenn Sie eine Internetseite im Browser aufrufen) wechseln sich üblicherweise mit längeren Pausen ab. In digitalen Netzen ist es deshalb oft vorteilhafter, einem Sender über einen begrenzten Zeitraum die komplette Bandbreite zur Verfügung zu stellen. Trotzdem sollen so viele Sender wie möglich die Chance erhalten, ihre Daten zu versenden.

Eins der ersten speziell auf digitalen Datenverkehr zugeschnittenen Netzwerke ist **ALOHA**. Dieses Netzwerk entstand in den Siebzigerjahren an der Universität von Hawaii und diente dazu, die auf verschiedenen Inseln verteilten Rechner, die nur schwierig über Kabel verbunden werden konnten, über Funk zu vernetzen.

In der einfachsten Variante **pure ALOHA** versenden die Stationen ihre Nachrichten zu einer beliebigen Zeit. Falls zu diesem Zeitpunkt bereits ein anderer Sender aktiv ist oder zwei Sender sich gleichzeitig dazu entscheiden, eine Nachricht zu senden, kommt es zu einer **Kollision**: Die Signale verschiedener Sender stören sich gegenseitig. Die Nachrichten gehen verloren und müssen erneut übertragen werden.

Der **Durchsatz**, d. h. die tatsächlich übertragene Datenmenge im Verhältnis zur verfügbaren Bandbreite, hängt in einem ALOHA-Netzwerk davon ab, wie häufig Kollisionen stattfinden. Wenn es dauerhaft zu Kollisionen kommt, ist der Durchsatz in einem ALOHA-Netzwerk null.

Verschiedene Varianten von ALOHA-Netzwerken haben zum Ziel, die Anzahl der Kollisionen zu reduzieren. So kann eine Station beispielsweise den Übertragungskanal ständig abhören. Sollte dieser bereits von einer anderen Station belegt sein, wartet sie zunächst ab, bis der Kanal frei wird, bevor sie ihre Nachricht versendet. Dieses Prinzip nennt sich **CSMA** (Carrier Sense Multiple Access).

Wenn zwei Stationen nahezu gleichzeitig zu senden beginnen, kann es trotz CSMA zu Kollisionen kommen. In der Variante **CSMA/CD** (CSMA with Collision Detection) bricht die Station in diesem Fall ihre Übertragung ab, um den Übertragungskanal nicht für ein Datenpaket, das erneut übertragen werden muss, zu blockieren. Eine Kollision kann jedoch nur dann frühzeitig erkannt werden, wenn alle Stationen die Signale aller beteiligten Sender empfangen können. Diese Annahme trifft jedoch für drahtlose Netzwerke in der Regel nicht zu (vgl. auch Abschnitt 4.2.1). Eine Kollision kann in diesem Fall nur durch das Ausbleiben eines Acknowledgement Frame vom Empfänger erkannt werden.

Eine zweite Möglichkeit, den Durchsatz von ALOHA-Netzwerken zu steigern, nennt sich **slotted ALOHA**. In diesem synchronisierten Verfahren kann eine Station nicht zu einer beliebigen Zeit mit dem Senden beginnen, sondern nur zu bestimmten Zeiteinheiten, den **Time Slots**.

Slotted ALOHA verdoppelt in etwa den Durchsatz von pure ALOHA. Um dies einzusehen, müssen wir die **Vulnerabilitätszeit** eines Datenpakets bzw. Frame in pure ALOHA betrachten. Das ist die Zeit, in der kein anderer Sender senden darf, damit ein Datenpaket nicht verloren geht, es also „verwundbar“ gegen Kollisionen ist. Angenommen, alle Datenpakete besitzen die gleiche Größe. Davon können wir in ALOHA-Netzwerken in der Regel ausgehen, denn eine einheitliche Frame-Größe erhöht den Durchsatz. Weiter-

hin sei angenommen, ein vollständiges Datenpaket benötigt die Zeit (Δt) zur Übertragung und wird zum Zeitpunkt t gesendet. Damit es zu keiner Kollision mit anderen Paketen kommt, darf kein anderer Sender im Intervall von t - Δt) bis $t + \Delta t$) senden (vgl. Abb. 3.8). Die Vulnerabilitätszeit eines Datenpakets in pure ALOHA beträgt somit $2 \cdot \Delta t$. Teilen wir die Zeit wie in slotted ALOHA in diskrete Zeitintervalle der Länge Δt, reduziert sie sich auf Δt.

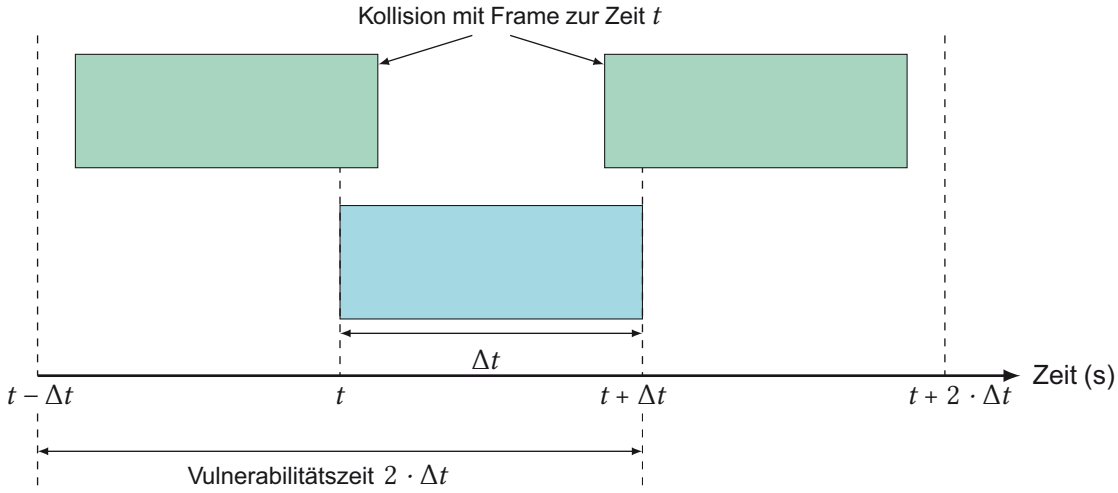


Abb. 3.8: Vulnerabilitätszeit eines Datenpakets in pure ALOHA

ALOHA besitzt gegenüber anderen Kanalzugriffsverfahren den Vorteil, dass keine explizite Koordinierung der Sendestationen erfolgen muss (abgesehen von einem Zeitsignal bei slotted ALOHA). Der Nachteil ist, dass der Durchsatz von der Anzahl der Kollisionen abhängt und im schlimmsten Fall gegen null gehen kann. Es kommt zu vielen Kollisionen, wenn viele Stationen gleichzeitig senden möchten. Aber genau dann wäre eigentlich eine möglichst hohe Datenübertragungsrate wünschenswert. Der Durchsatz von ALOHA verringert sich aufgrund der vielen Kollisionen somit genau dann, wenn eigentlich ein möglichst hoher Durchsatz notwendig ist.

Um diesem Überschwemmen des Kanals bei einem hohen Datenaufkommen entgegenzuwirken, werden in ALOHA-Netzwerken in der Praxis weitere Heuristiken zur Selbstregulierung eingesetzt. Der kabelgebundene Netzwerkstandard **Ethernet** nutzt zum Beispiel eine Kombination aus slotted ALOHA und CSMA/CD mit **Exponential Backoff**: Erkennt eine Station eine Kollision, wählt sie zunächst eine zufällige Zahl x aus dem Intervall $[0, 2^i - 1]$. Sie wartet sodann x Time Slots ab, bevor sie das Datenpaket erneut versendet. Sollte es dabei wieder zu einer Kollision kommen, wird die Zahl x beim nächsten Versuch aus dem doppelt so großen Intervall $[0, 2^{i+1}-1]$ gewählt. Auf diese Weise nehmen sich die Stationen bei einem hohen Datenaufkommen mit dem erneuten Versenden kollidierter Pakete stärker zurück, was den Gesamtdurchsatz des Netzwerks steigert.

Übung 3.6:

Weshalb erhöht eine einheitliche Paketlänge den Durchsatz von ALOHA?

3.3.3 Time Division Multiplexing und Multiple Access

TDM (Time Division Multiplexing) unterteilt die Zeit wie slotted ALOHA in ein festes Raster. Jeder Time Slot wird bei TDM jedoch exklusiv von einem Sender genutzt. Dabei steht dem Sender – im Unterschied zu FDM – das gesamte Frequenzband zur Verfügung. TDM eignet sich somit sehr gut zur digitalen Datenübertragung. Die Sender müssen sich aber koordinieren, damit sie nicht die Time Slots anderer Sender belegen.

TDMA (Time Division Multiple Access) ist das auf TDM basierende Kanalzugriffsverfahren (vgl. Abb. 3.9). Jedem Sender wird dabei ein eigener Time Slot mit der kompletten Bandbreite zugeteilt, der unabhängig von anderen Sendern genutzt werden kann.

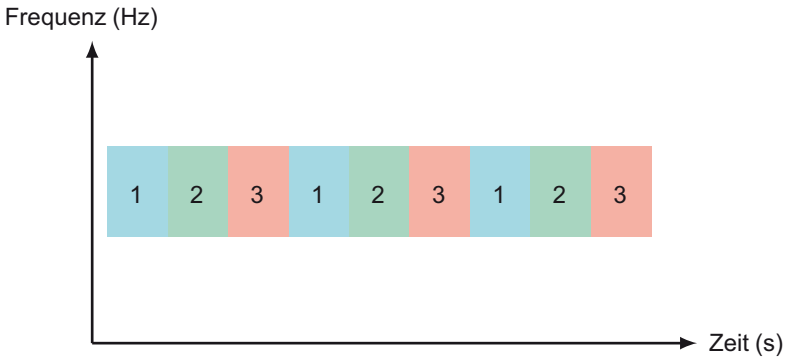


Abb. 3.9: Time Division Multiple Access

Die Zuweisung der Time Slots kann wie in Abb. 3.9 statisch erfolgen, wobei jede Station für die gesamte Dauer der Kommunikation einen bestimmten Slot belegt. Die Anzahl der Stationen, die gleichzeitig senden können, ist dabei durch die Anzahl der Time Slots begrenzt. Time Slots können aber auch dynamisch zugewiesen werden, um die verfügbare Bandbreite besser auszunutzen. Eine Station, die zu einem Zeitpunkt viele Daten übertragen muss, kann somit kurzfristig mehrere Time Slots erhalten. Stationen, die keine Daten senden, können Time Slots entzogen werden.

Übung 3.7:

Welchen finanziellen Vorteil hat TDM/TDMA für den Netzbetreiber im Vergleich zu (slotted) ALOHA?

3.3.4 Code Division Multiple Access

CDMA (Code Division Multiple Access) ist ein weiteres Kanalzugriffsverfahren, das in heutigen Mobilfunknetzen eingesetzt wird. Im Unterschied zu FDMA und TDMA steht einem Sender dabei die ganze Zeit die volle Bandbreite zur Verfügung – alle Sender senden gleichzeitig auf derselben Frequenz (vgl. Abb. 3.10). Anders als ALOHA geht CDMA jedoch nicht davon aus, dass sich die gleichzeitigen Signale verschiedener Sender gegenseitig stören. CDMA basiert stattdessen auf der Annahme, dass sich die Amplitudenstärken der Signale linear aufaddieren.

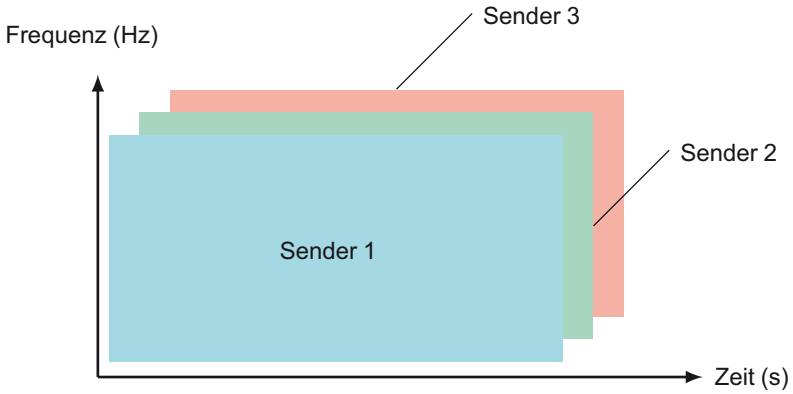


Abb. 3.10: Code Division Multiple Access

Um das Signal eines bestimmten Senders zu rekonstruieren, bedient sich CDMA der Codierungstheorie. Jeder Sender bekommt dazu eine eindeutige binäre **Chip-Sequenz** mit einer Länge m zugewiesen, die bestimmte mathematische Eigenschaften besitzt. Wenn eine Station S eine binäre 1 senden will, sendet sie stattdessen ihre Chip-Sequenz **S**. Bei einer 0 sendet sie das **Komplement** $\overline{\mathbf{S}}$ der Chip-Sequenz, wobei die Bits von **S** invertiert sind. Das Komplement der Chip-Sequenz $\mathbf{S} = (1,0,1,0)$ ist also $\overline{\mathbf{S}} = (0,1,0,1)$. Um die mathematischen Eigenschaften aufzuzeigen, werden Chip-Sequenzen üblicherweise **bipolar** notiert, d.h. eine binäre 0 wird als -1 dargestellt: aus der Sequenz (1,0,1,0) wird somit $(+1,-1,+1,-1)$.

Chip-Sequenzen sind paarweise **orthogonal**. Das heißt, das **innere Produkt** zweier Chip-Sequenzen **S** und **T**, mit $\mathbf{S} \neq \mathbf{T}$, ergibt null:

$$\mathbf{S} \cdot \mathbf{T} = \frac{1}{m} \sum_{i=1}^{m} S_i \cdot T_i = 0$$

Das innere Produkt einer Chip-Sequenz **S** mit sich selbst ergibt 1:

$$\mathbf{S} \cdot \mathbf{S} = \frac{1}{m} \sum_{i=1}^{m} S_i \cdot S_i = 1$$

Aus diesen beiden Eigenschaften folgt zudem: $\mathbf{S} \cdot \overline{\mathbf{T}} = 0$ und $\mathbf{S} \cdot \overline{\mathbf{S}} = -1$

Angenommen, drei Sender A, B und C senden gleichzeitig ihre Chip-Sequenz **A**, **B** und **C** oder das jeweilige Komplement. Beim Empfänger kommt die Summe **S** der drei Chip-Sequenzen an. Sei beispielsweise $\mathbf{S} = \mathbf{A} + \overline{\mathbf{B}} + \mathbf{C}$. Um eine Nachricht von Station **C** zu rekonstruieren, muss der Empfänger nun das innere Produkt des empfangenen Signals mit der Chip-Sequenz **C** bilden, denn

$$\mathbf{S} \cdot \mathbf{C} = (\mathbf{A} + \overline{\mathbf{B}} + \mathbf{C}) \cdot \mathbf{C} = \mathbf{A} \cdot \mathbf{C} + \overline{\mathbf{B}} \cdot \mathbf{C} + \mathbf{C} \cdot \mathbf{C} = 0 + 0 + 1 = 1$$

Ein Satz Chip-Sequenzen mit diesen Eigenschaften für die vier Sender A, B, C und D mit der Länge 4 wäre zum Beispiel:

$\mathbf{A} = (+1,+1,+1,+1)$
$\mathbf{B} = (+1,-1,+1,-1)$
$\mathbf{C} = (+1,+1,-1,-1)$
$\mathbf{D} = (+1,-1,-1,+1)$

Ortgogonale Chip-Sequenzen können über die sogenannte **Walsh-Funktion** ermittelt werden.

Die Anzahl gleichzeitiger Sender ist in CDMA theoretisch unbegrenzt. Viele Sender erfordern aber auch längere Chip-Sequenzen. Zudem treffen Signale weiter entfernter Sender schwächer beim Empfänger auf. Die Annahme, dass sich die Signale linear aufaddieren, kann somit in der Realität unter Umständen nicht aufrechterhalten werden. CDMA-Netzwerke wenden in diesem Fall eine Heuristik an, indem ein Sender seine Signalstärke reziprok zur empfangenen Signalstärke des Rückkanals vom Empfänger anpasst: Kommt ein schwaches Rücksignal an, erhöht der Sender seine Sendeleistung, andernfalls wird sie verringert.

Übung 3.8:

Welches Signal wird empfangen, wenn die vier Sender im obigen Beispiel die Bits 1 (Station A), 0 (Station B), 1 (Station C) und 0 (Station D) senden?

Zeigen Sie, dass die Signale der beiden Sender B und C mithilfe ihrer Chip-Sequenzen rekonstruiert werden können.

Zusammenfassung

Die Übertragung in Funknetzwerken basiert auf elektromagnetischen Wellen, die sich im luftleeren Raum mit Lichtgeschwindigkeit ausbreiten können. Das gesamte Frequenzspektrum elektromagnetischer Wellen reicht von niedrigfrequenten und langwelligen Radiowellen bis zur hochfrequenten Gammastrahlung. Kommunikationsnetze nutzen in erster Linie Radio- und Mikrowellen.

Das OSI-Referenzmodell beschreibt Netzwerke anhand von sieben Schichten mit jeweils einem zu den anderen Schichten abgegrenzten Aufgabengebiet. Auf jeder Schicht existiert ein eigenes Protokoll, mit dem die Instanzen auf einer gemeinsamen Ebene untereinander kommunizieren.

Um die verfügbare Bandbreite besser auszunutzen, verwenden Funknetzwerke verschiedene Kanalzugriffsverfahren. Diese Verfahren ermöglichen es mehreren Sendern, einen Kommunikationskanal gemeinsam zu nutzen. FDMA unterteilt hierzu das verfügbare Frequenzband und teilt jedem Sender einen eigenen Frequenzbereich zu. Bei ALOHA und TDMA nutzen die Sender das gesamte Frequenzband. Zu einem gegebenen Zeitpunkt kann aber nur ein Sender den Übertragungskanal belegen. Mit CDMA steht jedem Sender ebenfalls die gesamte Bandbreite zur Verfügung. Im Unterschied zu ALOHA geht CDMA jedoch von der Annahme aus, dass sich die Amplitudenstärken der Signale gleichzeitig sendender Stationen linear aufaddieren. Um das Signal eines bestimmten Senders zu rekonstruieren, bedient sich CDMA der Codierungstheorie.

Aufgaben zur Selbstüberprüfung

3.1 Welche Voraussetzungen müssen die Daten, die über ein ALOHA- oder TDMA-Netzwerk versendet werden, erfüllen? Gilt diese Voraussetzung auch für FDMA?

3.2 Wodurch unterscheidet sich ALOHA grundsätzlich von den anderen Kanalzugriffsverfahren?

3.3 Weshalb folgt aus den Eigenschaften $\mathbf{S} \cdot \mathbf{T} = 0$ und $\mathbf{S} \cdot \mathbf{S} = 1$ von Chip-Sequenzen, dass $\mathbf{S} \cdot \overline{\mathbf{T}} = 0$ und $\mathbf{S} \cdot \overline{\mathbf{S}} = -1$?

4 Mobilfunk- und Funknetzwerkstandards

Der Ausbau der digitalen Mobilfunknetze begann in den Neunzigerjahren mit dem Mobilfunkstandard GSM. Heute ermöglichen die Mobilfunkstandards der 4. Generation vielerorts die Nutzung einer Breitbandinternetverbindung von unterwegs.

Neben Mobilfunknetzen spielen eine Reihe weiterer drahtloser Netzwerke eine Rolle. In einem WLAN verbindet sich ein Mobilgerät mit einem lokalen Netzwerk oder weiteren Mobilgeräten. Über Bluetooth können Mobilgeräte untereinander oder mit anderen Geräten (meist Peripherie wie Headsets, Lautsprecherboxen, Tastaturen etc.) kommunizieren. NFC schließlich dient zum Datenaustausch über sehr kurze Distanzen, etwa zur bargeldlosen Bezahlung, für die Zugangskontrolle oder zum Abrufen von Informationen von einem Objekt ohne eigene Stromversorgung.

Am Ende dieses Kapitels kennen Sie den Aufbau von GSM im Detail und können die erzielbaren Datenraten dieses ersten digitalen Mobilfunkstandards technisch begründen. Weiterhin kennen Sie die Erweiterungen und Weiterentwicklungen GPRS, EDGE, UMTS, HSPA und LTE und können sie den verschiedenen Generationen der Mobilfunkstandards zuordnen. Die technischen Hintergründe und Einsatzgebiete von WLAN, Bluetooth und NFC sind Ihnen ebenfalls vertraut.

4.1 Mobilfunknetze

Die ersten Mobilfunknetze wie das C-Netz in Deutschland waren analog und auf die Übertragung von Sprache ausgelegt. Jedem Gesprächsteilnehmer wurde dabei für die Dauer eines Gesprächs ein Übertragungskanal fest zugewiesen (entsprechend Frequency Division Multiplexing, vgl. Abschnitt 3.3.1).

Das änderte sich mit dem Aufkommen des digitalen Mobilfunkstandards GSM (Global System for Mobile Communications) in den Neunzigerjahren. Heute übertragen Mobilfunknetze Daten in digitaler Form, d.h., die Schwingungen des Signals werden als Bits aufgefasst und nicht als Tonsignal (vgl. Abschnitt 3.1).

Gegenüber der Übertragung analoger Daten besitzen digitale Netze eine Reihe von Vorteilen. Zunächst kann Sprach- und Datenverkehr einheitlich behandelt werden. Sprache muss dafür jedoch zuerst digitalisiert und auf der Empfänger-Seite wieder decodiert werden. Dies übernehmen spezielle Hardwarebausteine, die Analog-Digital- und Digital-Analog-Wandler.

Durch neuere Kompressionsverfahren für digitalisierte Audiosignale wird die verfügbare Bandbreite besser ausgenutzt. Digitale Daten können einfacher und sicherer verschlüsselt werden als ein analoges Sprachsignal. Verschiedenen Quellen zufolge wächst das Übertragungsvolumen für den Datenverkehr in Mobilfunknetzen zudem deutlich stärker an als das Volumen für die Übertragung von Sprache (vgl. Abb. 4.1).

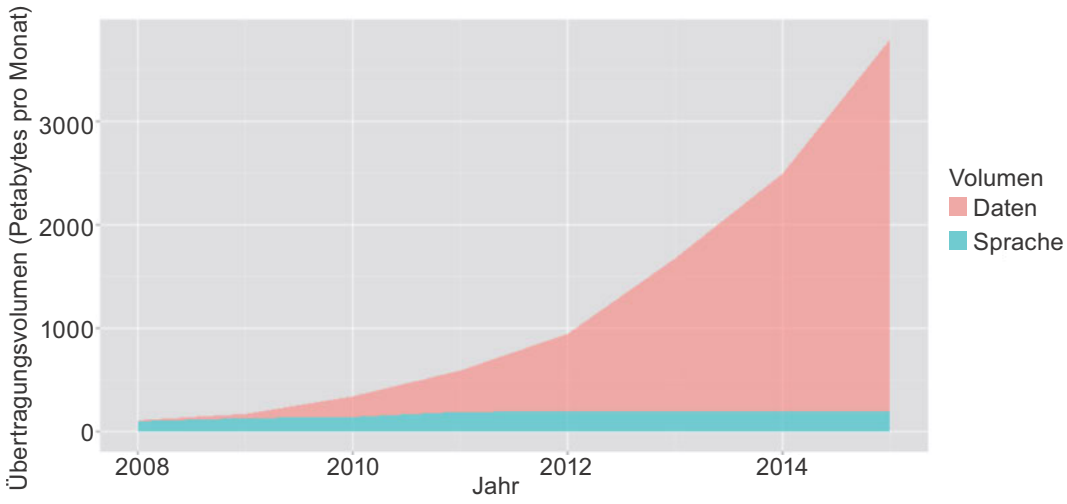


Abb. 4.1: Weltweites Übertragungsvolumen in Mobilfunknetzen (Quelle: Akmai/Ericsson https://www.stateoftheinternet.com)

Nicht zuletzt ermöglichen digitale Netze andere Kanalzugriffsverfahren wie ALOHA, TDMA oder CDMA und können somit die verfügbare Bandbreite dynamisch auf die verschiedenen Kommunikationsteilnehmer verteilen.

Heutige Mobilfunknetze sind in erster Linie **zelluläre Netzwerke**. Das Mobilfunknetz besteht aus einer Reihe stationärer, erdgebundener Basisstationen, die jeweils eine **Funkzelle** aufspannen (vgl. Abb. 4.2). Je nach Übertragungstechnik und Gegebenheiten deckt eine Funkzelle einen Bereich von einigen Hundert Metern (dicht bebautes Gebiet) bis wenigen Kilometern (offenes Gelände) ab. Die Funkzellen im Mobilfunkstandard GSM beispielsweise können einen Radius von bis zu 35 km aufweisen.

Gegenüber anderen Netzwerken, beispielsweise der Datenübertragung über Satelliten, haben sich zelluläre Netzwerke im Mobilfunkbereich vor allem aus Kostengründen durchgesetzt. Es werden zwar mitunter sehr viele Basisstationen benötigt, diese sind jedoch vergleichsweise günstig zu fertigen und zu warten und können häufig auf bereits vorhandenen Gebäuden installiert werden.

Zudem nutzen kleine Funkzellen die verfügbare Bandbreite sehr gut. Benachbarte Funkzellen müssen zwar unterschiedliche Kanäle verwenden, damit sie sich nicht gegenseitig stören. Eine Basisstation sendet jedoch gerade nur so stark, dass sie ihre eigene Funkzelle gut abdeckt. Bereits die übernächste Funkzelle kann somit dieselben Frequenzbänder wiederverwenden.

Die Mobilgeräte verbinden sich jeweils mit der Basisstation mit dem stärksten Signal. Bei Verlassen einer Funkzelle muss die Verbindung möglichst unterbrechungsfrei von der nächsten Basisstation übernommen werden. Dieser Vorgang wird als **Hand-Off** bezeichnet.

Übung 4.1:

Wie viele sich nicht überschneidende Frequenzbänder sind notwendig, damit sich die Funkzellen in Abb. 4.2 nicht gegenseitig stören?

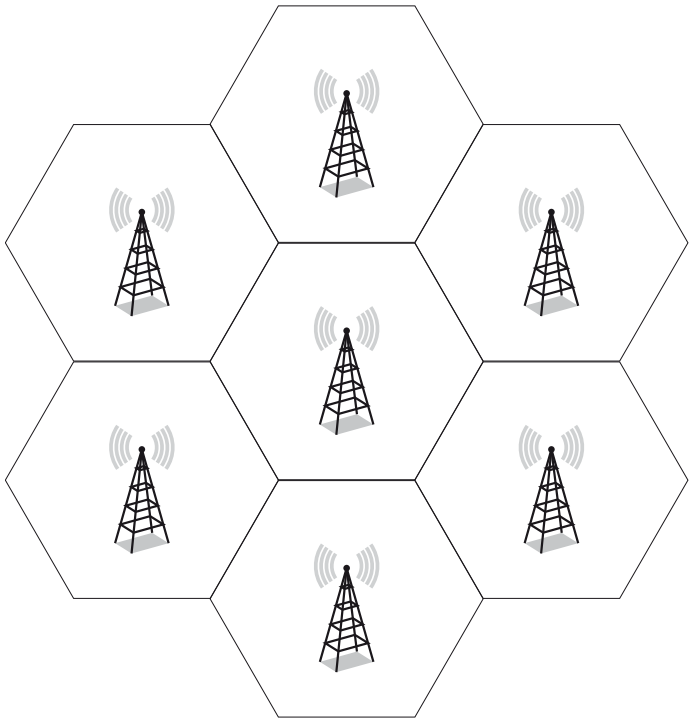


Abb. 4.2: Funkzellen mit Basisstationen in einem zellulären Netzwerk

Bis heute wurden bereits drei Generationen digitaler Mobilfunkstandards entworfen und sind weltweit erfolgreich in Betrieb. Der digitale Mobilfunkstandard GSM wird den Mobilfunknetzen der 2. Generation (kurz 2G) zugeordnet. Diese Generation löste die analogen Vorgänger der ersten Generation wie das C-Netz ab. Die heute schnellsten Mobilfunknetzwerke der dritten digitalen Generation bzw. der vierten Mobilfunkgeneration (4G) wie LTE Advanced erzielen Datenraten, die GSM um bis zu fünf Größenordnungen übertreffen.

Die wichtigsten Mobilfunkstandards können den vier Generationen wie folgt zugeordnet werden (vgl. auch Tab. 1.2):

1G: analoges zelluläres Mobilfunknetz für Sprachübertragung; Beispiel: C-Netz (1984–2000)

2G: digitales zelluläres Mobilfunknetz für Sprach- und Datenübertragung (vorwiegend SMS); Beispiel: GSM (ab 1992, 9,6 kbit/s), Erweiterungen GPRS (53,6 kbit/s), EDGE (236,8 kbit/s)

3G: digitales Mobilfunknetz der zweiten Generation; Beispiel: UMTS (ab 2000, etwa 384 kbit/s), Erweiterung HSPA (bis zu 42 Mbit/s)

4G: digitales Mobilfunknetz der dritten Generation; Beispiel: LTE (ab 2009, bis zu 300 Mbit/s), LTE-Advanced (1000 Mbit/s)

5G: digitales Mobilfunknetz der vierten Generation; ab 2020, ca. 3500 Mbit/s

Die Datenraten der Mobilfunknetze werden häufig durch eine schrittweise Verbesserung vorhandener Techniken gesteigert. Es existieren somit innerhalb einer Mobilfunkgeneration meist diverse Erweiterungen.

Die folgenden Unterabschnitte befassen sich mit den Mobilfunknetzen ab der ersten digitalen Generation (2G) und ihren wichtigsten Erweiterungen. Neben einer Begriffserklärung werden wir auch die erreichbaren Datenraten der Mobilfunkstandards anhand der in Kapitel 3 behandelten Grundlagen der Netzwerkübertragung begründen.

4.1.1 Mobilfunknetze der 2. Generation: GSM, GPRS und EDGE

Der digitale Mobilfunkstandard GSM zählt zur 2. Generation von Mobilfunknetzen und ist in Deutschland seit 1992 als Grundlage des D- und E-Netzes in Betrieb. GSM nutzt die Frequenzbereiche um 900 MHz (GSM-900) und 1800 MHz (GSM-1800, auch als DCS 1800 bezeichnet).

Für das Senden von den Mobilgeräten zur Basisstation (Uplink) sowie von der Basisstation zu den Mobilgeräten (Downlink) steht GSM ein jeweils 25 MHz breites Frequenzband zur Verfügung. Dieses Band wird weiter in 200 kHz breite Kanäle unterteilt. Einem Sender steht jedoch nicht die gesamte Bandbreite eines solchen Kanals zur Verfügung, sondern nur jeweils einer von acht Time Slots. GSM kombiniert somit Frequency Division Multiplexing mit Time Division Multiplexing auf jedem Kanal.

Das Schema der GSM-Kommunikation verdeutlicht Abb. 4.3. Angenommen, ein Mobilgerät erhält den zweiten Time Slot auf dem Uplink-Kanal 2. Es belegt sodann für die Dauer der Kommunikation diesen Slot. Die Basisstation nutzt einen eigenen Time Slot auf einem der Downlink-Kanäle, um Nachrichten an das Mobilgerät zu senden, im Beispiel der fünfte Time Slot im ersten Rückkanal. Die markierten Time Slots in Abb. 4.3 verdeutlichen den Ablauf einer Kommunikation zwischen Mobilgerät und Basisstation (nach Tanenbaum; Wetherall).

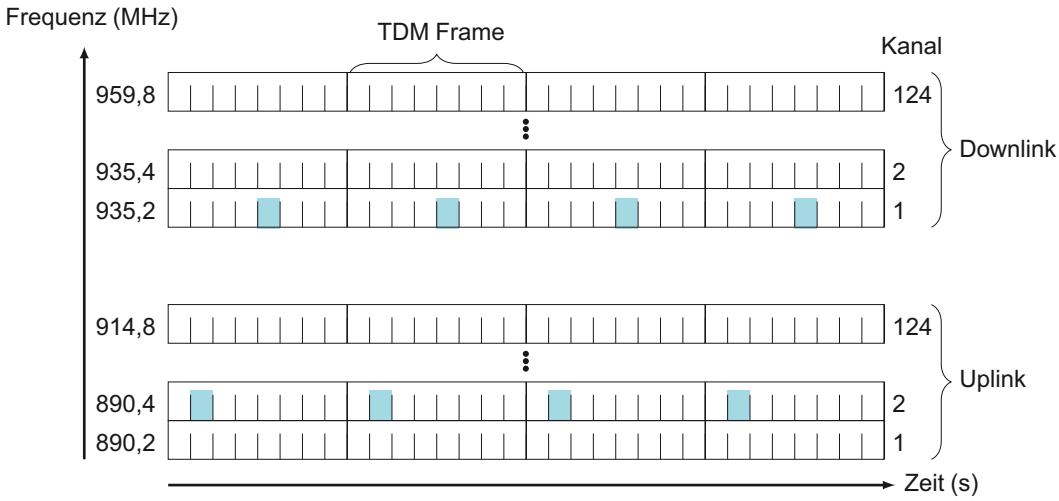


Abb. 4.3: GSM Frequenzbänder und Time Slots (nach Tanenbaum und Wetherall, 2011)

Die Wellensignale in GSM übertragen auf jeder Halbschwingung ein Bit. Bei einer Bandbreite von 200 kHz können somit nach Nyquist auf jedem Kanal maximal 2 · 200 kbit/s = 400 kbit/s übertragen werden (vgl. Abschnitt 3.1). Aufgrund des Störabstands werden in der Praxis allerdings nur etwa 270 kbit/s realisiert. Jedem Mobilgerät steht zudem nur einer von acht Time Slots auf einem der Kanäle zur Verfügung. Somit beträgt die **Bruttodatenrate**, d.h. die maximal mögliche Datenrate, für ein Gerät 270/8 kbit/s = 33,75 kbit/s.

Aber auch von diesen 33,75 kbit/s bleibt letztendlich nur eine **Nettodatenrate** von etwa 9,6 kbit/s für die eigentliche Datenübertragung übrig. Um dies einzusehen, schauen wir uns zunächst die Struktur eines Frame acht aufeinanderfolgender Time Slots an (vgl. Abb. 4.4). Ein solcher **TDM Frame** ist insgesamt 1250 Bit lang und enthält die acht **Data Frames** mit den in den jeweiligen Time Slots gesendeten Daten. Zwischen den Data Frames ist jeweils noch eine 8,25 Bit lange **Guard Time** eingefügt, um für ein wenig Toleranz beim Timing zu sorgen.

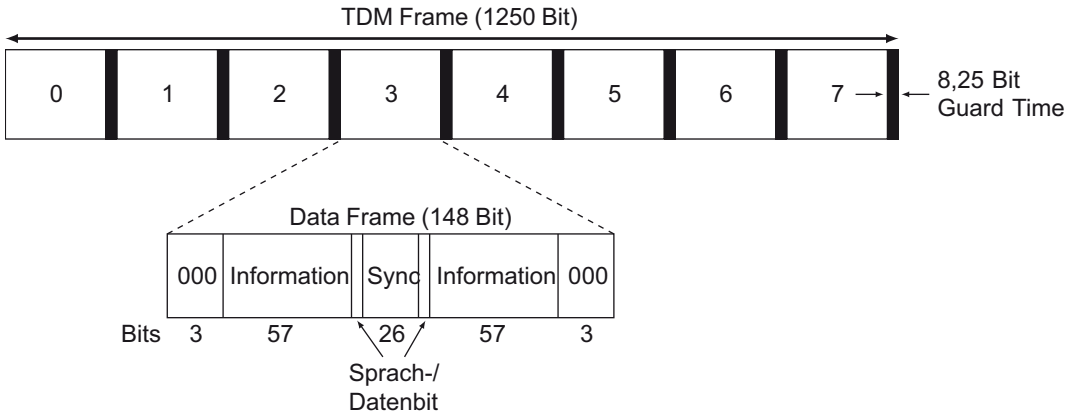


Abb. 4.4: TDM Frame und Data Frame in GSM (nach Tanenbaum und Wetherall, 2011)

Für jeden Data Frame bleiben somit 148 Bit, von denen jedoch nur 2· 57 Bit für die eigentliche Informationsübertragung genutzt werden können (vgl. Abb. 4.4). Die übrigen Bits werden zur Synchronisation und als Steuerbits (Sprach-/Datenbit) verwendet. Zudem enthalten nicht alle TDM Frames Nutzdaten. Je zwei von 26 TDM Frames sind stattdessen für Steuerzwecke reserviert. Diese Steuer-Frames sorgen für die Koordinierung der Mobilgeräte untereinander und mit der Basisstation.

Übung 4.2:

Welche maximale Datenrate für die Informationsübertragung können Sie anhand der Ihnen vorliegenden Informationen für ein Gerät im GSM-Standard ermitteln?

Der **Random Access Channel** ist ein spezieller Typ Steuer-Frame, über den sich Mobilgeräte bei einer Basisstation anmelden können, damit sie über einen weiteren Steuer-Frame einen Time Slot zugewiesen bekommen. Der Zugriff auf den Random Access Channel erfolgt nach dem Prinzip von slotted ALOHA.

Ein weiterer Typ Steuer-Frame koordiniert das Timing. Elektromagnetische Wellen bewegen sich zwar (annähernd) mit Lichtgeschwindigkeit, trotzdem dauert es je nach Entfernung eine gewisse Zeit, bis sie beim Empfänger auftreffen. Trifft das Signal deutlich zu spät ein, kollidiert es womöglich mit dem Data Frame auf einem benachbarten Time Slot. Um die unterschiedlichen Signallaufzeiten verschieden weit entfernter Geräte zu kompensieren, instruiert die Basisstation über den **Timing Advance** ein weit entferntes Mobilgerät, seine Data Frames entsprechend früher abzuschicken. Der Timing Advance wird in Bits angegeben und ist üblicherweise ein Wert zwischen 0 und 63.

Aus diesem Wert erklärt sich die Einschränkung der Zellgröße auf einen Radius von etwa 35 km: Bei einer Datenrate von 270 kbit/s beträgt die **Bit-Dauer**, d. h. die Zeit, in der ein einzelnes Bit versendet ist, 1 / 270 000 s. Beim größtmöglichen Timing Advance von 63 sendet eine Station ihre Data Frames somit bereits 63 / 270 000 s ≈ 0,233 ms früher. In dieser Zeit legt das Signal bei einer Lichtgeschwindigkeit von etwa 300 Millionen Meter pro Sekunde eine Strecke von 70 km zurück. Timing Advance kann somit die Signallaufzeiten von Mobilgeräten ausgleichen, sodass sich die Time Slots von zwei Mobilgeräten, die bis zu 70 km voneinander bzw. jeweils 35 km von der Basisstation entfernt senden, nicht überschneiden.

Übung 4.3:

Wie weit ist ein Gerät mit einem Timing Advance von 1 von der Basisstation ungefähr entfernt?

Von der in Übung 4.2 ermittelten Datenrate erreicht GSM in der Praxis nur einen Bruchteil. Störeinflüsse bei der Funkübertragung können zu Übertragungsfehlern führen. GSM verwendet deshalb ein aufwendiges Fehlererkennungs- und Korrekturverfahren mit zusätzlichem Overhead, sodass die Nettodatenrate pro Gerät etwa 9,6 kbit/s beträgt.

Die Datenrate von GSM kann mit verschiedenen Techniken gesteigert werden. Die Erweiterung **GPRS** (General Packet Radio Service) nutzt die gleichen Frequenzbänder und Frame-Strukturen wie GSM. Während GSM jedoch jedem Kommunikationsteilnehmer einen festen Time Slot für die gesamte Dauer der Verbindung zuweist (GSM ist deshalb verbindungsorientiert), werden die Time Slots mit GPRS dynamisch vergeben – GPRS ist somit paketorientiert. Eine Station kann in einem TDM Frame zudem je nach Geräteklasse bis zu vier Time Slots belegen, wenn kurzfristig viele Daten versendet werden müssen.

Zusätzlich ermöglicht GPRS die Wahl aus vier verschiedenen Codierungsschemata CS-1 bis CS-4, die abhängig von der Signalqualität unterschiedlich viele Bits für die Fehlererkennung und die Fehlerkorrektur benötigen. Das Codierungsschema CS-1 entspricht GSM mit einer Nettodatenrate von 9,6 kbit/s. CS-4 nutzt das Informations-Feld im Data Frame fast vollständig und erzielt eine Datenrate von etwa 21,4 kbit/s. Üblich ist bei GPRS eine Bündelung von vier Time Slots mit dem Codierungsschema CS-2 (13,4 kbit/s), wodurch sich eine Übertragungsrate von etwa 53,6 kbit/s ergibt.

EDGE (Enhanced Data Rates for GSM Evolution) baut ebenfalls auf GSM auf, ist wie GPRS paketorientiert und ermöglicht die Bündelung von Time Slots. Im Unterschied zu GSM und GPRS kennt EDGE effektivere Modulationsverfahren, die auf jeder Halbschwingung des Wellensignals bis zu drei Bit übertragen. Bei einem Übertragungsfehler gehen dabei aber auch mehr Bits als bei GSM bzw. GPRS verloren.

EDGE definiert neun verschiedene Übertragungsschemata MCS-1 bis MCS-9, um die Datenrate abhängig von der Signalqualität anzupassen. Das Schema MCS-1 entspricht im Wesentlichen GSM bzw. CS-1 bei GPRS (ein Bit pro Halbschwingung, großer Overhead für Fehlererkennung und Fehlerkorrektur). Das Schema MCS-9 wird bei optimaler Signalqualität eingesetzt und überträgt drei Bit pro Halbschwingung mit einem geringen Overhead für Fehlererkennung und Fehlerkorrektur. Pro Time Slot beträgt die Datenrate mit MCS-9 59,2 kbit/s. Bei einer Bündelung von vier Time Slots erzielt EDGE somit eine Nettodatenrate bis zu 236,8 kbit/s.

GPRS und EDGE werden als Zwischenstufen zu den Mobilfunknetzen der dritten Generation angesehen und deshalb mit **2,5G** bezeichnet. Der Vorteil dieser Erweiterungen ist es, dass sich die vorhandenen GSM-Netze leicht ausbauen lassen. Im besten Fall genügt ein Software-Update, damit eine Basisstation oder ein Endgerät GPRS bzw. EDGE unterstützt. Sollte neue Hardware notwendig sein, kann der Austausch schrittweise erfolgen. GPRS- und EDGE-fähige Endgeräte kommunizieren mit GSM-Basisstationen und eine GPRS- bzw. EDGE-fähige Basisstation unterstützt auch GSM-Geräte.

4.1.2 Mobilfunknetze der 3. Generation: UMTS und HSPA

Mit dem Mobilfunkstandard **UMTS** (Universal Mobile Telecommunications System) begann im Jahr 2000 der Ausbau der Mobilfunknetze der dritten Generation. UMTS nutzt je nach Land oder Kontinent unterschiedliche Frequenzbänder im Bereich von 700 bis 2100 MHz und kennt zwei verschiedene Modi.

Im Modus FDD (Frequency Division Duplex) gibt es wie bei GSM jeweils einen eigenen Frequenzbereich für den Uplink und den Downlink. In Deutschland teilen sich beispielsweise vier Netzbetreiber den Frequenzbereich von 1920–1980 MHz (Uplink) bzw. 2110-2170 MHz (Downlink), wobei jedem Netzbetreiber jeweils 10–20 MHz für Uplink und Downlink zur Verfügung stehen. Diese Bandbreite wird weiterhin in Kanäle von je 5 MHz unterteilt. Das Zugriffsverfahren auf einen dieser Kanäle ist CDMA (vgl. Abschnitt 3.3.4), d.h., im Unterschied zu GSM können alle Mobilgeräte den Kanal gleichzeitig nutzen. Im Modus FDD beträgt die Datenrate von UMTS bis zu 384 kbit/s.

Im Modus TDD (Time Division Duplex) teilen sich Basisstationen und Mobilgeräte den Uplink- und den Downlink-Kanal, senden jedoch in unterschiedlichen Time Slots. Somit kann das tatsächliche Datenaufkommen besser auf die verfügbare Bandbreite verteilt werden. Jeder Time Slot wird wiederum mittels CDMA von mehreren Sendern genutzt. Im Modus TDD sind Datenraten bis zu 1920 kbit/s möglich. TDD ist in Deutschland derzeit nicht in Betrieb.

HSPA (High Speed Packet Access) bezeichnet eine Familie von Erweiterungen für UMTS, die höhere Datenraten aufgrund effizienterer Modulationsverfahren ermöglichen (d.h. mehr Bits pro Halbschwingung übertragen). HSPA nutzt hierzu unter Anderem die Übertragungstechnik **MIMO** (Multiple Input Multiple Output) wobei ein Signal auf einem Frequenzband von mehreren Antennen gleichzeitig versendet und empfangen wird. Die Theorie hinter MIMO ist vergleichsweise komplex, da sie auf Interferenzphänomenen von Signalen (bspw. Reflexion und Streuung) basiert. Auf der Empfängerseite muss das ursprüngliche Signal aus den Interferenzmustern wieder rekonstruiert werden. MIMO ermöglicht es somit, einen Übertragungskanal effizienter zu nutzen, und es lassen sich mehr Bits pro Sekunde auf einem Frequenzband übertragen.

Der Standard **HSDPA** (High Speed Downlink Packet Access) beschreibt eine Reihe von „Kategorien“ mit unterschiedlichen Übertragungsschemata für den Downlink-Kanal, die Datenübertragungsraten zwischen 1,2 und 42 Mbit/s ermöglichen. HSPA auf dem Uplink nennt sich **HSUPA** (High Speed Uplink Packet Access) und erreicht nicht ganz so hohe Datenraten (0,73 bis 23 Mbit/s).

HSPA wird als Zwischenstufe zu den Mobilfunknetzen der 4. Generation angesehen und deshalb auch mit **3,5G** bezeichnet. In Deutschland ist HSDPA derzeit mit einer Datenrate von bis zu 42 Mbit/s und HSUPA bis 5,8 Mbit/s verfügbar.

4.1.3 Mobilfunknetze der 4. Generation: LTE und LTE-Advanced

Der Mobilfunkstandard **LTE** (Long Term Evolution) spezifiziert einige „evolutionäre" Verbesserungen von Techniken, die bereits in UMTS/HSPA-Netzwerken zum Einsatz kommen, wie eine effizientere Modulation, verbesserte Fehlerkorrektur und Mehrantennennutzung. Der wesentliche Unterschied zu UMTS besteht darin, dass LTE Bandbreiten zwischen 1,4 bis 20 MHz pro Kanal zulässt, während UMTS die Bandbreite für jeden Kanal auf 5 MHz begrenzt. Zudem stehen den Netzbetreibern üblicherweise deutlich mehr Frequenzbänder als für UMTS zur Verfügung. In Deutschland teilen sich die großen Netzbetreiber beispielsweise verschiedene Frequenzbänder im Bereich von 700, 800, 1800, 2000 und 2600 MHz. Die maximale Datenrate von LTE beträgt 300 Mbit/s im Downlink bzw. 75 Mbit/s im Uplink.

LTE wird vom Standardisierungskomitee der **ITU** (International Telecommunication Union) als **3,9G** klassifiziert, zählt also offiziell nicht zu den 4G-Mobil-funknetzen. Erst die Erweiterung **LTE Advanced**, die aufgrund verbesserter MIMO-Technologie Datenraten von jeweils bis zu 1000 Mbit für den Downlink und den Uplink ermöglicht, wird von der ITU als 4G bezeichnet.

4.1.4 Mobilfunknetze der 5. Generation

Seit 2018 ist der Mobilfunkstandard der fünften Generation **5G** in einigen Testnetzen in Betrieb. Der 5G-Standard nutzt die Frequenzbänder um 2,1 GHz (freiwerdende UMTS-Frequenzen) sowie Frequenzbänder im deutlich höherfrequenten Bereich ab 3,6 GHz. Aktuelle Spezifikationen nennen Kanalbandbreiten von maximal 100 MHz. Während in Standard-MIMO-Verfahren zwei bis vier Antennen genutzt werden, kommt in 5G-Netzen die Weiterentwicklung **Massive MIMO** mit bis zu 128 Antennen zum Einsatz.

Der chinesische Hersteller Huawei konnte im Testbetrieb im Frequenzbereich zwischen 4 und 8 GHz Datenraten von bis zu 18 GBit/s erzielen, bei mehreren Clients waren im Mittel über 6 GBit/s möglich. Die hohen Datenraten wurden unter anderem mittels Massive-MIMO und 200 MHz breiten Kanälen erreicht.

4.2 Sonstige Funknetzwerke

Neben den Mobilfunknetzen unterstützen Mobilgeräte eine Reihe weiterer Netzwerkstandards mit unterschiedlichen Anwendungsgebieten. Mittels WLAN verbinden sich Mobilgeräte mit einem lokalen Netzwerk, in der Regel einem Ethernet, und können so eine vorhandene kabelgebundene Breitbandinternetverbindung mitbenutzen (bspw. in einem „WLAN Hotspot" in einem öffentlichen Gebäude). Über Bluetooth können Mobilgeräte untereinander Daten über kurze Distanzen austauschen, ohne dabei auf eine vorhandene Netzwerkinfrastruktur angewiesen zu sein. Der Übertragungsstandard NFC (Near Field Communication) dient zum Datenaustausch über sehr kurze Distanzen von wenigen Zentimetern und wird vorwiegend zur bargeldlosen Bezahlung genutzt.

Im Unterschied zu den Mobilfunknetzen operieren diese Funknetzwerke auf freien, d.h. unregulierten, Frequenzbändern. Damit sind sie auch anfälliger gegen Störungen durch andere Geräte, die dieselben Frequenzbänder benutzen, etwa Garagenöffner, Schnurlostelefone oder Mikrowellengeräte. Die Reichweite dieser Funknetze ist üblicherweise auf einen deutlich kleineren Bereich als bei den Mobilfunknetzen begrenzt.

4.2.1 WLAN

WLAN (Wireless Local Area Network), auch **WiFi** genannt, ist der Oberbegriff für eine Reihe von Netzwerkstandards, die von der Standardisierungskommission **IEEE** (Institute of Electrical and Electronics Engineers) unter dem Code **802.11** zusammengefasst werden. Tab. 4.1 zeigt eine Auswahl unterschiedlicher Versionen dieses Standards:

Tab. 4.1: IEEE 802.11-Familie

Standard	Datenrate	Frequenzband
802.11	1-2 Mbit/s	2,4 GHz
802.11a	54 Mbit/s	5 GHz
802.11b	11 Mbit/s	2,4 GHz
802.11g	54 Mbit/s	2,4 GHz
802.11n	450 Mbit/s	2,4 GHz, 5 GHz

WLANs senden auf den lizenzfreien Frequenzbändern um 2,4 GHz (genauer: 2,4 bis 2,4835 GHz) sowie 5 GHz (5,15 bis 5,725 GHz). Das verfügbare Frequenzband wird üblicherweise in Kanäle von 20 bis 40 MHz Bandbreite unterteilt. WLANs besitzen Reichweiten von wenigen Metern innerhalb von Gebäuden bis zu einigen hundert Metern im freien Gelände.

WLANs können in zwei Modi operieren (vgl. Abb. Abb. 4.5): Im **Infrastrukturmodus** kommunizieren die Geräte mit einer fest installierten Basisstation, dem **Access Point**. Der Access Point leitet die Datenpakete entweder an kabelgebundene Geräte in einem lokalen Netzwerk oder andere über WLAN erreichbare Mobilgeräte weiter, ermöglicht aber auch eine Verbindung mit weit entfernten Geräten über das Internet. Im **Ad-hoc-Modus**, auch als **WiFi Direct** bezeichnet, organisieren sich die Geräte ohne eine feste Basisstation selbst. In beiden Fällen spannt das WLAN eine Funkzelle auf, die über einen eindeutigen Namen, den **SSID** (Service Set Identifier), identifiziert wird.

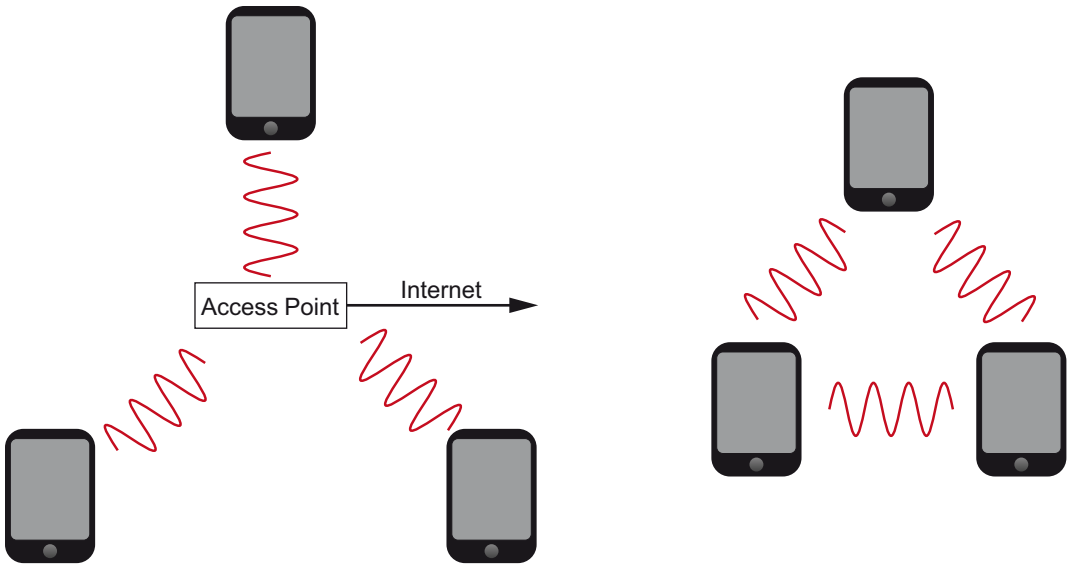


Abb. 4.5: WLAN-Netzwerkstruktur links: Infrastrukturmodus rechts: Ad-hoc Modus

Das Kanalzugriffsverfahren in WLANs ist slotted ALOHA in der Variante **CSMA/CA** (Carrier Sense Multiple Access with Collision Avoidance). In einem WLAN kann eine Station durch Abhören des Kanals aufgrund des **Hidden Station Problems** nicht sicher feststellen, ob der Kanal frei ist oder ein versendetes Datenpaket mit einem anderen Paket kollidiert. Angenommen, ein Mobilgerät A möchte ein Datenpaket an ein anderes Gerät oder eine Station B versenden (vgl. Abb. 4.6). B befindet sich innerhalb der Reichweite des Signals von A. Das Signal eines weiteren Geräts C kann ebenfalls von B empfangen werden, aufgrund der begrenzten Reichweite jedoch womöglich nicht von A. Wenn nun A und C gleichzeitig senden, kollidieren diese Pakete beim Empfänger B, ohne dass A oder C dies durch Abhören des Kanals erkennen kann. Eine Kollision kann somit erst durch Ausbleiben eines Acknowledgement Frame sicher erkannt werden.

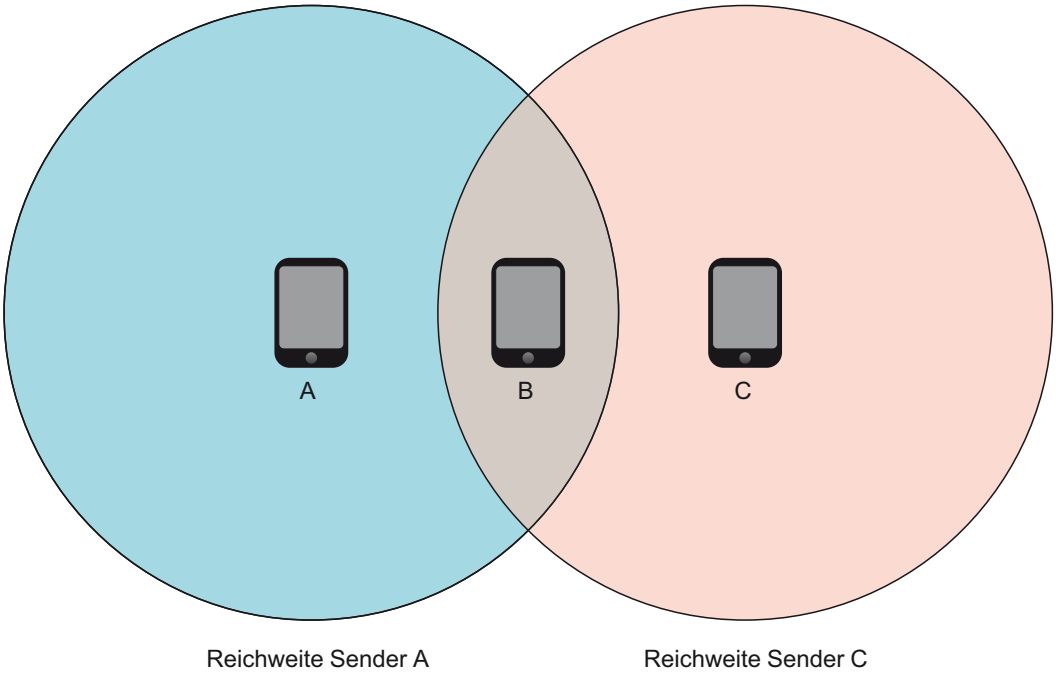


Abb. 4.6: Hidden Station Problem

CSMA/CA versucht unter diesen Umständen, die Häufigkeit von Kollisionen durch eine bestmögliche Kollisionsvermeidung zu reduzieren. Im Unterschied zu CSMA/CD sendet eine Station nicht sofort, sobald sie den Kanal unbelegt vorfindet. Stattdessen wählt sie einen zufälligen Wert x aus einem Intervall und wartet sodann x freie Time Slots ab, bevor sie zu senden beginnt. Auf diese Weise treten die Stationen weniger stark in Konkurrenz um den Übertragungskanal, sobald dieser von einer Station als frei wahrgenommen wird. Wird eine Kollision durch Ausbleiben des Acknowledgement Frame festgestellt, wird x aus einem doppelt so großen Intervall gewählt (entsprechend Exponential Backoff, vgl. Abschnitt 3.3.2).

Übung 4.4:

Weshalb wartet eine Station x freie Time Slots ab?

Neben slotted ALOHA mit CSMA/CA, um den Kanalzugriff innerhalb einer Funkzelle zu organisieren, verwenden WLANs zudem FDMA und CDMA. Damit sich benachbarte WLANs nicht gegenseitig stören, senden sie auf anderen Frequenzen und benutzen andere Chip-Sequenzen.

Die Steigerung der Datenraten in den 802.11-Standards (vgl. Tab. 4.1) basiert auf ähnlichen Techniken wie bei den Mobilfunknetzen, etwa effizientere Codierungs- und Modulationsverfahren oder verbesserte Mehrantennennutzung. Erwähnenswert ist das in 802.11a/g verwendete Modulationsverfahren **OFDM** (Orthogonal Frequency Division Multiplexing). Bei diesem Verfahren wird das verfügbare Frequenzband in kleinere Bänder unterteilt, sodass mehrere Bits gleichzeitig auf diesen Sub-Bändern übertragen werden können. Der Standard 802.11n ermöglicht zudem die parallele Nutzung von bis zu jeweils vier Empfangs- und Sendeantennen (MIMO 4x4).

Die 802.11-Familie definiert Authentifizierungs- und Verschlüsselungsmethoden, um das Abhören privater Daten in einem WLAN zu verhindern. Der ursprüngliche Standard **WEP** (Wired Equivalent Privacy) hat sich als unsicher herausgestellt. Wenn ein Angreifer beispielsweise die Kommunikation einer erfolgreichen WEP-Authentifizierung mithört, kann er aus diesen Informationen einen gültigen Authentifizierungsschlüssel berechnen und sich erfolgreich im Netz anmelden. WEP wurde deshalb inzwischen von dem sichereren Verfahren **WPA2** (WiFi Protected Access 2) abgelöst, das neben einem anderen Authentifizierungsmechanismus auch ein besseres Verschlüsselungsverfahren einsetzt.

4.2.2 Bluetooth

WLANs wurden in erster Linie entwickelt, um tragbare Computer, später auch Mobilgeräte, in eine vorhandene drahtgebundene Netzwerkinfrastruktur einzubinden (der Ad-hoc-Modus wurde erst vergleichsweise spät hinzugefügt und wird auch nur von wenigen Geräten unterstützt). Dagegen hatte der von einem Zusammenschluss mehrerer Firmen unter der Initiative von Ericsson Ende der Neunzigerjahre entwickelte Funknetzwerkstandard **Bluetooth** zum Ziel, Peripheriegeräte wie Drucker, Tastaturen oder Mäuse kabellos an ein Rechnersystem anzubinden.

Die daraus resultieren Anforderungen – günstig herzustellende Hardware, geringer Energiebedarf bei einer Überbrückung von vergleichsweise kurzen Distanzen – lassen sich direkt auf Mobilgeräte übertragen. Bluetooth gehört heute üblicherweise zur Grundausstattung jedes Mobilgeräts. Mittels Bluetooth können Mobilgeräte untereinander sowie mit einer unüberschaubaren Anzahl verschiedener Peripheriegeräte wie portable Lautsprecher, Kopfhörer, Freisprechanlagen, Fitness Tracker, Smart Watches, Game Controller oder ferngesteuerte Drohnen drahtlos kommunizieren.

Bluetooth sendet auf dem gleichen 2,4 GHz-Frequenzband wie WLAN. Um Störungen mit einem vorhandenen Netzwerk zu vermeiden, verwendet Bluetooth ein spezielles **Frequenzsprungverfahren**. Dabei wird das verfügbare etwa 80 MHz breite Band in 79 kleinere Bänder von jeweils 1 MHz unterteilt und der Kommunikationskanal wechselt bis zu 1600 mal pro Sekunde unter diesen Bändern. Das Frequenzsprungverfahren bei Bluetooth nennt sich **AFH** (Adaptive Frequency Hopping), da es „adaptiv“ die Bänder vermeidet, auf denen ein anderes Netzwerk, beispielsweise ein WLAN, sendet.

Die erste Version des Standards wurde 1999 verabschiedet und erzielt eine Datenrate von bis zu 732,2 kbit/s (vgl. Tab. 4.2). Version 2.0 verdreifacht diese Datenrate aufgrund eines verbesserten Modulationsverfahrens, das anstelle von einem Bit drei Bit pro Halbschwingung des Wellensignals überträgt. Bluetooth Version 3.0 sollte ein vorhandenes WLAN mitbenutzen und somit Datenraten von theoretisch bis zu 480 Mbit/s ermöglichen, konnte sich jedoch aufgrund diverser Schwierigkeiten nicht durchsetzen und wird heute nicht mehr weiter verfolgt.

Tab. 4.2: Bluetooth-Versionen

Version	Jahr	Datenrate
1.0	1999	732,2 kbit/s
2.0	2004	2,1 Mbit/s
3.0	2009	≥ 2,1 Mbit/s
4.0	2009	2,1 Mbit/s

Bluetooth 4.0 brachte vor allem Neuerungen in Bezug auf eine verbesserte Energieeersparnis. Im „Low Energy"-Modus verbrauchen die Geräte weniger Energie, die Datenrate sinkt jedoch auf etwa 260 kbit/s. Die derzeit aktuelle Version 4.2 erhöht die Datenrate im „Low Energy"-Modus auf bis zu 650 kbit/s.

Die Struktur eines Bluetooth-Netzwerks zeigt Abb. 4.7: Die Kommunikation läuft über ein ausgezeichnetes Gerät, den **Master**. Der Master spannt ein Netzwerk mit einer Reichweite von etwa 10 Metern auf. Ein Master kann in solch einem **Piconet** bis zu 255 weitere Geräte, die **Slaves**, ansprechen. Davon können jedoch nur sieben gleichzeitig aktiv sein. Die übrigen Slaves werden „geparkt", d.h. in einen speziellen Energiesparmodus versetzt, aus dem sie der Master wieder aufwecken kann. Die sieben aktiven Slaves teilen sich die verfügbare Bandbreite. Der Kanalzugriff innerhalb eines Piconet erfolgt mittels TDMA. Zwei Piconets können sich auch über einen ausgezeichneten Bridge Slave zu einem größeren Netzwerk zusammenschließen.

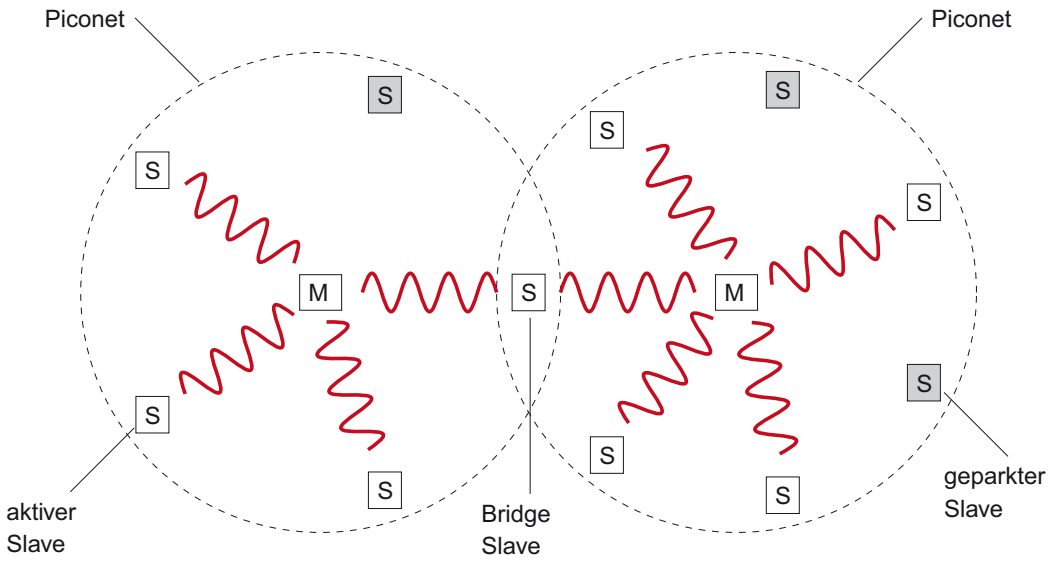


Abb. 4.7: Bluetooth-Netzwerkstruktur

Da die gesamte Steuerung des Netzwerks durch den Master erfolgt, benötigen die Slaves keine komplexe Hardware. Module für den Bluetooth-Empfang können somit sehr kostengünstig und platzsparend hergestellt werden. Sollten mehrere Geräte über Master-Funktionalität verfügen, erhebt sich das Gerät, das die Verbindung initiiert, zum Master über die anderen.

Bluetooth-Netzwerke können über Authentifizierung und Verschlüsselung mit einem 16-stelligen PIN-Code abgesichert werden. Es sind allerdings Angriffe bekannt, die bei einem zu kurzen PIN oder Mithören der Authentifizierung einen unerlaubten Zugriff auf das Netzwerk ermöglichen.

4.2.3 NFC

NFC (Near Field Communication) ist ein Übertragungsstandard über sehr kurze Distanzen (1–10 cm) und basiert auf der Übertragungstechnik **RFID** (Radio Frequency Identification). RFID unterscheidet zwischen aktiven und passiven Sendern. Ein passiver Sender besitzt keine eigene Stromversorgung. Stattdessen stellt der aktive Sender die Energie für die Rückantwort auf eine Anfrage über eine induktive Spule bereit, deren elektromagnetisches Feld einen Kondensator im passiven Sender auflädt. Die passive Komponente wird daher als **Transponder** bezeichnet, eine Wortkombination aus „Transmitter“ und „Responder“. Der Begriff **RFID Tag** ist ebenfalls üblich.

RFID-Transponder können sehr klein und günstig in einem Druckverfahren hergestellt werden. Sie dienen beispielsweise zur Kennzeichnung von Waren im Güterverkehr, werden in Haustiere implantiert oder speichern biometrische Daten in Personalausweisen.

Der vom NFC-Forum[13]}, einem Zusammenschluss aus über 100 Unternehmen, geleitete NFC-Standard nutzt die RFID-Technik zur Datenübertragung von und zu Mobilgeräten. NFC sendet auf einem etwa 1,8 MHz breiten Frequenzband im unregulierten Frequenzbereich um 13,56 MHz und erzielt Datenraten von bis zu 424 kBit/s bei einer Entfernung von 4 cm.

NFC ermöglicht zahlreiche Anwendungen, beispielsweise Zugangskontrolle, elektronische Tickets, bargeldloses Bezahlen kleiner Beträge („Micro Payment“), Austausch von Kontaktinformationen oder Auslesen von Informationen, die auf Poster oder Fahrpläne aufgedruckt sind. NFC definiert hierzu drei Modi:

Card Emulation: In diesem Modus verhält sich das Gerät wie ein passiver Transponder und kommuniziert mit einem externen Lesegerät.

Peer to Peer: Dieser Modus ermöglicht den Datenaustausch zwischen zwei aktiven NFC-fähigen Geräten.

Reader/Writer: Dieser Modus dient zum Auslesen eines passiven RFID Tags.

Unerlaubtes Abhören des Funkverkehrs ist aufgrund der niedrigen Reichweite von NFC zwar vergleichsweise schwierig, aber nicht unmöglich. Aus diesem Grund definiert NFC Authentifizierungs- und Verschlüsselungsverfahren. Im Modus Peer to Peer kann ein (sicherer) Verbindungsaufbau auch über Bluetooth oder WLAN erfolgen.

13. http://nfc-forum.org

Übung 4.5:

Was unterscheidet einen Transponder von einem Transceiver?

Zusammenfassung

Von GSM über UMTS bis LTE Advanced ist heute bereits die dritte Generation digitaler zellulärer Mobilfunknetze in Betrieb. Die Entwicklung der Mobilfunkstandards erfolgte häufig evolutionär über Zwischenstufen wie GPRS, EDGE, HSPA und LTE, sodass vorhandene Hardware weiterverwendet werden kann. Die Steigerungen der Datenraten begründen sich einerseits aus einer effizienteren Ausnutzung der verfügbaren Bandbreite, etwa durch dynamische Zuweisung von Time Slots und effizientere Codierungs- und Modulationsverfahren. Es kommen aber auch andere Frequenzbandaufteilungen und Kanalzugriffsverfahren (z.B. CDMA) zum Einsatz sowie eine Kanalbündelung über Mehrantennentechnik.

WLANs sind drahtlose lokale Netzwerke, die in zwei Modi operieren können. Im Infrastrukturmodus kommunizieren die Mobilgeräte mit dem Access Point. Dieser leitet die Datenpakete entweder an kabelgebundene Geräte in einem lokalen Netzwerk, an andere im gleichen WLAN angemeldete Geräte oder ins Internet weiter. Im Ad-hoc-Modus WiFi Direct dagegen organisieren sich die Mobilgeräte ohne feste Basisstation selbst.

Bluetooth ist ein Netzwerkstandard, um Peripherie drahtlos an ein Gerät anzubinden, dient aber auch zur Kommunikation von Mobilgeräten untereinander. Eines der Geräte erhebt sich dabei zum Master über die übrigen Geräte (die Slaves) und spannt ein Bluetooth-Piconet mit einer Reichweite von etwa zehn Metern auf. Da die Netzwerkkommunikation vom Master koordiniert wird, benötigen die Slaves keine komplexe Hardware.

RFID ist eine Übertragungstechnik zum Auslesen digitaler Informationen von passiven, stromlosen Transpondern. Der Standard NFC überträgt diese Technik auf Mobilgeräte, sodass diese sich entweder wie ein passiver Transponder (Modus Card Emulation) oder ein aktives Lesegerät (Modus Reader / Writer) verhalten. Ein dritter Modus (Peer to Peer) ermöglicht den Datenaustausch zwischen zwei aktiven NFC-fähigen Geräten.

Aufgaben zur Selbstüberprüfung

4.1 Ergänzen Sie den folgenden Lückentext:

Das Kanalzugriffsverfahren im 2G-Mobilfunkstandard GSM ist eine Kombination aus … … … … mit … Kanälen von je … kHz Bandbreite, sowie … … … … auf jedem Kanal mit 8 … … in einem … Frame. Zusätzlich kommt das Kanalzugriffsverfahren … im … … Channel zum Einsatz.

Die 2,5G-Erweiterung GPRS weist die … dynamisch zu und ist somit im Gegensatz zu GSM … . Die …-Erweiterung EDGE definiert zudem neun verschiedene Übertragungsschemata mit unterschiedlichem … für die … … … sowie verschiedene … .

Im 3G-Standard … senden die Stationen … auf einem Kanal mit … MHz Bandbreite. Das Kanalzugriffsverfahren ist … … … … .

Die 3,9G und 4G-Standards … und … … verwenden … Bandbreiten von bis zu … MHz pro Kanal. Zudem stehen üblicherweise mehrere … zur Verfügung.

4.2 Weshalb verwenden die WLAN-Standards der 802.11-Familie eine Variante von slotted ALOHA als Kanalzugriffsverfahren und welche Besonderheit weist diese Variante auf?

4.3 Welches Problem kann bei 5G-Netzwerken insbesondere im ländlichen Bereich auftreten?

A. Lösungen der Übungen im Text

1.1 Aufgrund ihrer Größe sind Tablets eher stationär, d. h., sie werden mehr in einem abgegrenzten Bereich wie zu Hause oder im Zug benutzt und weniger wie ein Smartphone ständig am Körper getragen. Deshalb ist Telefoniefunktion oder GPS-Ortsbestimmung oft verzichtbar. Eine weitere Rolle spielt der Preis, denn Tablets werden zum Teil deutlich günstiger als Smartphones verkauft (das iPhone ist etwa doppelt so teuer wie die Tablet-Version iPad). Dafür besitzen Tablets einen größeren Bildschirm, womit sie sich einfacher bedienen lassen. Aufgrund ihrer Größe kann auch leistungsfähigere Hardware oder ein größerer Akku verbaut werden.

1.2 Die Idee, eine maßgeschneiderte Plattform für Mobilanwendungen zu entwickeln, entstand bereits in den Siebzigerjahren. Erste Geräte kamen in den Achtziger- und Neunzigerjahren auf den Markt, konnten sich jedoch aus verschiedenen Gründen nicht durchsetzen.

Erst der Ausbau der digitalen Funknetze Ende der Neunzigerjahre sowie die Einführung der Multitouch-Steuerung mit dem ersten iPhone führten zu der rasanten Verbreitung von Smartphones und Tablets. Hinzu kamen diverse Entwicklungen im Hardwarebereich, vgl. auch Abschnitt 1.3.8.

Auch das große Softwareangebot im App Store bzw. Google Play ist ein nicht zu unterschätzender Faktor, der zur Verbreitung beigetragen hat, vgl. auch Abschnitt 1.2.

1.3 Bei einer Displayauflösung von 750 x 1334 Bildpunkten müssen bei einer Bildwiederholfrequenz von 60 Hz mindestens 750\ x 1334 x 60 = 60 030 000 Bildpunkte pro Sekunde gefüllt werden.

1.4 Tablets verfügen im Vergleich zu Smartphones über größere Gehäuse. Somit ist auch mehr Platz für eine leistungsfähigere GPU sowie einen größeren Akku vorhanden.

Die im Vergleich zu Smartphones größeren Displays besitzen häufig auch eine höhere Auflösung, was eine höhere Fill Rate erfordert.

1.5 Das Floating Gate ist eine zusätzliche Steuerleitung in einer NAND-Speicherzelle. Durch eine spezielle elektrische Isolierung kann eine elektrische Ladung im Floating Gate permanent gespeichert werden.

1.6 Damit Ladungen abfließen können, muss der Gegenstand leitfähig sein. Ein Handschuh jedoch isoliert.

1.7 Scrolling funktioniert auf einem kapazitiven Display besser, da kein mechanischer Druck ausgeübt und ausgewertet werden muss. Das kapazitive Display kann auf Bewegungen schneller reagieren, da die x- und y-Koordinaten gleichzeitig bestimmt werden.

1.8 Falls Sie das Mobilgerät hochkant vor der Brust halten, zeigt die y-Achse nach oben. Der Roll-Winkel kann dann nicht bestimmt werden.

1.9 Ein Accelerometer misst die lineare Beschleunigung in eine vorgegebene Richtung. Ein Gyroskop dagegen misst die Winkelgeschwindigkeit einer Rotation um eine bestimmte Achse. Beide messen Änderungsraten der Position bzw. Orientierung. Der Accelerometer misst jedoch Änderungsraten zweiter Ordnung (zweite Ableitung der Position), das Gyroskop erster Ordnung (erste Ableitung des Winkels).

1.10 Um einen Motor mit 50 mW Leistung zwei Stunden lang anzutreiben, wird eine Energie von 100 mWh benötigt. Bei einer Betriebsspannung von 3 Volt benötigt der Akku eine Nennkapazität von 100 mWh/3 Volt ≈ 33,33 mAh.

1.11 Der Energieverbrauch eines Mobilgeräts wird durch geeignete Maßnahmen wie Drosselung der CPU-Frequenz, Reduzierung der Betriebsspannungen oder Abschalten bestimmter Komponenten optimiert. Je nach Anwendungsszenario (z.B. Spielen, langes Telefonieren, GPS-Ortsbestimmung) kann der Energieverbrauch somit signifikant variieren.

2.1 Anwender einer iOS-Anwendung kommen in erster Linie mit den Komponenten aus dem UIKit Framework in Berührung.

Als Anwendungsentwickler/-in werden Sie ebenfalls häufig mit den Komponenten aus dem UIKit Framework arbeiten, zudem mit den beiden Foundation Frameworks. Welche weiteren Frameworks Sie einsetzen, hängt von der Art der Anwendung ab, die Sie erstellen möchten.

2.2 Das MVC-Entwurfsmuster ermöglicht es, die Komponenten unabhängig voneinander zu entwickeln und zu warten. So kann beispielsweise das Datenhaltungsmodell leicht ausgetauscht und etwa von einer lokalen MySQL-Datenbank auf eine Datenbank auf einem entfernten Server oder ein Cloud-Modell umgestellt werden. Auch die grafische Benutzeroberfläche kann leicht ausgetauscht oder angepasst werden, ohne dass dies Änderungen in den anderen Komponenten zur Folge hat.

2.3 Im Beispiel bilden das Label und der Button die Views und der Controller ist die Klasse `ViewController`. Ein eigenes Datenhaltungsmodell existiert in diesem einfachen Beispiel nicht.

2.4 Gerätehersteller müssen die Linux-Kernel-Schicht mit ihren Schnittstellen zum Hardware Abstraction Layer implementieren.

2.5 Die Ressourcen in einem Android-Projekt sind in Unterordnern des Ordners „res“ zu finden.

2.6 Die Browser Engine unter iOS heißt „Webkit“. Der Zugriff auf WebKit erfolgt über das WebKit Framework, das Bestandteil der zweiten Schicht Core Services ist (vgl. Abschnitt 2.2). Unter Android heißt die Browser Engine „Chromium“ und ist eine Komponente der dritten Schicht Libraries (vgl. Abb. 2.9).

Anmerkung: Bis Android 4.4 wurde unter Android ebenfalls Webkit als Browser Engine eingesetzt.

3.1 Elektromagnetische Wellen benötigen kein Übertragungsmedium und breiten sich (im Vakuum) konstant mit Lichtgeschwindigkeit aus.

3.2 Sichtbares Licht wird eher weniger für Datenübertragung von Mobilgeräten genutzt, jedoch in Glasfaser-Kabeln und zur Audioübertragung bei Stereoanlagen.

3.3 Die Basisstationen senden im Frequenzbereich von 935–960 MHz, also mit einer Bandbreite von 25 MHz. Diese Bandbreite wird weiter in 124 Kanäle unterteilt, somit verbleiben etwa 200 kHz Bandbreite pro Kanal. Die tatsächliche Datenübertragungsrate ist neben der Bandbreite auch von dem gewählten Modulationsverfahren, der Signalqualität bzw. dem Störabstand sowie der Ausnutzung des Kanals (Sendet die Station die ganze Zeit? Sendet die Station auf mehreren Kanälen gleichzeitig?) abhängig.

3.4 Die iOS- und Android-Systemarchitekturen sind ebenfalls als Schichtenmodelle organisiert, vgl. Abschnitt 2.2.1 und Abschnitt 2.3.1.

3.5 Die Adressierung über die IP geschieht in der Vermittlungsschicht (OSI) bzw. Internetschicht (TCP/IP). Die Vermittlungsschicht liefert die Pakete an die gewünschte Netzwerkadresse bzw. das gewünschte Empfangsgerät aus, unter Umständen indirekt über mehrere Netzwerkknoten.

Die Port-Angabe spezifiziert die Anwendung, für die die Daten bestimmt sind (z.B. Port 80: Webbrowser). Sie wird von der Transportschicht (OSI und TCP/IP) verwaltet.

3.6 Die Chance, dass eine Kollision auftritt, ist abhängig von der Paketlänge und der Anzahl versendeter Pakete. Mit der Länge eines Pakets wächst auch die Vulnerabilitätszeit. Zu klein sollten die Pakete auch nicht sein, denn dann müssen die Stationen unter Umständen sehr viele Pakete senden.

3.7 Ein Netzbetreiber kann mit TDMA die verfügbare Bandbreite recht gut auf viele Teilnehmer verteilen. Zudem könnte ein Netzbetreiber Kunden, die mehr zahlen, auch mehrere Time Slots zuweisen.

3.8 Gesendet wird die Summe

$$\begin{aligned}\mathbf{S} &= \mathbf{A}+\overline{\mathbf{B}}+\mathbf{C}+\overline{\mathbf{D}}\\ &= (+1,+1,+1,+1)\\ &+ (-1,+1,-1,+1)\\ &+ (+1,+1,-1,-1)\\ &+ (-1,+1,+1,-1)\\ &= (0,+4,0,0)\end{aligned}$$

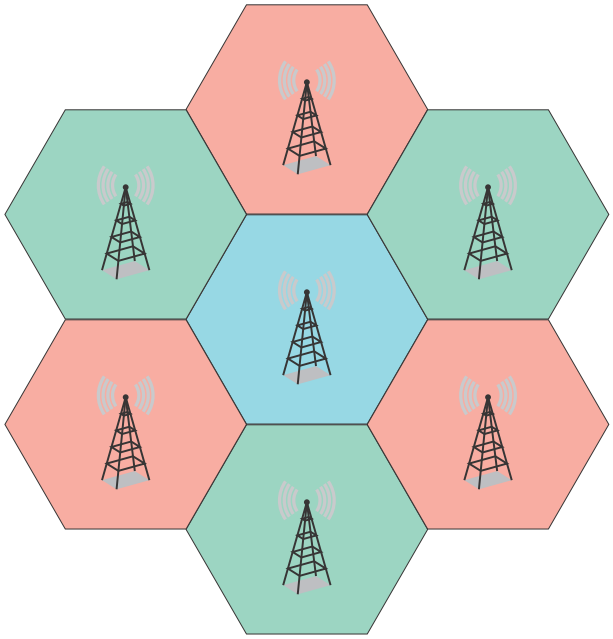
Um das Signal von Station B zu rekonstruieren, wird die Summe mit B's Chip-Sequenz multipliziert (inneres Produkt):

$$\mathbf{S}\cdot\mathbf{B} = \frac{1}{4}(0,+4,0,0)\cdot(+1,-1,+1,-1) = \frac{1}{4}\cdot -4 = -1$$

Analog für Station C:

$$\mathbf{S}\cdot\mathbf{C} = \frac{1}{4}(0,+4,0,0)\cdot(+1,+1,-1,-1) = \frac{1}{4}\cdot 4 = 1$$

4.1 Es genügen drei sich nicht überschneidende Frequenzbänder. Um das einzusehen, färben wir die Zellen so ein, dass keine zwei benachbarten Zellen die gleiche Farbe besitzen. Hierzu reichen drei Farben aus:



Jede Farbe entspricht dabei einem Frequenzband.

4.2 Von 26 TDM Frames bleiben 24 Frames (etwa 92,3 %) für die eigentliche Kommunikation übrig. Von diesen 24 TDM Frames kann eine Station jeweils $2 \cdot 57$ von 1250 Bits (9,12 %) nutzen.

Somit ergibt sich eine Datenrate pro Gerät von

$$270\,\text{kbit/s} \cdot (24/26) \cdot (2 \cdot 57)/1250 \approx 22{,}73\,\text{kbit/s}$$

Die tatsächliche Datenrate von GSM-Netzen ist jedoch noch deutlich niedriger.

4.3 Ein Timing Advance von 1 bedeutet, dass das Gerät eine Bit-Dauer früher sendet. Während einer Bit-Dauer legt das Signal die Strecke von

$$300\,000\,000\,\text{m/s} \cdot 1/270\,000\,\text{m} \approx 1111\,\text{m}$$

zurück. Die Entfernung zur Basisstation beträgt die Hälfte dieser Strecke, also ungefähr 550 Meter.

4.4 Erst wenn *x freie* Time Slots verstrichen sind, kann ein Sender davon ausgehen, dass die momentane Netzauslastung vergleichsweise gering ist und damit die Chance steigt, dass das Paket ohne Kollisionen beim Empfänger ankommt. Würde *x* dagegen auch bei einem belegten Time Slot dekrementiert, würde es zu mehr Kollisionen kommen.

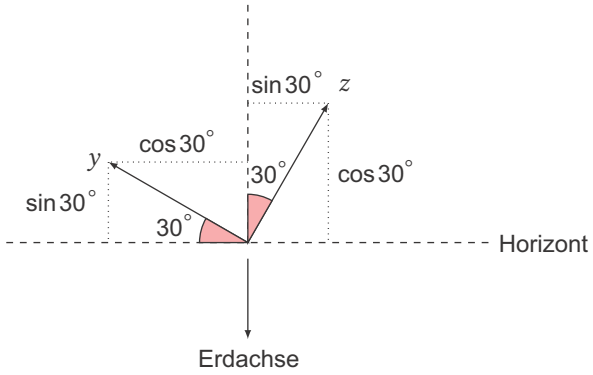
4.5 Ein Transceiver (Transmitter + Receiver) ist ein Sende- und Empfangsbaustein mit eigener Stromversorgung. Ein Transponder (Transmitter + Responder) ist ein passiver Baustein ohne eigene Stromversorgung. Die für die Antwort benötigte Energie wird vom Sender bereitgestellt.

B. Lösungen der Aufgaben zur Selbstüberprüfung

1.1 RISC-Prozessoren besitzen einen vergleichsweise einfachen Befehlssatz und eine große Anzahl von Registern. Da die Befehle einheitlich strukturiert sind, kann der Befehlsdurchsatz einfacher mittels Pipelining optimiert werden. Komplexe Operationen müssen jedoch durch eine Reihe einfacher Befehle nachgebildet werden.

Durch den einfachen und einheitlichen Befehlssatz benötigt ein RISC-Prozessor nur eine vergleichsweise kleine Chip-Fläche und vergleichsweise wenig Energie.

1.2 Wird das Gerät mit einem Pitch-Winkel von 30° gehalten, stellt sich zwischen dem Erdhorizont und der lokalen *y*-Achse des Geräts der gleiche Winkel ein, ebenso zwischen der dem Erdmittelpunkt entgegengesetzten Achse und der lokalen *z*-Achse des Geräts:



Bei einer Erdbeschleunigung von 1 g misst der *y*-Accelerometer aufgrund der Sätze der Trigonometrie einen Wert von $\sin 30° \cdot 1\,g = 0{,}5\,g$ und der *z*-Accelerometer $\cos 30° \cdot 1\,g \approx 0{,}866\,g$. Der Accelerometer, der die Beschleunigung in *x*-Richtung misst, zeigt einen Wert von 0 g.

1.3 Wenn wir von den Angaben in Abb. 1.5 ausgehen, erhalten wir wegen $\mathbf{z} \cdot -\mathbf{x} = \mathbf{y}$

$$0{,}01\,\mathrm{kg} \cdot \mathrm{m/s^2} = 2 \cdot 0{,}005\,\mathrm{kg} \cdot 1\,\mathrm{m/s} \cdot x\,\mathrm{rad/s}$$

Auflösen nach *x* ergibt die Winkelgeschwindigkeit von 1 rad/s.

Nach drei Sekunden wurde das Gyroskop somit um 3 rad bzw. ungefähr 172 Grad gedreht.

1.4 Liegt das Gerät mit dem Display nach unten flach auf einem Tisch, wird dies über den Accelerometer erkannt und das Display wird abgeschaltet.

Bei der Positionsbestimmung mittels GPS muss von einer bekannten Position aus keine neue Geoposition von den Satelliten empfangen werden, wenn Accelerometer und Gyroskop keine oder nur sehr geringe Änderungsraten anzeigen.

Befindet sich das Gerät für längere Zeit in Ruhelage, kann es in einen Tiefschlafmodus versetzt und der Prozessortakt sowie die Betriebsspannung reduziert werden.

2.1 Wenn wir die Verkaufszahlen aus Abschnitt 1.2 zugrunde legen, dann würde bei einer Erdbevölkerung von sieben Milliarden Menschen mehr als jeder zweite über ein iOS- oder Android-Gerät verfügen. Diese Zahl ist jedoch vermutlich zu hoch.

Eine realistischere Schätzung basiert auf der Annahme, dass ein Smartphone etwa zwei Jahre in Gebrauch ist, bevor es durch ein neueres Modell ersetzt wird. Einige Personen besitzen zudem neben Smartphones noch weitere iOS- oder Android-Geräte wie Tablets. Da beide Systeme seit 2007/2008 auf dem Markt sind, könnte eine Person über durchschnittlich vier iOS- bzw. Android-Geräte verfügen. Somit besitzen vermutlich bereits eine Milliarde Menschen (ein Siebtel der Erdbevölkerung) ein iOS- oder Android-Gerät.

2.2 Eine Anwendung kann eine andere Komponente, bspw. eine Activity, implizit über die Angabe der von dieser Komponente zu erfüllenden Anforderungen starten. Auf welche Anforderungen die zu startende Komponente reagieren soll, wird über einen Intent-Filter im App Manifest der zugehörigen Anwendung definiert. Diese lose Kopplung funktioniert auch über Anwendungsgrenzen hinweg.

2.3 Die View-Komponente zur Eingabe von Text unter Android heißt „EditText“. Zunächst ist der Layout-Datei aus Code 2.4 hinzuzufügen:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout

    <EditText
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Bitte Text eingeben"
        android:id="@+id/edittext"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        android:id = "@+id/textview_hello"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@id/edittext"/>
  <!-- wie vor -->
</android.support.constraint.ConstraintLayout>
```

Code B.1: Modifizierte Layout-Datei „activity_main.xml“

Zu beachten ist, dass das Attribut `app_layout_constraintTop_toTopOf` der TextView-Komponente auf die ID der EditText-Komponente gesetzt werden muss, damit der TextView unterhalb der EditText-Komponente platziert wird.

Die Methode `onButtonClick` der Klasse `MainActivity` wird wie folgt modifiziert:

```
public void onButtonClick(View view) {
  EditText edittext =
  (EditText)findViewById(R.id.edittext);
  TextView textview =
  (TextView)findViewById(R.id.textview_hello);
  textview.setText(edittext.getText());
}
```

Code B.2: Modifizierte Methode `onButtonClick` der `MainActivity`

3.1 Im Unterschied zu FDMA senden die Stationen mit ALOHA und TDMA nicht die ganze Zeit auf ihrem Frequenzband. ALOHA und TDMA müssen die Daten deshalb auf irgendeine Weise puffern, wozu sich digitale Daten besser eignen.

3.2 ALOHA ist unkoordiniert. Die Stationen müssen sich weder einig werden, wer welchen Frequenzbereich, welche Time Slots oder welche Chip-Sequenz erhält. Dafür hat ALOHA das Problem, dass der Durchsatz gerade dann sinkt, wenn viele Stationen senden, also eine hohe Datenrate notwendig wäre.

3.3 Wenn $\mathbf{S} \cdot \mathbf{T} = 0$, dann gibt es genauso viele positive (+1) wie negative (−1) Einzelprodukte $S_i \cdot T_i$. Das Vorzeichen jedes Produkts dreht sich um, wenn statt $\mathbf{T}$ mit dem Komplement $\overline{\mathbf{T}}$ multipliziert wird. Somit folgt $\mathbf{S} \cdot \overline{\mathbf{T}} = 0$.

$\mathbf{S} \cdot \mathbf{S} = 1$ gilt, da jedes Einzelprodukt +1 ergibt, denn$(+1)^2 = (-1)^2 = 1$. Multiplizieren wir $\mathbf{S}$ mit dem Komplement, ist jedes der Produkte negativ, denn entweder gilt $S_i \cdot \overline{S}_i = +1 \cdot -1$ oder $S_i \cdot \overline{S}_i = -1 \cdot +1$, somit:

$$\frac{1}{m}\sum_{i}^{m} S_i \cdot \overline{S}_i = -1$$

4.1 Das Kanalzugriffsverfahren im 2G-Mobilfunkstandard GSM ist eine Kombination aus **Frequency Division Multiple Access** mit **124** Kanälen von je **200** kHz Bandbreite, sowie **Time Division Multiple Access** auf jedem Kanal mit acht **Time Slots** in einem **TDM** Frame. Zusätzlich kommt das Kanalzugriffsverfahren **ALOHA** im **Random Access Channel** zum Einsatz.

Die 2,5G-Erweiterung GPRS weist die **Time Slots** dynamisch zu und ist somit im Gegensatz zu GSM **paketorientiert**. Die **2,5G**-Erweiterung EDGE definiert zudem neun verschiedene Übertragungsschemata mit unterschiedlichem **Overhead** für die **Fehlererkennung und Korrektur** sowie verschiedene **Modulationsverfahren**.

Im 3G-Standard **UMTS** senden die Stationen **gleichzeitig** auf einem Kanal mit **5** MHz Bandbreite. Das Kanalzugriffsverfahren ist **Code Division Multiple Access**.

Die 3,9G- und 4G-Standards **LTE** und **LTE Advanced** verwenden **variable** Bandbreiten von bis zu **20** MHz pro Kanal. Zudem stehen üblicherweise mehrere **Frequenzbänder** zur Verfügung.

4.2 Der WLAN-Standard sollte kompatibel zu Ethernet sein und Ethernet verwendet slotted ALOHA mit CSMA/CD (vgl. Abschnitt 3.3.2.). Zudem erfordert ALOHA keine zentrale Koordinierung des Netzwerks und eignet sich gut für paketorientierte digitale Netze.

Im Gegensatz zu einem Ethernet können Kollisionen aufgrund des Hidden Station Problems jedoch nur durch das Ausbleiben eines Acknowledgement Frame robust erkannt werden. Um das Risiko einer Kollision zu reduzieren, wartet eine Station zunächst eine zufällige Anzahl freier Time Slots ab, bevor sie mit dem Senden beginnt (CSMA/CA).

4.3 Hochfrequente Signale besitzen nur eine geringe Reichweite. Um die versprochenen Datenraten zu erzielen, müssen im ländlichen Bereich sehr viele Basisstationen aufgestellt werden, was die Kosten erhöht.

C. Literaturverzeichnis

Apple (2015).
iOS Technology Overview. https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iOSTechOverview.pdf.

Apple (2015 b).
The Swift Programming Language. https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/

Google (2017).
Kotlin and Android. https://developer.android.com/kotlin/index.html

Tanenbaum A.; Wetherall D. (2011).
Computer Networks (fifth edition). New Jersey USA,: Prentice Hall.

D. Abbildungsverzeichnis

E. Tabellenverzeichnis

F. Codeverzeichnis

G. Sachwortverzeichnis

H. Einsendeaufgabe Typ A

Technik in der App-Entwicklung Teil 1

Einsendeaufgabencode:
AET01-XX1-K06

Name:	Vorname:	Tutor/-in:
Postleitzahl und Ort:	Straße:	Datum:
Matrikel-Nr.:	Studiengangs-Nr.:	Note:
Heftkürzel: **AET01**	Druck-Code: **0319K06**	Unterschrift:

Bitte reichen Sie Ihre Lösungen über StudyOnline ein. Falls Sie uns diese per Post senden wollen, dann fügen Sie bitte die Aufgabenstellung und den Einsendeaufgabencode hinzu.

1. Angenommen, einem Smartphone-Akku steht genügend Energie zur Verfügung, um einen Prozessor eine Stunde lang mit der Taktfrequenz f zu betreiben. Sei weiterhin angenommen, das Power Management kann den Prozessortakt gegebenenfalls auf $f/4$ reduzieren.

   Wie viele Minuten muss der Prozessortakt auf $f/4$ gedrosselt werden, damit der Prozessor genau zwei Stunden lang mit Energie versorgt werden kann?

   15 Pkt.

2. Vergleichen Sie die beiden Softwareplattformen iOS und Android. Welche Gemeinsamkeiten und Unterschiede weisen ihre Systemarchitekturen auf? Wie sind Anwendungen für die jeweilige Plattform strukturiert? Welche Unterschiede müssen Sie bei der Anwendungsentwicklung beachten?

   20 Pkt.

3. Welche Kanalzugriffsverfahren kommen in den folgenden Funknetzwerkstandards jeweils zum Einsatz:

   a) GSM

   b) UMTS

   c) WLAN

   d) Bluetooth

   10 Pkt.

4. Das im Mobilfunkstandard UMTS verwendete Modulationsverfahren nennt sich QPSK (Quadrature Phase Shift Keying) und codiert vier verschiedene Zustände pro Halbschwingung.

   a) Welche Datenrate kann ein Übertragungskanal in UMTS nach Nyquist höchstens erzielen?

   b) Welche Datenrate erzielt UMTS tatsächlich und wie erklärt sich die Differenz?

   20 Pkt.

5. a) Spezifizieren Sie zwei verschiedene Sätze („Codes“) orthogonaler Chip-Sequenzen der Länge 2 in der bipolaren Darstellung.

6 Pkt.

b) Können Chip-Sequenzen ungerade Längen aufweisen? Begründen Sie.

4 Pkt.

c) Die *Walsh-Matrix W* ist eine quadratische Matrix, in der die Chip Sequenzen zeilenweise eingetragen sind. Für einen Code der Länge 2 mit den Chip Sequenzen **S** = (S_1, S_2)und **T** = (T_1, T_2) erhalten wir die 2 x 2-Matrix

$$W(2) = \begin{bmatrix} S_1 S_2 \\ T_1 T_2 \end{bmatrix}$$

Diese Matrix entspricht der sogenannten *Hadamard-Matrix* erster Ordnung $H(2^1)$. Die Hadamard-Matrix zweiter Ordnung $H(2^2)$ist eine 4 x 4-Matrix und wird rekursiv über die Vorschrift

$$H(2^2) = \begin{bmatrix} H(2^1) H(2^1) \\ H(2^1) - H(2^1) \end{bmatrix}$$

bestimmt. Allgemein gilt:

$$H(2^k) = \begin{bmatrix} H(2^{k-1}) H(2^{k-1}) \\ H(2^{k-1}) - H(2^{k-1}) \end{bmatrix}$$

Zeigen Sie, dass Sie mithilfe dieser Vorschrift Ihre Codes aus Aufgabenteil a) auf Codes der Länge 4 erweitern können.

Welche Chip-Sequenzen der Länge 2 generieren den in Abschnitt 3.3.4 gezeigten Code?

Welche Länge haben die Chip-Sequenzen in der nächsten Iteration der rekursiven Vorschrift?

10 Pkt.

6. Erörtern Sie, weshalb sich die Übertragung über Satelliten in Mobilfunknetzen nicht durchsetzen konnte.

Weshalb eignet sich Satellitenübertragung dagegen für GPS?

15 Pkt.

WILHELM BÜCHNER HOCHSCHULE

Eine Hochschule der Klett Gruppe

Wilhelm Büchner Hochschule
Hilpertstraße 31
64295 Darmstadt

06151 3842-404
Mo.-Fr. 8:00 bis 20:00 Uhr
Sa. 9:00 bis 15:00 Uhr

beratung@wb-fernstudium.de

www.wb-fernstudium.de

Fragen und Anregungen direkt zum Studienheft bitte an folgende Adresse: autor@wb-fernstudium.de. Wir stellen dann für Sie den Kontakt zum/zur Autor:in oder Tutor:in her.

WILHELM BÜCHNER HOCHSCHULE