

# **Webgestütztes GPIO Management am Beispiel des BeagleBone Black**

**Bachelorarbeit im Fachbereich Medienproduktionstechnik an der  
Fachhochschule Köln**

Caspar Friedrich  
Geboren am 16. Oktober 1986  
Mat.-Nr. 11062078

Köln, den 1. November 2014

Betreut durch Prof. Dr. Klaus Ruelberg  
Zweitprüfer: Prof. Dr. Luigi Lo Iacono



# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>9</b>
1.1. Zielsetzung . . . . .	9
1.2. Definitionen . . . . .	9
<b>I. Grundlagen</b>	<b>11</b>
<b>2. Hardware</b>	<b>13</b>
2.1. Single-board Computer (SBC) . . . . .	13
2.2. System on a Chip (SOC) . . . . .	14
2.2.1. BeagleBone Black . . . . .	14
<b>3. Betriebssysteme</b>	<b>17</b>
3.1. Linux . . . . .	17
3.1.1. Linux Distributionen . . . . .	17
<b>4. Webtechnologien</b>	<b>19</b>
4.1. Webserver . . . . .	19
4.1.1. Lighttpd . . . . .	19
4.1.2. Weitere Webserver . . . . .	19
4.2. WebSockets . . . . .	19
4.2.1. WebSockets vs. PHP . . . . .	19
4.3. Node.js . . . . .	19
<b>II. Konfiguration des Betriebssystems</b>	<b>21</b>
<b>5. Betriebssystem</b>	<b>23</b>
<b>6. Zusätzliche Software</b>	<b>25</b>
6.1. nodejs . . . . .	25
<b>III. Implementierung</b>	<b>27</b>
<b>7. Implementierung</b>	<b>29</b>
7.1. boneserver . . . . .	29

7.2. Webinterface . . . . .	29
<b>IV. Fazit</b>	<b>31</b>
<b>V. Anhang</b>	<b>33</b>
<b>A. Tabellen</b>	<b>35</b>

# Abbildungsverzeichnis

7.1. Test figure . . . . . 29



# Tabellenverzeichnis

A.1. BeagleBone Black Expansion Header (P8)	36
A.2. BeagleBone Black Expansion Header (P9)	37





# 1. Einleitung

Das *Internet of Things* ist ein rasant wachsender Anwendungsbereich. Angetrieben von einer zunehmenden Akzeptanz digitaler Systeme und der günstigen Entwicklung der Baugröße, Leistung und Zuverlässigkeit und nicht zuletzt der sinkende Preis haben dazu geführt, dass digitale Systeme heute in allen Lebensbereichen anzutreffen sind. Durch die Annäherung der Hardware-Hersteller an die „Hobby“-Entwickler und den Erfolg von Arduino und Co. ist der Entwicklungsaufwand eigener Hardware-Projekte erheblich kleiner als noch vor wenigen Jahren.

## 1.1. Zielsetzung

Nach dieser Entwicklung ist es naheliegend, auch den Laboralltag dergestalt zu digitalisieren, dass die verschiedenen Anforderungen und Aufgaben mit einem einzigen flexiblen System realisiert werden können. Heute sind viele Geräte und Programme sehr spezialisiert, so dass man in komplexen Arbeitszusammenhängen viele verschiedene Systeme braucht. Dadurch ist oft ein hoher finanzieller Aufwand erforderlich.

Ziel dieser Arbeit ist es, ein Steuersystem für Messanwendungen zu entwickeln, das sich einfach konfigurieren lässt, flexibel in der Anwendung ist und gleichzeitig kostengünstig bleibt. Besonderes Augenmerk soll dabei auf der ausreichenden Verfügbarkeit verschiedener GPIO<sup>1</sup> liegen. Insbesondere Pulsbreitenmodulation und Analog/Digital-Konverter sind für Messanwendungen interessant. Die anfallenden Messdaten sollen protokolliert werden und extern verwendbar sein. Die zu entwickelnde Applikation soll auf keinen bestimmten Anwendungsfall spezialisiert sein. Sie soll vielmehr dem Anwender die Möglichkeit geben, sich eine für sein jeweiliges Projekt passende Umgebung zusammenzustellen. Weiter soll das System ohne ständige Überwachung und ohne Anbindung an ein Labornetzwerk autark arbeiten können. Es soll möglich sein, eine entfernte Messstation aufzubauen, die nach belieben via LAN, WLAN oder auch GSM konfiguriert und überwacht werden kann.

[GRAFIK!!!]

## 1.2. Definitionen

In dieser Arbeit wird ein Singleboard-Computer des Typs **BeagleBone Black Rev. A5C** verwendet. Andere Versionen dieses Computers sind, sofern kompatibel, ebenfalls

---

<sup>1</sup>General Purpose Input/Output

## *1. Einleitung*

verwendbar, allerdings nicht getestet. Um eine gute Lesbarkeit zu ermöglichen ist mit „BeagleBone“ im Folgenden immer diese Version gemeint.

## **Teil I.**

# **Grundlagen der verwendeten Technologien und Hardware**



## 2. Hardware

### 2.1. Single-board Computer (SBC)

Ein Single Board Computer oder auch SBC, zu deutsch ein Einplatinenrechner, ist ein Computersystem bei dem alle für die Verwendung nötigen Bauteile auf einer einzelnen Platine verbaut sind. Hierbei sind neben den essenziellen Komponenten wie Prozessor, RAM und ROM auch Controller für verschiedene I/O-Schnittstellen, Oszillatoren oder Co-Prozessoren verbaut. Single Board Computer werden vor allem in der Industrie als Steuersysteme eingesetzt, da sie oft billiger und flexibler sind als fest verdrahtete Steuersysteme. Mit zunehmender Miniaturisierung und steigender Leistungsfähigkeit finden SBC's heute auch in alltäglichen Geräten wie Autos, Waschmaschinen oder Fernbedienungen Verwendung.

Technisch gesehen sind auch erste Heimcomputer wie der *C64* oder *Atari ST* Single Board Computer, allerdings lassen sich diese ohne Ein- und Ausgabegeräte wie Maus, Tastatur, Bildschirm nicht sinnvoll einsetzen und werden in der Regel nicht als solche bezeichnet.

#### Schnittstellen

Single Board Computer verfügen, je nach Anwendungsgebiet, über eine Vielzahl verschiedener analoger und digitaler I/O-Schnittstellen.

Übliche Schnittstellen sind

- GPIO, darunter digitale I/O, PWM und Analog/Digital Converter (ADC)
- UART<sup>1</sup>
- SPI
- I<sup>2</sup>C

Aktuelle (Entwickler-)Systeme haben in der Regel einen oder mehrere USB-Anschlüsse (sowohl Client als auch Host Ports sind üblich), oder zumindest einen JTAG-Port, was die Programmierung wesentlich vereinfacht. Des Weiteren verfügen leistungstärkere Systeme oft auch über einen Grafikausgang<sup>2</sup>.

---

<sup>1</sup>Hierüber ist eine Implementierung der verbreiteten RS232/422/485-Schnittstelle möglich und auch üblich

<sup>2</sup>Meist HDMI oder eine der Miniaturvarianten

### 2.2. System on a Chip (SOC)

Eng verknüpft mit der Entwicklung der SBC ist das Konzept der System-on-a-Chip bzw. SOC. Hierbei werden die meisten oben genannten Komponenten eines Systems direkt in einem Einzelnen IC verbaut. Meist sind nur ROM und Controller für höhere Schnittstellen USB oder LAN (in manchen Fällen auch Grafik) extern angebunden.

Heutige Single-board Computer mit einem SOC können sehr leistungsstark sein, sind als Mehrkernsystem aufgebaut und haben Taktraten von mehreren GHz. Diese Computer sind vom Design her stark an Desktop-Systeme angepasst und können oft mit einem vollwertigen Linux- oder Windows-System betrieben werden.

Gerade bei diesen leistungsstarken SOC's hat sich die ARM-Architektur durchgesetzt. 1983 als Nebenprojekt gegründet hatte die 32-Bit-Architektur bereits 2002 einen Marktanteil von fast 80% (2)

Single Board Computer lassen sich (sehr) grob in zwei Klassen unterteilen:

#### 1. Leistungsschwache Systeme

Die Taktraten dieser Prozessoren liegen üblicherweise unter 50MHz, in seltenen Fällen über 100MHz. Diese Systeme werden meist direkt programmiert und finden vor allem im low energy-Sektor Anwendung.

#### 2. Leistungsstarke Systeme

Hier liegen die Taktraten meist im GHz-Bereich. Hauptanwendungsbereiche sind Mobilfunksysteme und embedded computing in der Industrie. Gerade im Mobilfunkbereich sind oft Mehrkernsysteme anzutreffen und es wird bis auf wenige Ausnahmen oberhalb eines Betriebssystems, meist Linux bzw. Android, programmiert.

#### 2.2.1. BeagleBone Black

Für diese Arbeit verwende ich einen BeagleBone Black Rev. A5C (im Folgenden BeagleBone), ein quelloffenes Entwickler-Board mit einem ARM® Cortex™-A8 Prozessor (Single Core) von Texas Instruments.

Die wichtigsten Features:

- 1GHz Taktrate
- 512MB DDR3 RAM
- 2GB<sup>3</sup> Onboard Flash Memory
- 10/100 Mbit/s Ethernet

---

<sup>3</sup>4GB ab Rev. C

- 69<sup>4</sup> GPIO mit mehreren PWM-Ausgängen und analogen Eingängen.
- Verhältnismäßig geringer Preis von ca. 45 €

---

<sup>4</sup>Laut Dokumentation. 27 sind ohne weitere Konfiguration direkt verfügbar





## 3. Betriebssysteme

Da die Recourcen des BeagleBone Black sehr begrenzt sind, wird für diese Arbeit ein schlankes Betriebssystem benötigt, welches nur wenig Speicher benötigt und geringen Leistungs-Overhead verursacht. Für diesen Zweck gibt es spezielle Versionen der bekannten Betriebssysteme wie Microsoft Windows oder Linux sowie verschiedene „uinoide“ Betriebssysteme.

### 3.1. Linux

Linux hat den Vorteil, dass nahezu alle Software als source code verfügbar ist und im Zweifel angepasst werden kann. Zu dem ist es üblich Lizenzen zu verwenden, die eine nicht-kommerzielle Anwendung sowie Anpassungen kostenfrei zulassen.

Ein eigenes Linux zu entwickeln oder ein *build system*<sup>1</sup> zu verwenden wäre aus Sicht der Performance sicherlich die beste Wahl und ist auch in der Industrie weitgehend üblich, würde allerdings den Rahmen dieser Arbeit sprengen. Zu dem gibt es einige sehr schlanke und bereits für den BeagleBone angepasste Linux Distributionen.

#### 3.1.1. Linux Distributionen

BeagleBoard.org bietet auf für den BeagleBone Black zwei verschiedene Distributionen an: Ångström und Debian. Beide Distributionen haben ihre Vor- und Nachteile. Ein weiteres Projekt, welches sich unter Entwicklern großer Beliebtheit erfreut ist Arch Linux, welches auch als Basis für diese Anwendung dienen soll.

**The Ångström Distribution** ist auf dem BeagleBone vorinstalliert und stellt die Hauptdistribution dar. Diese Distribution nutzt ein build system und findet im wesentlichen Anwendung bei Speichersystemen wie NAS oder FTP-Server, wichtigstes feature ist daher der geringe Leistungs- und Speicherbedarf. Bei dieser Distribution muss allerdings nahezu jede nicht-standard Software selbst kompiliert und eingerichtet werden.

**Debian Linux** gilt im allgemeinen als (rock-)stable und ist eine der verbreitetsten Distributionen, zu dem basieren einige weitere namhafte Distributionen auf Debian Linux. Stärke und gleichzeitig auch Schwäche dieser Distribution sind die langen und umfangreichen Softwaretests. Wenn ein Paket in den offiziellen repositories verfügbar ist kann

---

<sup>1</sup>Einige Distributionen verwenden ein sog. build system bei dem die benötigten Kernel-Module und Software-Pakete selbst zusammengestellt werden können.

### 3. Betriebssysteme

man zwar davon ausgehen, dass es fehlerfrei funktioniert und zu allen anderen angebotenen Paketen kompatibel ist, allerdings liegt es meist nicht mehr in der aktuellen Version vor. Das kann gerade bei Software aus dem Bereich Netzwerk/Internet problematisch werden.

**ArchLinux** Arch Linux hat gegenüber den oben genannten Distributionen zwei wesentliche Vorteile: Zum einen gibt es eine (Sub-)Distribution speziell für ARM-Prozessoren, bei der das Basissystem mit ca. 500MB sehr schlank ist und zum anderen ein sehr umfangreiches software repository mit sehr hoher Aktualität. Zusätzlich gibt es das *Arch User Repository*, ein freies Repository in dem jeder Nutzer seine Pakete einstellen kann. Sämtliche in diesem Projekt verwendete Software lässt sich entweder direkt aus den offiziellen Repositories installieren oder aus den User Repositories kompilieren. Zwar kann es durchaus passieren, dass die eingestellte software nicht out-of-the-box funktioniert aber in der sehr aktiven Community hinter Arch Linux bekommt man relativ schnell Hilfe.

Arch Linux ARM verwendet ein sog. *Rolling Release*<sup>2</sup> es ist daher wesentlich einfacher das System aktuell und sicher zu halten. Da die Kernel-Entwicklung derzeit sehr schnell voran schreitet, wird praktischerweise, zusätzlich zur regulären Kernel-Entwicklung, ein legacy-Paket mit einer stabilen Version gepflegt. Es muss nach einem Update des Kernel-Paketes nicht erst die Kompatibilität wieder hergestellt werden.

---

<sup>2</sup>Kontinuierliche Software-Entwicklung bei der Pakete separat aktuell gehalten und weiter entwickelt werden, gibt keine explizite Betriebssystemversion (Wikipedia)

## **4. Webtechnologien**

### **4.1. Webserver**

Was ist ein Webserver? Welche sind die verbreitetsten und was sind ihre Besonderheiten.

#### **4.1.1. Lighttpd**

Warum wird Lighttpd verwendet?

#### **4.1.2. Weitere Webserver**

Apache

### **4.2. WebSockets**

#### **4.2.1. WebSockets vs. PHP**

### **4.3. Node.js**

Was ist Node.js, wie wird es verwendet.



## **Teil II.**

# **Konfiguration des Betriebssystems**



## 5. Betriebssystem

Abweichend von der regulären Kernel-Entwicklung wird ein etwas älterer Kernel der Version 3.8 verwendet. Dieser wird über das Paket *legacybla* bereitgestellt.





## 6. Zusätzliche Software

Zusätzlich zu den mitgelieferten Paketen der Distribution werden noch ein HTTP server, ein FTP server und die JavaScript/Node.js engine benötigt. Zusätzlich wird noch ein Proxy server benötigt um mit geringem Aufwand SSL-Verschlüsselte Verbindungen zu ermöglichen und nach außen über einen einzigen Netzwerk-Port zu kommunizieren zu können.

**haproxy** HAProxy ist ein Proxy server, der eigentlich eingesetzt wird um TCP-Anfragen auf mehrere Server zu verteilen. Wesentlich interessanter für diese Arbeit ist allerdings, dass der HAProxy nativ SSL-Verschlüsselte Verbindungen verarbeiten kann und dabei in der Basis sehr leicht zu konfigurieren ist.<sup>(1)</sup>

In diesem wird wird HAProxy eingesetzt um WebSocket requests von regulären HTTP requests zu trennen und auf unterschiedliche Dienste weiterzuleiten. Ziel dieser Maßnahme ist es nach außen die gesamte Website hinter einem Port zu betreiben obwohl die beiden Prozessen völlig von einander getrennt sind. So ist die gefahr, dass, bei einem Feldeinsatz, der Port für den WebSocket server von einer Firewall blockiert wird minimal. Die website ist entweder vollständig oder überhaupt nicht zu erreichen. Auch ist der der WebSocket server, der systembedingt mit root-Rechten laufen muss, ausschließlich per WebSocket über den Proxy zu erreichen und ist so gegenüber Angriffen von außen weitgehend sicher.

Ein weiterer wichtiger Punkt ist, dass sich für jeden Server die maximale Anzahl der aktiven Verbindungen bequem per Config File einstellen lassen. So kann ohne besondere Programmierung sichergestellt werden, dass immer nur eine Verbindung zum WebSocket server besteht. Alle weiteren verbindungsanfragen werden auf eine Warteliste gesetzt und weitergeleitet sobald der Websocket Server wieder frei wird.

**lighttpd** Lighttpd ist ein sehr leichtgewichtiger Webserver, der

**vsftpd** Ein ftp-server...

### 6.1. nodejs

Zusätzlich zu den von Google mitgelieferten Modulen werden noch einige Module benötigt um einen WebSocket Server zur verfügung zu stellen und die Steuerung der GPIO zu übernehmen.

## 6. Zusätzliche Software

**ws** Eine der einfachsten aber auch eine der schnellsten (wenn nicht die schnellste) WebSocket-Implementierung. WS wird von vielen komplexeren WebSocket-Modulen als Grundlage bzw. als WebSocket-Unterstützung verwendet. Das Modul ist sehr einfach zu verwenden, fehlende Funktionalität gegenüber z. B. *Socket.IO* fällt in diesem Projekt nicht ins Gewicht, da der Webserver separat via Lighttpd zur Verfügung gestellt wird.

### **shelljs**

### **bonescript**

Beschreibung der bonescript<sup>1</sup> library.

---

<sup>1</sup><https://github.com/jadonk/bonescript>

# **Teil III.**

## **Implementierung**



# 7. Implementierung

Das webinterface besteht aus zwei Teilen: Einem WebSocket server, der die Steuerung der GPIO erledigt und einem Webserver, der die Dokumente ausliefert.

## 7.1. boneserver

Der WebSocket server ist via Node.js implementiert und verwendet die *bonescript*<sup>1</sup> library zur steuerung der GPIO.

## 7.2. Webinterface

---

<sup>1</sup><https://github.com/jadonk/bonescript>

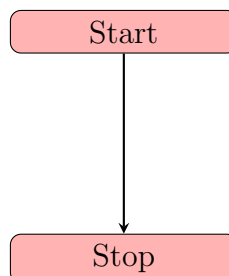


Abbildung 7.1.: Test figure



## **Teil IV.**

### **Fazit und Erweiterungsmöglichkeiten**





# **Teil V.**

## **Anhang**



## A. Tabellen

## 36

e.g. OUTPUT GPIO(mode7) 0x07 pullup, 0x17 pullup, 0x27 no pullup/down  
e.g. INPUT GPIO(mode7) 0x27 pulldown, 0x37 pullup, 0x47 no pullup/down

Head_pin	SPINS	ADDR/OFFSET	Name	GPIO NO.	Mode7	Mode6	Mode5	Mode4	Mode3	Mode2	Mode1	Mode0	Pin	Notes
P9_01			GND											Ground
P9_02			GND											Ground
P9_03			DC_3.3V											250mA Max Current
P9_04			DC_3.3V											250mA Max Current
P9_05			VDD_5V											1A Max Current (only if DC jack powered)
P9_06			VDD_5V											1A Max Current (only if DC jack powered)
P9_07			SYS_5V											250mA Max Current
P9_08			SYS_5V											250mA Max Current
P9_09			PWR_BUTTON											Has a 5V Level (pulled up by TPS65217C)
P9_10			SYS_RESETn											
P9_11	28	0x870/070	UART1_RXD	30	gpio[30]	uart4_rxd_mux2		mmio_sclsd	rmii2_crs_dv	gpmc_cs14	mi2_crs	RESET_OUT	A10	
P9_12	30	0x878/078	GPIO_28	60	gpio[28]	mcasp0_adclk_mux3		gpmc_dir	mmio2_dat3	gpmc_cs16	mi2_col	gpmc_wait0	T17	NB: GPIOs limit current to 4-6mA output and approx. 8mA on input.
P9_13	29	0x874/074	UART1_TXD	31	gpio[31]	uart4_txd_mux2		mmio2_sclsd	rmii2_nerr	gpmc_cs15	mi2_nerr	gpmc_wgn	U18	
P9_14	18	0x848/048	EHFPM1A	50	gpio[18]	ehfpm1A_mux1		gpmc_a18	mmio2_dat1	gpmc_cs17	mi2_dat1	gpmc_wgn	U17	
P9_15	16	0x840/040	GPIO_16	48	gpio[16]	ehfpm1A_mux1		gpmc_a16	mmio2_dat1	gpmc_cs18	mi2_dat1	gpmc_wgn	U14	
P9_16	19	0x844/044	EHFPM1B	51	gpio[19]	ehfpm1B_mux1		gpmc_a19	mmio2_dat2	gpmc_cs19	mi2_dat2	gpmc_wgn	R13	
P9_17	87	0x95c/15c	I2C1_SCL	5	gpio[5]			gpmc_a19	mmio2_dat2	gpmc_cs20	mi2_dat2	gpmc_wgn	T14	
P9_18	86	0x958/158	I2C1_SDA	4	gpio[4]			gpmc_a19	mmio2_dat2	gpmc_cs21	mi2_dat2	gpmc_wgn	A16	
P9_19	95	0x97c/17c	I2C2_SCL	13	gpio[13]			gpmc_a17	mmio2_dat2	gpmc_cs22	mi2_dat2	gpmc_wgn	B16	
P9_20	94	0x978/178	I2C2_SDA	12	gpio[12]			gpmc_a17	mmio2_dat2	gpmc_cs23	mi2_dat2	gpmc_wgn	D17	Allocated (Group: gpmc_mux12_2_pins)
P9_21	85	0x954/154	UART2_TXD	3	gpio[3]	EMU3_mux1		gpmc_a17	mmio2_dat2	gpmc_cs24	mi2_dat2	gpmc_wgn	D18	Allocated (Group: gpmc_mux12_2_pins)
P9_22	84	0x950/150	UART2_RXD	2	gpio[2]	EMU3_mux1		gpmc_a17	mmio2_dat2	gpmc_cs25	mi2_dat2	gpmc_wgn	B17	
P9_23	17	0x844/044	GPIO_17	49	gpio[17]	ehfpm0_sync0		gpmc_a17	mmio2_dat2	gpmc_cs26	mi2_dat2	gpmc_wgn	A17	
P9_24	97	0x984/184	UART1_TXD	15	gpio[15]			gpmc_a17	mmio2_dat2	gpmc_cs27	mi2_dat2	gpmc_wgn	V14	
P9_25	107	0x98c/18c	GPIO_21	117	gpio[21]			gpmc_a17	mmio2_dat2	gpmc_cs28	mi2_dat2	gpmc_wgn	D15	
P9_26	96	0x980/180	UART1_RXD	14	gpio[14]			gpmc_a17	mmio2_dat2	gpmc_cs29	mi2_dat2	gpmc_wgn	A14	Allocated (Group: mcasp0_pins)
P9_27	105	0x944/144	GPIO_19	115	gpio[19]			gpmc_a17	mmio2_dat2	gpmc_cs30	mi2_dat2	gpmc_wgn	D16	
P9_28	103	0x94c/14c	SPI1_CS0	113	gpio[17]			gpmc_a17	mmio2_dat2	gpmc_cs31	mi2_dat2	gpmc_wgn	C13	Allocated (Group: mcasp0_pins)
P9_29	101	0x944/144	SPI1_CS0	111	gpio[15]			gpmc_a17	mmio2_dat2	gpmc_cs32	mi2_dat2	gpmc_wgn	C12	Allocated (Group: mcasp0_pins)
P9_30	102	0x948/148	SPI1_CS1	112	gpio[16]			gpmc_a17	mmio2_dat2	gpmc_cs33	mi2_dat2	gpmc_wgn	B13	Allocated (Group: mcasp0_pins)
P9_31	100	0x940/140	SPI1_CS1	110	gpio[14]			gpmc_a17	mmio2_dat2	gpmc_cs34	mi2_dat2	gpmc_wgn	D12	
P9_32			VDD					gpmc_a17	mmio2_dat2	gpmc_cs35	mi2_dat2	gpmc_wgn	A13	Allocated (Group: mcasp0_pins)
P9_33			AGND					gpmc_a17	mmio2_dat2	gpmc_cs36	mi2_dat2	gpmc_wgn	C8	Voltage reference for ADC (NB: 1.8V)
P9_34			AGND					gpmc_a17	mmio2_dat2	gpmc_cs37	mi2_dat2	gpmc_wgn		NB: 1.8V tolerant
P9_35			AGND					gpmc_a17	mmio2_dat2	gpmc_cs38	mi2_dat2	gpmc_wgn	A8	Ground for ADC
P9_36			AGND					gpmc_a17	mmio2_dat2	gpmc_cs39	mi2_dat2	gpmc_wgn	B8	NB: 1.8V tolerant
P9_37			AGND					gpmc_a17	mmio2_dat2	gpmc_cs40	mi2_dat2	gpmc_wgn	B7	NB: 1.8V tolerant
P9_38			AGND					gpmc_a17	mmio2_dat2	gpmc_cs41	mi2_dat2	gpmc_wgn	A7	NB: 1.8V tolerant
P9_39			AGND					gpmc_a17	mmio2_dat2	gpmc_cs42	mi2_dat2	gpmc_wgn	B6	NB: 1.8V tolerant
P9_40			AGND					gpmc_a17	mmio2_dat2	gpmc_cs43	mi2_dat2	gpmc_wgn	C7	NB: 1.8V tolerant
P9_41A	109	0x944/144	CLKOUT2	20	gpio[20]	EMU3_mux0		gpmc_a17	mmio2_dat2	gpmc_cs44	mi2_dat2	gpmc_wgn	D14	Both signals are connected to P21 of P11
P9_41B			GPIO_20	116	gpio[20]			gpmc_a17	mmio2_dat2	gpmc_cs45	mi2_dat2	gpmc_wgn	D13	Both signals are connected to P21 of P11
P9_42A	89	0x944/144	GPIO_7	7	gpio[7]	xdma_event_innr2		gpmc_a17	mmio2_dat2	gpmc_cs46	mi2_dat2	gpmc_wgn	C18	Both signals are connected to P22 of P11
P9_42B			GPIO_18	114	gpio[18]			gpmc_a17	mmio2_dat2	gpmc_cs47	mi2_dat2	gpmc_wgn	B12	Both signals are connected to P22 of P11
P9_43			GND					gpmc_a17	mmio2_dat2	gpmc_cs48	mi2_dat2	gpmc_wgn		- See P9_50 of the SRM
P9_44			GND					gpmc_a17	mmio2_dat2	gpmc_cs49	mi2_dat2	gpmc_wgn		Allocated (Group: mcasp0_pins)
P9_45			GND					gpmc_a17	mmio2_dat2	gpmc_cs50	mi2_dat2	gpmc_wgn		Ground
P9_46			GND					gpmc_a17	mmio2_dat2	gpmc_cs51	mi2_dat2	gpmc_wgn		Ground
P9_Header	cat SPINS	ADDR +	Name	GPIO NO.	Mode 7	Mode 6	Mode 5	Mode 4	Mode 3	Mode 2	Mode 1	Mode 0	CPU	Notes
	Allocated	44e10000		(Mode 7)										For updates see: <a href="http://www.derekmolloy.ie">www.derekmolloy.ie</a>
		Offset from:												Please e-mail me directly at:
		44e10800												<a href="mailto:derek@derekmolloy.ie">derek@derekmolloy.ie</a>
														If you notice a mistake

Bit 6	Bit 5	Bit 4	Bit 3	Bit 2,1,0
Slew Control	Receiver Active	Enable Pullup/Pulldown	Enable Pullup/Pulldown	Mux Mode
0 Fast	0 Disable	0 Pulldown select	0 Enabled	000 Mode 0 to
1 Slow	1 Enable	1 Pullup select	1 Disabled	111 Mode 7

e.g. OUTPUT gpio(mode7) 0x07 pullup, 0x37 pulldown, 0x77 no pullup/down  
e.g. INPUT gpio(mode7) 0x27 pulldown, 0x37 pullup, 0x77 no pullup/down

by Derek Molloy ([www.derekmolloy.ie](http://www.derekmolloy.ie))

Tabelle A.2.: BeagleBone Black Expansion Header (P9)



# Literaturverzeichnis

- [1] KÜHNAST, Charly: Passthrough und Offloading: HTTPS balancieren mit HA-Proxy 1.5. In: *ADMIN Magazin* (2014), Nr. 3, S. 32–34
- [2] STILLER, Andreas: Die ARM-Story. In: *c't - magazin für computertechnik* (2002), Nr. 2, S. 70
- [3] WEISER, Mark: The Computer for the 21st Century. In: *Scientific American* (1991), Nr. 265, S. 94–104