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 19. Oktober 2014

Betreut durch Prof. Dr. Klaus Ruelberg
Zweitprüfer: Prof. Dr. Luigi LoIacono

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Das *Internet of Things* ist ein rasant wachsender Anwendungsbereich. Angetrieben durch eine zunehmende Akzeptanz digitaler Systeme und durch eine günstige Entwicklung in Baugröße, Leistung und Zuverlässigkeit und nicht zuletzt im Preis haben dazu geführt, dass digitale Systeme heute in allen Lebensbereichen anzutreffen sind. Durch die Annäherung der Hardware-Hersteller an die „Hobby"-Entwickler und dem 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 zu digitalisieren und die Verschiedenen Anforderungen und Aufgaben auf einem einzigen flexibelen System zu realisieren.

Ziel dieser Arbeit ist es dabei ein Steuersystem für Messanwendungen zu schaffen, das sich einfach konfigurieren lässt, flexibel in der Anwendung ist und gleichzeitig kostengünstig bleibt. Besonderes Augenmerk soll dabei auf der ausreichenden Verfügbarkeit verschiedener GPIO[1] 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 hin spezialisiert sein sondern dem Endanwender die Möglichkeit geben sich eine für sein Projekt passende Umgebung zusammen zu stellen. Weiter soll das System auch autark, vor allem über das Internet, arbeiten können. Es soll möglich sein eine entfernte Messtation auf zu bauen, die nach belieben via LAN, WLAN oder auch GSM konfiguriert und überwacht werden kann.

[GRAFIK!!!]

1.2. Definitionen

In dieser Arbeit wird ein **BeagleBone Black Rev. A5C** verwendet. Andere Versionen des Boards sind, sofern kompatibel, ebenfalls verwendbar allerdings nicht getestet. Um eine gut Lesabrkeit zu ermöglichen ist mit „BeagleBone" im Folgenden immer diese Version gemeint.

[1] General Purpose Input/Output

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 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, Waschmaschienen 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 Ausgeabegräte 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

- Digitale IOs
- PWM
- Analog/Digital Converter (ADC)
- UART[1]
- SPI
- I^2C

[1]Hierüber ist eine Implementierung der verbreiteten RS232/422/485-Schnittstelle möglich und auch üblich

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 wesendlich vereinfacht. Des weiteren verfügen leistungsstärkere Systeme oft auch über einen Grafikausgang[2].

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 SOCs 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 überlicherweise 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 ligen 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

[2]Meist HDMI oder eine der Miniaturvarianten

- 512MB DDR3 RAM
- 2GB[3] Onboard Flash Memory
- 10/100 Mbit/s Ethernet
- 69[4] GPIO mit mehreren PWM-Ausgängen und analogen Eingängen.
- Verhältnismäßig geringer Preis von ca. 45 €

[3] 4GB ab Rev. C
[4] Laut Dokumentation. 27 sind ohne weitere Konfiguration direkt verfügbar

3. Betriebsysteme

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 „uinxoide“Betriebssysteme.

3.1. Linux

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

Ein eigenes Linux zu entwickeln oder ein *build system*[1] 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 wesendlichen 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 eigerichtet werden.

Debian Linux gilt im allgemeinen ans (rock-)stable und ist eine der verbreitetsten Distributionen, zu dem basieren einige weitere namhafte Distributionen aud 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

[1] Einige Distributionen verwenden ein sog. build system bei dem die benötigten Kernel-Module und Software-Pakete selbst zusammengestellt werden können.

man zwar davon ausgehen, dass es fehlerfrei funktioniert und zu allen anderen angebotenen Pakete 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 wesendliche 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 Aktulaität. 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 aud 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*[2] es ist daher wesendlich einfacher das System aktuell und sicher zu halten. Da die Kernel-Entwicklung derzeit sehr schnell vorran schreitet, wird praktischerweise, zusätlich 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.

[2]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 Pakteten 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. Wesendlich 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.(1)

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.

ws Eine der einfachsten aber auch eine der schnellsten (wenn nicht die schnellste) WebSocket-Implemtierung. 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[1] library.

[1] https://github.com/jadonk/bonescript

Teil III.

Implementierung

7. Implementierung

Das webinterface besteht im 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 und verwendet die *bonescript*[1] library zur steuerung der GPIO.

7.2. Webinterface

[1] https://github.com/jadonk/bonescript

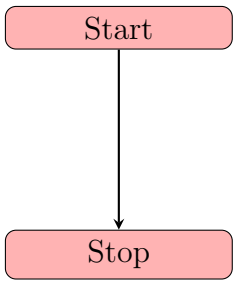

Abbildung 7.1.: Test figure

Teil IV.

Fazit und Erweiterungsmöglichkeiten

Teil V.

Anhang

A. Tabellen

Head_pin	$PINS	ADDR/OFFSET	GPIO NO.	Name	Mode7	Mode6	Mode5	Mode4	Mode3	Mode2	Mode1	Mode0	PIN	Notes
P8_01				DGND										Ground
P8_02				DGND										Ground
P8_03	6	0x818/018	38	GPIO1_6	gpio1[6]						mmc1_dat6	gpmc_ad6	R9	Used on Board (Group: pinmux_emmc2_pins)
P8_04	7	0x81c/01c	39	GPIO1_7	gpio1[7]						mmc1_dat7	gpmc_ad7	T9	Used on Board (Group: pinmux_emmc2_pins)
P8_05	2	0x808/008	34	GPIO1_2	gpio1[2]						mmc1_dat2	gpmc_ad2	R8	Used on Board (Group: pinmux_emmc2_pins)
P8_06	3	0x80c/00c	35	GPIO1_3	gpio1[3]						mmc1_dat3	gpmc_ad3	T8	Used on Board (Group: pinmux_emmc2_pins)
P8_07	36	0x890/090	66	TIMER4	gpio2[2]					timer4		gpmc_advn_ale	R7	
P8_08	37	0x894/094	67	TIMER7	gpio2[3]					timer7		gpmc_oen_ren	T7	
P8_09	39	0x89c/09c	69	TIMER5	gpio2[5]					timer5		gpmc_be0n_cle	T6	
P8_10	38	0x898/098	68	TIMER6	gpio2[4]					timer6		gpmc_wen	U6	
P8_11	13	0x834/034	45	GPIO1_13	gpio1[13]			eQEP2B_in	mmc2_dat1	mmc1_dat5	lcd_data18	gpmc_ad13	R12	
P8_12	12	0x830/030	44	GPIO1_12	gpio1[12]			EQEP2A_IN	MMC2_DAT0	MMC1_DAT4	LCD_DATA19	GPMC_AD12	T12	
P8_13	9	0x824/024	23	EHRPWM2B	gpio0[23]			ehrpwm2B	mmc2_dat5	mmc1_dat1	lcd_data22	gpmc_ad9	T10	
P8_14	10	0x828/028	26	GPIO0_26	gpio0[26]			ehrpwm2_tripzone_in	mmc2_dat6	mmc1_dat2	lcd_data21	gpmc_ad10	T11	
P8_15	15	0x83c/03c	47	GPIO1_15	gpio1[15]			eQEP2_strobe	mmc2_dat3	mmc1_dat7	lcd_data16	gpmc_ad15	U13	
P8_16	14	0x838/038	46	GPIO1_14	gpio1[14]			eQEP2_index	mmc2_dat2	mmc1_dat6	lcd_data17	gpmc_ad14	V13	
P8_17	11	0x82c/02c	27	GPIO0_27	gpio0[27]			ehrpwm0_synco	mmc2_dat7	mmc1_dat3	lcd_data20	gpmc_ad11	U12	
P8_18	35	0x88c/08c	65	GPIO2_1	gpio2[1]	mcasp0_fsr			mmc2_clk	gpmc_wait1	lcd_memory_clk	gpmc_clk_mux0	V12	
P8_19	8	0x820/020	22	EHRPWM2A	gpio0[22]			ehrpwm2A	mmc2_dat4	mmc1_dat0	lcd_data23	gpmc_ad8	U10	
P8_20	33	0x884/084	63	GPIO1_31	gpio1[31]					mmc1_cmd	gpmc_be1n	gpmc_csn2	V9	Used on Board (Group: pinmux_emmc2_pins)
P8_21	32	0x880/080	62	GPIO1_30	gpio1[30]					mmc1_clk	gpmc_clk	gpmc_csn1	U9	Used on Board (Group: pinmux_emmc2_pins)
P8_22	5	0x814/014	37	GPIO1_5	gpio1[5]						mmc1_dat5	gpmc_ad5	V8	Used on Board (Group: pinmux_emmc2_pins)
P8_23	4	0x810/010	36	GPIO1_4	gpio1[4]						mmc1_dat4	gpmc_ad4	U8	Used on Board (Group: pinmux_emmc2_pins)
P8_24	1	0x804/004	33	GPIO1_1	gpio1[1]						mmc1_dat1	gpmc_ad1	V7	Used on Board (Group: pinmux_emmc2_pins)
P8_25	0	0x800/000	32	GPIO1_0	gpio1[0]						mmc1_dat0	gpmc_ad0	U7	Used on Board (Group: pinmux_emmc2_pins)
P8_26	31	0x87c/07c	61	GPIO1_29	gpio1[29]							gpmc_csn0	V6	
P8_27	56	0x8e0/0e0	86	GPIO2_22	gpio2[22]						gpmc_a8	lcd_vsync	U5	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_28	58	0x8e8/0e8	88	GPIO2_24	gpio2[24]						gpmc_a10	lcd_pclk	V5	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_29	57	0x8e4/0e4	87	GPIO2_23	gpio2[23]						gpmc_a9	lcd_hsync	R5	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_30	59	0x8ec/0ec	89	GPIO2_25	gpio2[25]						gpmc_a11	lcd_ac_bias_en	R6	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_31	54	0x8d8/0d8	10	UART5_CTSN	gpio0[10]	uart5_ctsn		uart5_rxd	mcasp0_axr1	eQEP1_index	gpmc_a18	lcd_data14	V4	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_32	55	0x8dc/0dc	11	UART5_RTSN	gpio0[11]	uart5_rtsn		mcasp0_axr3	mcasp0_ahclkx	eQEP1_strobe	gpmc_a19	lcd_data15	T5	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_33	53	0x8d4/0d4	9	UART4_RTSN	gpio0[9]	uart4_rtsn		mcasp0_axr3	mcasp0_fsr	eQEP1B_in	gpmc_a17	lcd_data13	V3	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_34	51	0x8cc/0cc	81	UART3_RTSN	gpio2[17]	uart3_rtsn		mcasp0_axr2	mcasp0_ahclkr	ehrpwm1B	gpmc_a15	lcd_data11	U4	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_35	52	0x8d0/0d0	8	UART4_CTSN	gpio0[8]	uart4_ctsn		mcasp0_axr2	mcasp0_aclkr	eQEP1A_in	gpmc_a16	lcd_data12	V2	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_36	50	0x8c8/0c8	80	UART3_CTSN	gpio2[16]	uart3_ctsn			mcasp0_axr0	ehrpwm1A	gpmc_a14	lcd_data10	U3	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_37	48	0x8c0/0c0	78	UART5_TXD	gpio2[14]	uart2_ctsn		uart5_txd	mcasp0_aclkx	ehrpwm1_tripzone_in	gpmc_a12	lcd_data8	U1	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_38	49	0x8c4/0c4	79	UART5_RXD	gpio2[15]	uart2_rtsn		uart5_rxd	mcasp0_fsx	ehrpwm0_synco	gpmc_a13	lcd_data9	U2	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_39	46	0x8b8/0b8	76	GPIO2_12	gpio2[12]				eQEP2_index		gpmc_a6	lcd_data6	T3	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_40	47	0x8bc/0bc	77	GPIO2_13	gpio2[13]			pr1_edio_data_out7	eQEP2_strobe		gpmc_a7	lcd_data7	T4	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_41	44	0x8b0/0b0	74	GPIO2_10	gpio2[10]				eQEP2A_in		gpmc_a4	lcd_data4	T1	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_42	45	0x8b4/0b4	75	GPIO2_11	gpio2[11]				eQEP2B_in		gpmc_a5	lcd_data5	T2	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_43	42	0x8a8/0a8	72	GPIO2_8	gpio2[8]				ehrpwm2_tripzone_in		gpmc_a2	lcd_data2	R3	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_44	43	0x8ac/0ac	73	GPIO2_9	gpio2[9]				ehrpwm0_synco		gpmc_a3	lcd_data3	R4	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_45	40	0x8a0/0a0	70	GPIO2_6	gpio2[6]				ehrpwm2A		gpmc_a0	lcd_data0	R1	Allocated (Group: nxp_hdmi_bonelt_pins)
P8_46	41	0x8a4/0a4	71	GPIO2_7	gpio2[7]				ehrpwm2B		gpmc_a1	lcd_data1	R2	Allocated (Group: nxp_hdmi_bonelt_pins)
P9 Header	cat $PINS	ADDR +	GPIO NO.	Name	Mode 7	Mode 6	Mode 5	Mode 4	Mode 3	Mode 2	Mode 1	Mode 0	CPU	Updates Available at www.derekmolloy.ie
		44e10000	(Mode 7)										PIN	
		Offset from:												
		44e10800												

User LEDs				
USR0	21	0x854/054	53	GPIO1_21
USR1	22	0x858/058	86	GPIO2_22
USR2	23	0x85c/05c	87	GPIO2_23
US3	24	0x860/060	88	GPIO2_24

GPIO Settings				
Bit 6	Bit 5	Bit 4	Bit 3	Bit 2,1,0
Slew Control	Receiver Active	Pullup/Pulldown	Enable Pullup/down	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 pulldown, 0x17 pullup, 0x?f no pullup/down

e.g. INPUT GPIO(mode7) 0x27 pulldown, 0x37 pullup, 0x?f no pullup/down

by Derek Molly (www.derekmolloy.ie)

Tabelle A.1.: BeagleBone Black Expansion Header (P8)

Head_pin	$PINS	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_BUT											Has a 5V Level (pulled up by TPS65217C)
P9_10			SYS_RESETn									RESET_OUT	A10	
P9_11	28	0x870/070	UART4_RXD	30	gpio0[30]	uart4_rxd_mux2		mmc1_sdcd	rmii2_crs_dv	gpmc_csn4	mii2_crs	gpmc_wait0	T17	NB: GPIOs limit current to 4-6mA output
P9_12	30	0x878/078	GPIO1_28	60	gpio1[28]	mcasp0_aclkr_mux3		gpmc_dir	mmc2_dat3	gpmc_csn6	mii2_col	gpmc_be1n	U18	and approx. 8mA on input.
P9_13	29	0x874/074	UART4_TXD	31	gpio0[31]	uart4_txd_mux2		mmc2_sdcd	rmii2_rxerr	gpmc_csn5	mii2_rxerr	gpmc_wpn	U17	
P9_14	18	0x848/048	EHRPWM1A	50	gpio1[18]	ehrpwm1A_mux1		gpmc_a18	mmc2_dat1	rgmii2_td3	mii2_txd3	gpmc_a2	U14	
P9_15	16	0x840/040	GPIO1_16	48	gpio1[16]	ehrpwm1_tripzone_input		gpmc_a16	mii2_txen	rmii2_tctl	gmii2_txen	gpmc_a0	R13	
P9_16	19	0x84c/04c	EHRPWM1B	51	gpio1[19]	ehrpwm1B_mux1		gpmc_a19	mmc2_dat2	rgmii2_td2	mii2_txd2	gpmc_a3	T14	
P9_17	87	0x95c/15c	I2C1_SCL	5	gpio0[5]				ehrpwm0_synci	I2C1_SCL	mmc2_sdwp	spi0_cs0	A16	
P9_18	86	0x958/158	I2C1_SDA	4	gpio0[4]				ehrpwm0_tripzone	I2C1_SDA	mmc1_sdwp	spi0_d1	B16	
P9_19	95	0x97c/17c	I2C2_SCL	13	gpio0[13]			spi1_cs1	I2C2_SCL	dcan0_rx	timer5	uart1_rtsn	D17	Allocated (Group: pinmux_i2c2_pins)
P9_20	94	0x978/178	I2C2_SDA	12	gpio0[12]			spi1_cs0	I2C2_SDA	dcan0_tx	timer6	uart1_ctsn	D18	Allocated (Group: pinmux_i2c2_pins)
P9_21	85	0x954/154	UART2_TXD	3	gpio0[3]	EMU3_mux1			ehrpwm0B	I2C2_SCL	uart2_txd	spi0_d0	B17	
P9_22	84	0x950/150	UART2_RXD	2	gpio0[2]	EMU2_mux1			ehrpwm0A	I2C2_SDA	uart2_rxd	spi0_sclk	A17	
P9_23	17	0x844/044	GPIO1_17	49	gpio1[17]	ehrpwm0_synco		gpmc_a17	mmc2_dat0	rgmii2_rxdv	gmii2_rxdv	gpmc_a1	V14	
P9_24	97	0x984/184	UART1_TXD	15	gpio0[15]				I2C1_SCL	dcan1_rx	mmc2_sdwp	uart1_txd	D15	
P9_25	107	0x9ac/1ac	GPIO3_21	117	gpio3[21]			EMU4_mux2	mcasp1_axr1	mcasp0_axr3	eQEP0_strobe	mcasp0_ahclkx	A14	Allocated (Group: mcasp0_pins)
P9_26	96	0x980/180	UART1_RXD	14	gpio0[14]				I2C1_SDA	dcan1_tx	mmc1_sdwp	uart1_rxd	D16	
P9_27	105	0x9a4/1a4	GPIO3_19	115	gpio3[19]			EMU2_mux2	mcasp1_fsx	mcasp0_axr3	eQEP0B_in	mcasp0_fsr	C13	
P9_28	103	0x99c/19c	SPI1_CS0	113	gpio3[17]			eCAP2_in_PWM2_out	spi1_cs0	mcasp0_axr2	ehrpwm0_synci	mcasp0_ahclkr	C12	Allocated (Group: mcasp0_pins)
P9_29	101	0x994/194	SPI1_D0	111	gpio3[15]			mmc1_sdcd_mux1	spi1_d0		ehrpwm0B	mcasp0_fsx	B13	Allocated (Group: mcasp0_pins)
P9_30	102	0x998/198	SPI1_D1	112	gpio3[16]			mmc2_sdcd_mux1	spi1_d1		ehrpwm0_tripzone	mcasp0_axr0	D12	
P9_31	100	0x990/190	SPI1_SCLK	110	gpio3[14]			mmc0_sdcd_mux1	spi1_sclk		ehrpwm0A	mcasp0_aclkx	A13	Allocated (Group: mcasp0_pins)
P9_32			VADC											Voltage Reference for ADC (NB: 1.8V)
P9_33			AIN4										C8	NB: 1.8V tolerant
P9_34			AGND											Ground for ADC
P9_35			AIN6										A8	NB: 1.8V tolerant
P9_36			AIN5										B8	NB: 1.8V tolerant
P9_37			AIN2										B7	NB: 1.8V tolerant
P9_38			AIN3										A7	NB: 1.8V tolerant
P9_39			AIN0										B6	NB: 1.8V tolerant
P9_40			AIN1										C7	NB: 1.8V tolerant
P9_41A	109	0x9b4/1b4	CLKOUT2	20	gpio0[20]	EMU3_mux0		timer7_mux1	clkout2	tclkin		xdma_event_intr1	D14	Both signals are connected to P21 of P11
P9_41B		0x9a8/1a8	GPIO3_20	116	gpio3[20]			emu3	Mcasp1_axr0		eQEP0_index	mcasp0_axr1	D13	Both signals are connected to P21 of P11
P9_42A	89	0x964/164	GPIO0_7	7	gpio0[7]	xdma_event_intr2	mmc0_sdwp	spi1_sclk	pr1_ecap0_ecap_capin_apwm_o	spi1_cs1	uart3_txd	eCAP0_in_PWM0_out	C18	Both signals are connected to P22 of P11
P9_42B		0x9a0/1a0	GPIO3_18	114	gpio3[18]				Mcasp1_aclkx	Mcaspo_axr2	eQEP0A_in	Mcasp0_aclkr	B12	Allocated (Group:mcasp0_pins)
P9_43			GND											- See Pg.50 of the SRM
P9_44			GND											Ground
P9_45			GND											Ground
P9_46			GND											Ground
														For updates see: www.derekmolloy.ie
P9 Header	cat $PINS	ADDR +	Name	GPIO NO.	Mode 7						Mode 1	Mode 0	CPU	Notes
	Allocated	44e10000		(Mode 7)										Please e-mail me directly at:
		Offset from:												derek@derekmolloy.ie
		44e10800												if you notice a mistake

GPIO Settings				
Bit 6	Bit 5	Bit 4	Bit 3	Bit 2,1,0
Slew Control	Receiver Active	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 pulldown, 0x17 pullup, 0x?f no pullup/down

e.g. INPUT GPIO(mode7) 0x27 pulldown, 0x37 pullup, 0x?f no pullup/down

by Derek Molly (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