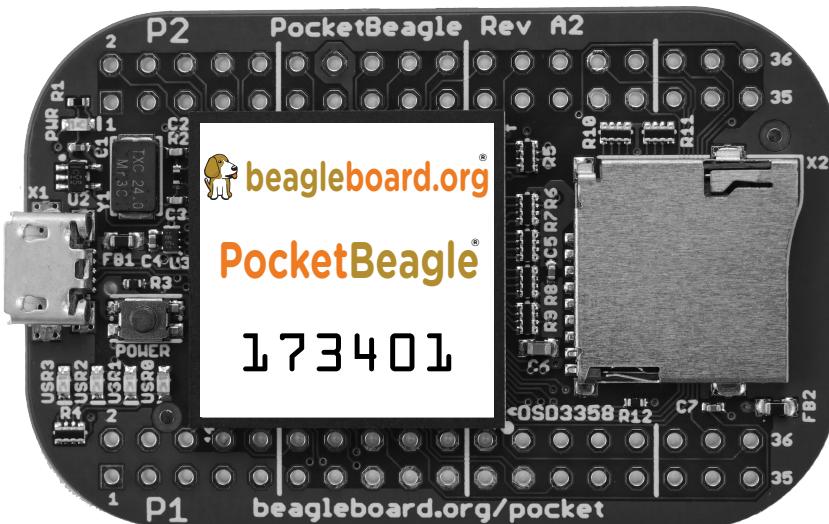


PocketBeagle®, the tiniest \$25 key-fob computer you can buy

The newest BeagleBoard.org® board is PocketBeagle®, an ultra-tiny-yet-complete Linux-enabled, community-supported, open-source USB-key-fob computer. PocketBeagle® features an incredible low cost, slick design and simple usage, making PocketBeagle® the ideal development board for beginners and professionals alike. You develop directly in a web browser and PocketBeagle® can easily be set back to factory conditions, leaving you free to experiment.

Key features:

- * Low cost Linux computer with tremendous expansibility
 - * Opportunity to learn many programming aspects from educators on-line
 - * Openness and flexibility tear-down limits on your imagination



The whats and whys of PocketBeagle®

What is a USB key-fob computer?

PocketBeagle® is the size of a tiny mint-tin (35mm by 55mm), less than half the size of the larger mint-tin or credit-card sized BeagleBone® Black (55mm by 86mm). Unlike a desktop computer where you connect a monitor, keyboard and mouse, PocketBeagle® is made to live inside your project and enables you to define its interfaces.

PocketBeagle® is easily programmed through a web browser running on any other connected desktop.

What can I do with PocketBeagle®?

Getting to the Linux command-line and text editor via your web browser is simple, providing you with a playground for programming and electronics. Exploring is made easy with several available libraries and tutorials and many more coming. Once you get a bit familiar with Linux and electronics, you are free to explore numerous more advanced projects from the community.

The sky is no limit; PocketBeagle® makes a great starting point for building something as advanced as a computer for a CubeSat and there are several BeagleBone® Black examples out there already today. Flying a bit lower, PocketBeagle® is a good target for flight controller software, such as ArduPilot, similar to what is done on BeagleBone® Blue but with the flexibility of choosing all your own sensors and interconnects. Touching the ground, the combination of a 1-GHz Linux computer and 2 powerful 200-MHz hard-real-time shared-memory programmable real-time (PRU) microcontrollers makes driving robotic machines like 3D printers, CNC mills and laser cutters fast and simple with software such as Redeem, MachineKit or BeagleG as great starting points. If you'd like your ground-based machine to talk back to the cloud, the SPI, USB and UART expansion makes adding your own Ethernet, WiFi, Bluetooth and long-range wireless connectivity easy with Linux drivers and Node.JS or Python libraries to add smarts. If what you want is just fun, add a SPI-based display and run off of a single-cell LiPo battery to create your own custom gaming device with the sensors, such as cameras and software like OpenCV or just simple accelerometers, of your own choice to go on an adventure of your own.

What are Programmable Real-Time Units (PRUs) anyway?

Texas Instruments' Sitara AM335x processor is built of several central processing units (CPUs), including two programmable real-time units (PRUs). These are 200MHz 32-bit microcontroller CPUs with a zero-depth pipeline and single-cycle access to a collection of pins and peripherals optimized for implementing software-defined peripherals. The PRUs don't have pre-defined tasks in they system, so they are free to run your software. They are ideal for tight control loops; a must-have feature for building quadcopters, 3D printers and balancing robots, just to name a few.

PRUs are ideal for predictable low-latency, whereas the ARM processor is good for throughput. Latency is how quickly you can respond. Throughput is how quickly you can move once you reach top speed. When you need to handle a lot of little tasks, you need low latency and low overhead. When you have a large payload to process, you want good throughput. PocketBeagle® is designed to be good at both and efficient in enabling them to work together.

Why is PocketBeagle® good for a novice?

PocketBeagle® is affordable, enabling you to dedicate one to live permanently in each of your different projects. Even though PocketBeagle® is very low-cost, it is made with top quality engineering and manufacturing. In the unlikely event it is damaged due to misuse, it won't cost much to replace.

PocketBeagle® will boot directly from on-board ROM that cannot be accidentally modified and will load software via USB, serial or microSD cards. A Chrome plug-in or cross-platform Node.JS Electron app can boot your board to add a Linux distribution to an attached microSD card. This means you can create reliably reproducible instructions on using the board, because it will behave the same way every time.

PocketBeagle® runs GNU/Linux, which means you can leverage many different high-level programming languages and a large body of drivers that prevent you from needing to write a lot of your own software.

PocketBeagle® doesn't require you to install a bunch of tools on your host computer. You develop directly in your web browser, a tool you already know, and most users won't even need administrator privileges on their host computer.

Why is PocketBeagle® good for a professional?

PocketBeagle® utilizes a simple open-source hardware design making it easy for you to take control of your own destiny, either customizing the board or sourcing and manufacturing it in any method you'd like. The processor is well-documented and available broadly, avoiding lurking issues that might otherwise be difficult to resolve.

PocketBeagle® avoids consuming precious hardware resources, leaving them available for your design goals while still exposing features for expansion.

Most importantly, PocketBeagle® respects your time and helps you avoid risk by providing a solution for rapid prototyping without slowing your path to production. BeagleBoard.org's community of thousands of serious developers means that someone has probably already tried to do something similar and will be willing to share answers and experiences. PocketBeagle® avoids re-inventing the wheel, just makes it smaller, simpler and more affordable while keeping it flexible.

PocketBeagle® support is pushed upstream in Linux and u-boot such that you'll always be able to leverage the latest features and bug fixes in those projects.

Why does PocketBeagle® really matter anyway?

The BeagleBoard.org Foundation has the goal of increasing computer, electronics and robotics literacy at a point they are critical in human history. PocketBeagle® increases access to inspirational hardware and we believe hands-on experience is the best way to build understanding and empowerment. We also believe in empowerment at all ages and experience, so we give you the tools to take our designs forward.

How did we make PocketBeagle® so small and affordable?

PocketBeagle® is built around Octavo Systems' System-In-Package that integrates a high-performance TI AM3358 processor, 512MB of DDR3, power management, non-volatile serial memory and over 140 passive components into a single package. This integration saves cost and a small amount of power by eliminating several packages that would otherwise need to be placed on the board, but more notably it saves the space of all those packages and simplifies our board design so we can focus on the user experience.

Specifications:

- a. Texas Instruments® Sitara™ AM3358 Processor (Integrated in the OSD3358-SM):
 - i. 1GHz ARM® Cortex-A8 with NEON floating-point accelerator
 - ii. SGX530 graphics accelerator
 - iii. 2x programmable real-time unit (PRU) 32-bit 200MHz microcontrollers with single-cycle I/O latency
 - iv. ARM® Cortex-M3 for power and security management functions
- b. Memory:
 - i. 512MB DDR3 800MHZ RAM (Integrated in the OSD3358-SM)
 - ii. 4kB I2C EEPROM (Integrated in the OSD3358-SM)
 - iii. SD/MMC Connector for microSD
- c. Software Compatibility
 - i. Debian GNU/Linux images customized for BeagleBone
 - ii. Cloud9 IDE on Node.js w/ BoneScript library
 - iii. Any BeagleBone Black software not needing access to unavailable expansion pins
- d. Connectivity
 - i. High speed USB 2.0 OTG (host/client) micro-B connector (USB0)
 - ii. Bootable microSD card slot (MMC0)
- e. Expansion header
 - i. High speed USB 2.0 OTG (host/client) control signals (USB1)
 - ii. 8 analog inputs with 6 at 1.8V and 2 at 3.3V along with 1.8V voltage references
 - iii. 44 digital GPIOs accessible with 18 enabled by default including 2 shared with the 3.3V analog input pins
 - iv. 3 UARTs accessible with 2 enabled by default (UART0, UART4)
 - v. 2 I2C busses enabled by default (I2C1, I2C2)
 - vi. 2 SPI busses with single chip selects enabled by default (SPI0, SPI1)
 - vii. 4 PWM outputs accessible with 2 enabled by default (PWM0A, PWM1A)
 - viii. 2 quadrature encoder inputs accessible
 - ix. 2 CAN bus controllers accessible
 - x. 23 programmable real-time unit (PRU) 32-bit microcontroller I/O pins including options for the PRU UART and eCAP accessible with 7 I/O pins enabled by default for PRU0 and 1 enabled by default for PRU1

xi. 3 voltage inputs with 1 for battery, 1 shared with the USB connector and 1 for power-line input and a battery temperature sense input

xii. 2 voltage outputs, 1 with a 3.3V LDO and 1 with switch from voltage input

xiii. Power and reset button I/Os

f. Power management:

i. TPS65217C PMIC is used along with a separate LDO to provide power to the system (Integrated in the OSD3358) with integrated 1-cell LiPo battery support

g. Debug Support:

i. JTAG test points

ii. gdb and other monitor-mode debug possible

h. Power Source

i. microUSB connector

ii. expansion header (3 options: battery, VIN or USB-VIN)

i. User Input / Output

i. Power Button with press detection interrupt via TPS65217C PMIC (hold for 10s to initiate hardware power cycle)

PocketBeagle Expansion Headers

P1										P2									
USB1_V_EN	GPIO	109	3	4	89	SYS	VIN	1	2	87	6	AIN 3.3V	9	PRU1	PWM1_A	50	1	2	59
VBUS	5	6	5	6	2	GPIO	7	8	9	11	CS	TX	PRU	PWM2_B	23	3	4	58	
VIN	7	8	2	7	8	GPIO	9	10	3	10	CLK	RX	UART2	UART4_RX	30	5	6	57	
DN	9	10	3	9	10	GPIO	11	12	4	12	MISO	SP10	TX	TX	UART4_TX	31	7	8	60
ID	13	14	3.3V	13	14	GPIO	15	16	GND	16	MOSI	RX	PRU	CAN1_RX	15	9	10	52	
GND	15	16	GND	15	16	GPIO	17	18	REF+	18	REF+	AIN 1.8V	16(in)	PRU0	CAN1_TX	14	11	12	PWR_BTN
AIN 1.8V	0	19	20	19	20	GPIO	21	22	GND	21	VOUT	16(in)	PRU0	SDA	14	11	12	VIN_BTN	
	1	21	22	21	22	GPIO	23	24	VOUT	23	16(in)	PRU0	16(out)	TEMP	15	16	17	18	
	2	23	24	23	24	GPIO	25	26	SDA	25	16(in)	PRU0	16(out)	STRB	19	20	64	47	
	3	25	26	25	26	GPIO	27	28	I2C2_SDA	27	16(in)	PRU0	16(out)	QEP2	21	22	46	47	
	4	27	28	27	28	GPIO	29	30	I2C2_SCL	29	16(in)	PRU0	16(out)	15i	29	30	48	48	
	5	29	30	29	30	GPIO	31	32	I2C2_RX	31	16(in)	PRU0	16(out)	PRU0	31	32	112	113	
	6	31	32	31	32	GPIO	33	34	UART0_RX	33	16(in)	PRU1	16(out)	NRST	32	33	34	113	
	7	32	33	32	33	GPIO	34	35	UART0_TX	34	16(in)	PRU1	16(out)	SYS	32	33	34	113	
	8	33	34	33	34	GPIO	35	36	PRU0_PWM0_A	35	16(in)	PRU1	16(out)	IDX	31	32	33	113	
	9	34	35	34	35	GPIO	36	37	PRU0_PWM0_B	36	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	10	35	36	35	36	GPIO	37	38	PRU0_PWM0_B	37	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	11	36	37	36	37	GPIO	38	39	PRU0_PWM0_B	38	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	12	37	38	37	38	GPIO	39	40	PRU0_PWM0_B	39	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	13	38	39	38	39	GPIO	40	41	PRU0_PWM0_B	40	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	14	39	40	39	40	GPIO	41	42	PRU0_PWM0_B	41	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	15	40	41	40	41	GPIO	42	43	PRU0_PWM0_B	42	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	16	41	42	41	42	GPIO	43	44	PRU0_PWM0_B	43	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	17	42	43	42	43	GPIO	44	45	PRU0_PWM0_B	44	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	18	43	44	43	44	GPIO	45	46	PRU0_PWM0_B	45	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	19	44	45	44	45	GPIO	46	47	PRU0_PWM0_B	46	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	20	45	46	45	46	GPIO	47	48	PRU0_PWM0_B	47	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	21	46	47	46	47	GPIO	48	49	PRU0_PWM0_B	48	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	22	47	48	47	48	GPIO	49	50	PRU0_PWM0_B	49	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	23	48	49	48	49	GPIO	50	51	PRU0_PWM0_B	50	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	24	49	50	49	50	GPIO	51	52	PRU0_PWM0_B	51	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	25	50	51	50	51	GPIO	52	53	PRU0_PWM0_B	52	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	26	51	52	51	52	GPIO	53	54	PRU0_PWM0_B	53	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	27	52	53	52	53	GPIO	54	55	PRU0_PWM0_B	54	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	28	53	54	53	54	GPIO	55	56	PRU0_PWM0_B	55	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	29	54	55	54	55	GPIO	56	57	PRU0_PWM0_B	56	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	30	55	56	55	56	GPIO	57	58	PRU0_PWM0_B	57	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	31	56	57	56	57	GPIO	58	59	PRU0_PWM0_B	58	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	32	57	58	57	58	GPIO	59	60	PRU0_PWM0_B	59	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	33	58	59	58	59	GPIO	60	61	PRU0_PWM0_B	60	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	34	59	60	59	60	GPIO	61	62	PRU0_PWM0_B	61	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	35	60	61	60	61	GPIO	62	63	PRU0_PWM0_B	62	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	36	61	62	61	62	GPIO	63	64	PRU0_PWM0_B	63	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	37	62	63	62	63	GPIO	64	65	PRU0_PWM0_B	64	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	38	63	64	63	64	GPIO	65	66	PRU0_PWM0_B	65	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	39	64	65	64	65	GPIO	66	67	PRU0_PWM0_B	66	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	40	65	66	65	66	GPIO	67	68	PRU0_PWM0_B	67	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	41	66	67	66	67	GPIO	68	69	PRU0_PWM0_B	68	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	42	67	68	67	68	GPIO	69	70	PRU0_PWM0_B	69	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	43	68	69	68	69	GPIO	70	71	PRU0_PWM0_B	70	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	44	69	70	69	70	GPIO	71	72	PRU0_PWM0_B	71	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	45	70	71	70	71	GPIO	72	73	PRU0_PWM0_B	72	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	46	71	72	71	72	GPIO	73	74	PRU0_PWM0_B	73	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	47	72	73	72	73	GPIO	74	75	PRU0_PWM0_B	74	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	48	73	74	73	74	GPIO	75	76	PRU0_PWM0_B	75	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	49	74	75	74	75	GPIO	76	77	PRU0_PWM0_B	76	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	50	75	76	75	76	GPIO	77	78	PRU0_PWM0_B	77	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	51	76	77	76	77	GPIO	78	79	PRU0_PWM0_B	78	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	52	77	78	77	78	GPIO	79	80	PRU0_PWM0_B	79	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	53	78	79	78	79	GPIO	80	81	PRU0_PWM0_B	80	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	54	79	80	79	80	GPIO	81	82	PRU0_PWM0_B	81	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	55	80	81	80	81	GPIO	82	83	PRU0_PWM0_B	82	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	56	81	82	81	82	GPIO	83	84	PRU0_PWM0_B	83	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	57	82	83	82	83	GPIO	84	85	PRU0_PWM0_B	84	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	58	83	84	83	84	GPIO	85	86	PRU0_PWM0_B	85	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	59	84	85	84	85	GPIO	86	87	PRU0_PWM0_B	86	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	60	85	86	85	86	GPIO	87	88	PRU0_PWM0_B	87	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	61	86	87	86	87	GPIO	88	89	PRU0_PWM0_B	88	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	62	87	88	87	88	GPIO	89	90	PRU0_PWM0_B	89	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	63	88	89	88	89	GPIO	90	91	PRU0_PWM0_B	90	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	64	89	90	89	90	GPIO	91	92	PRU0_PWM0_B	91	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	65	90	91	90	91	GPIO	92	93	PRU0_PWM0_B	92	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	66	91	92	91	92	GPIO	93	94	PRU0_PWM0_B	93	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	67	92	93	92	93	GPIO	94	95	PRU0_PWM0_B	94	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	68	93	94	93	94	GPIO	95	96	PRU0_PWM0_B	95	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	69	94	95	94	95	GPIO	96	97	PRU0_PWM0_B	96	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	70	95	96	95	96	GPIO	97	98	PRU0_PWM0_B	97	16(in)	PRU1	16(out)	QEP0	32	33	34	113	
	71	96	97	96	97	GPIO	98	99	PRU0_PWM										