TECHNOLOGY IN ACTION™

Raspberry Pi Hardware Reference

COMPLETE COVERAGE OF THE RASPBERRY PI'S HARDWARE

Warren Gay

For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.

Apress®

Contents at a Glance

Introduction

After receiving your first Raspberry Pi, the first question in your mind is probably "What can this hardware do?" What are its capabilities and limitations? Hardware is the more urgent question because software is so easily altered or replaced.

The one perplexing problem I immediately ran up against when I started out with the Pi was that the hardware information seemed to be scattered. The basic information was accessible and well known, but other important parameters such as GPIO source or sink current limits required research. After researching these questions, I often discovered that the answer was "It depends." It was the answering of these classes of questions that led to the writing of *Mastering the Raspberry Pi*.

Content of This Book

This book is focused mainly on the Raspberry Pi's hardware. The content is extracted from the complete work, *Mastering the Raspberry Pi*. As such, it will serve you as an owner's manual of sorts, saving time as a ready reference about the hardware you purchased.

While this is a volume focused on hardware, some software coverage must coexist. For example, it is through the physical memory management that software gains access to the hardware peripheral registers. Another example is the discussion about the CPU, where the pthread API is covered for reference purposes. Through the application of this API, you further utilize that ARM CPU.

This book begins by introducing the Pi in general terms in Chapter 1. Then attention immediately turns to the important topic of power in Chapter 2. Many people suffer needless problems because of neglect in this area. The chapter ends with some notes about running from battery or solar power.

Chapter 3 documents the header strips, LEDs, and Reset inputs. This is information that should be bookmarked. Next is Chapter 4 on memory, which documents the various Raspbian Linux measures and controls for memory allocation. The CPU and its API are described in Chapter 5.

The focus of Chapter 6 is USB. USB-specific power issues and its API are explained. Wired and wireless Ethernet networking is discussed in Chapter 7. SD card technology is examined in Chapter 8, describing the interface and the specifics of the Raspberry Pi interface. The topic of wear leveling is also included.

Serial communication, RS-232 converters, serial consoles, and dedicated serial ports are covered in Chapter 9. The serial interface, some historical influences, and flow control are discussed. Included is an organized description of the Linux API for utilizing the serial interface.

Chapter 10 covers the important area of the GPIO interface. Every aspect of GPIO is covered, including its configuration after reset and boot. Logic levels, drive strength, input pullup resistor control, and output totem pole configuration are explained. Each is examined from an electronics viewpoint. Additionally, the various ways of applying these GPIO pins in software are described.

The GPIO coverage also includes guidance about how to budget the +3.3 V supply current. Configuration of the pins and selection of alternate I/O functions are also discussed. Finally, a design procedure is provided for a single transistor driver, when more power is required.

The next three chapters concern themselves with Raspbian Linux–supported peripheral buses. The one-wire driver is supported through a Linux driver and described in Chapter 11. The I2C bus is another important peripheral bus, which is documented with its API in Chapter 12. Finally, the SPI bus is explained with its API in Chapter 13. With this coverage, you will be fully informed of what is available and how to leverage Raspbian Linux to drive it.

Assumptions About You

Apart from the C language software API presented in this book, much of the content of this volume is electronics based. You should therefore have a basic understanding of digital electronics. This includes a good grasp of DC voltage, current, resistance, power, and mastery of Ohm's law (you may also refer to Appendix C). For a full appreciation of the concepts behind the I2C bus, you should also be familiar with the operation of an open collector driver.

The transistor driver design procedure provided in Chapter 10 (GPIO) uses a *light* engineering approach where formulas are *assumed* (an engineering text would also include the *derivation* of the formulas). The intent here is to simply demonstrate that the use of design procedure can solve problems that might otherwise cause students to look for a *chip* solution when a *transistor* would suffice. Let's take the fear out of design.

Learn and Design

The main assumption throughout this book is that you are looking to learn how to design things for yourself. Through an appreciation of the involved hardware parameters, design procedure, and the software API, you will be able to build custom solutions using the Raspberry Pi. To further assist in this, several charts and tables were provided in this reference. Any real designer takes delight in having the necessary parameters available at their disposal.

CHAPTER 1

The Raspberry Pi

Before considering the details about each resource within the Raspberry Pi, it is useful to take a high-level inventory. In this chapter, let's just list what you get when you purchase a Pi.

In later chapters, you'll be looking at each resource from two perspectives:

- The hardware itself—what it is and how it works
- The driving software and API behind it

In some cases, the hardware will have one or more kernel modules behind it, forming the device driver layer. They expose a software API that interfaces between the application and the hardware device. For example, applications communicate with the driver by using `ioctl(2)` calls, while the driver communicates with the I2C devices on the bus. The `/sys/class` file system is another way that device drivers expose themselves to applications. You'll see this when you examine GPIO in Chapter 10.

There are some cases where drivers don't currently exist in Raspbian Linux. An example is the Pi's PWM peripheral that is covered in Chapter 9 of *Experimenting with Raspberry Pi* (Apress, 2014). Here we must map the device's registers into the application memory space and drive the peripheral directly from the application. Both direct access and driver access have their advantages and disadvantages.

So while our summary inventory here simply lists the hardware devices, you'll be examining each from a hardware and software point of view in the chapters ahead.

Models

A hardware inventory is directly affected by the model of the unit being examined. The Raspberry Pi comes in two models:

- Model A (introduced later as a hardware-reduced model)
- Model B (introduced first and is the full hardware model)

Figure 1-1 shows the Model B and its interfaces. Table 1-1 indicates the differences between the two models.

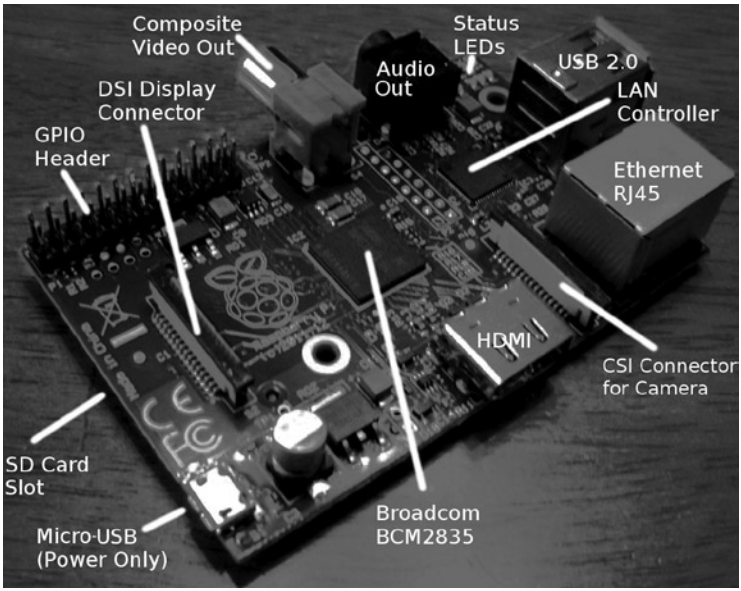

Figure 1-1. *Model B interfaces*

Table 1-1. *Model Differences*

Resource	Model A	Model B
RAM	256 MB	512 MB
USB ports	1	2
Ethernet port	None	10/100 Ethernet (RJ45)
Power consumption[10]	300 mA (1.5 W)	700 mA (3.5 W)
Target price[9]	$25.00	$35.00

As you can see, one of the first differences to note is the amount of RAM available. The revision 2.0 (Rev 2.0) Model B has 512 MB of RAM instead of 256 MB. The GPU also shares use of the RAM. So keep that in mind when budgeting RAM.

In addition, the Model A does not include an Ethernet port but can support networking through a USB network adapter. Keep in mind that only one USB port exists on the Model A, requiring a hub if other USB devices are needed.

Finally, the power consumption differs considerably between the two models. The Model A is listed as requiring 300 mA vs. 700 mA for the Model B. Both of these figures should be considered low because consumption rises considerably when the GPU is active (when using the desktop through the HDMI display port).

The maximum current flow that is permitted through the 5 V micro-USB connection is about 1.1 A because of the fuse. However, when purchasing a power supply/adapter, it is recommended that you seek supplies that are rated higher than 1.2 A because they often don't live up to their specifications. Chapter 2 provides more details about power supplies.

Hardware in Common

The two Raspberry Pi models share some common features, which are summarized in Table 1-2.[9] The Hardware column lists the broad categories; the Features column provides additional specifics.

Table 1-2. *Common Hardware Features*

Hardware	Features	Comments
System on a chip	Broadcom BCM2835	CPU, GPU, DSP, SDRAM, and USB port
CPU model Clock rate	ARM1176JZF-S core	With floating point
	700 MHz	Overclockable to 800 MHz
GPU	Broadcom VideoCore IV	
	OpenGL ES 2.0	3D
	OpenVG	3D
	MPEG-2	
	VC-1	Microsoft, licensed
	1080p30 H.264	Blu-ray Disc capable, 40 Mbit/s
	MPEG-4	AVC high-profile decoder and encoder
	1 Gpixel/s, 1.5 Gtexels/s	24 GFLOPS with DMA
Video output	Composite RCA	PAL and NTSC
	HDMI	Rev 1.3 and 1.4
	Raw LCD panels	Via DSI
Audio output	3.5 mm jack HDMI	

(*continued*)

Table 1-2. *(continued)*

Hardware	Features	Comments
Storage	SD/MMC/SDIO	Card slot
Peripherals	8 × GPIO	
	UART	
	I2C bus	100 kHz
	SPI bus	Two chip selects, +3.3 V, +5 V, ground
Power source	5 V via micro-USB	

Which Model?

One of the questions that naturally follows a model feature comparison is why the Model A? Why wouldn't everyone just buy Model B?

Power consumption is one deciding factor. If your application is battery powered, perhaps a data-gathering node in a remote location, then power consumption becomes a critical factor. If the unit is supplemented by solar power, the Model A's power requirements are more easily satisfied.

Cost is another advantage. When an Arduino/AVR class of application is being considered, the added capability of the Pi running Linux, complete with a file system on SD, makes it irresistible. Especially at the model A price of $25.

Unit cost may be critical to students in developing countries. Networking can be sacrificed, if it still permits the student to learn on the cheaper Model A. If network capability is needed later, even temporarily, a USB network adapter can be attached or borrowed.

The main advantage of the Model B is its networking capability. Networking today is so often taken for granted. Yet it remains a powerful way to integrate a larger system of components. The project outlined in Chapter 8 of *Experimenting with Raspberry Pi* (Apress, 2014) demonstrates how powerful ØMQ (ZeroMQ) can be in bringing separate nodes together.

CHAPTER 2

Power

One of the most frequently neglected parts of a system tends to be the power supply—at least when everything is working. Only when things get weird does the power supply begin to get some scrutiny.

The Raspberry Pi owner needs to give the power supply extra respect. Unlike many AVR class boards, where the raw input voltage is followed by an onboard 5 V regulator, the Pi expects its power to be regulated at the input. The Pi does include onboard regulators, but these regulate to lower voltages (3.3 V and lower).

Figure 2-1 illustrates the rather fragile Micro-USB power input connector. There is a large round capacitor directly behind the connector that people often grab for leverage. It is a mistake to grab it, however, as many have reported "popping it off" by accident.

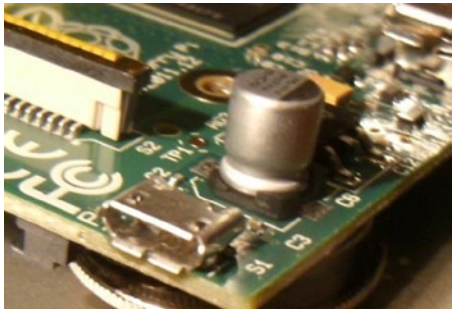

Figure 2-1. *Micro-USB power input*

Calculating Power

Sometimes power supplies are specified in terms of voltage, and power handling capability in watts. The Pi's input voltage of 5 V must support a *minimum* of 700 mA (Model B). Let's compute a power supply figure in watts (this does not include any added peripherals):

$$\begin{aligned} P &= V \times I \\ &= 5 \times 0.7 \\ &= 3.5\ W \end{aligned}$$

The 3.5 W represents a minimum requirement, so we should overprovision this by an additional 50%:

$$P = 3.5 \times 1.50$$
$$= 5.25 \text{ W}$$

The additional 50% yields a power requirement of 5.25 W.

■ **Tip** Allow 50% extra capacity for your power supply. A power supply gone bad may cause damage or many other problems. One common power-related problem for the Pi is loss of data on the SD card.

Current Requirement

Since the power supply being sought produces one output voltage (5 V), you'll likely see adapters with advertised *current* ratings instead of power. In this case, you can simply factor a 50% additional current instead:

$$I_{supply} = I_{Pi} \times 1.50$$
$$= 0.700 \times 1.50$$
$$= 1.05 \text{ A}$$

To double-check our work, let's see whether this agrees with the power rating we computed earlier:

$$P = V \times I$$
$$= 5 \times 1.05$$
$$= 5.25 \text{ W}$$

The result does agree. You can conclude this section knowing that you *minimally* need a 5 V supply that produces one of the following:

- 5.25 W or more
- 1.05 A or more (ignoring peripherals)

Supplies that can meet either requirement, should be sufficient. However, you should be aware that not all advertised ratings are what they seem. Cheap supplies often fail to meet their own claims, so an additional margin must always be factored in.

Peripheral Power

Each additional circuit that draws power, especially USB peripherals, must be considered in a power budget. Depending on its type, a given USB peripheral plugged into a USB 2 port can expect up to 500 mA of current, assuming it can obtain it. (Pre Rev 2.0 USB ports were limited to 140 mA by polyfuses.)

Wireless adapters are known to be power hungry. Don't forget about the keyboard and mouse when used, since they also add to the power consumption. If you've attached an RS-232 level shifter circuit (perhaps using MAX232CPE), you should budget for that small amount also in the 3 V supply budget. This will indirectly add to your +5 V budget, since the 3 V regulator is powered from it. (The USB ports use the +5 V supply.) Anything that draws power from your Raspberry Pi should be tallied.

Model B Input Power

The Raspberry Pi's input voltage is fixed at exactly 5 V (±0.25 V). Looking at the schematic in Figure 2-2, you can see how the power enters the micro-USB port on the pin marked VBUS. Notice that the power flows through fuse F3, which is rated at 6 V, 1.1 A. If after an accidental short, you find that you can't get the unit to power up, check that fuse with an ohmmeter.

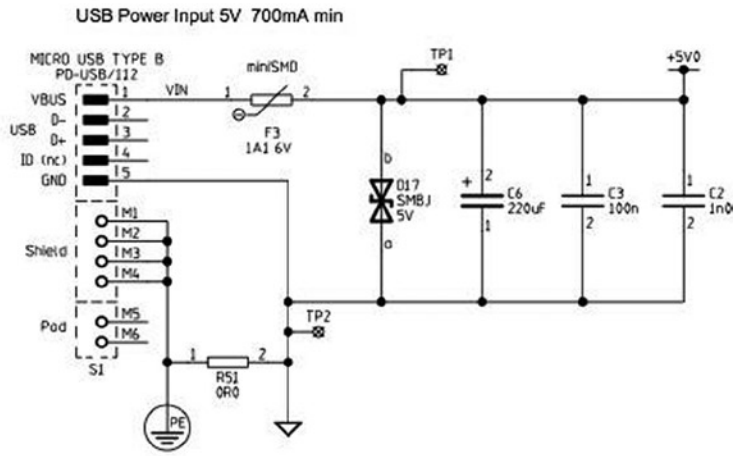

Figure 2-2. *Model B Rev 2.0 input power*

If you bring the input +5 V power into the Pi through header P1, P5, or TP1, for example, you will lose the safety of the fuse F3. So if you bypass the micro-USB port to bring in power, you may want to include a safety fuse in the supplying circuit.

Figure 2-3 shows the 3.3 V regulator for the Pi. Everything at the 3.3 V level is supplied by this regulator, and the current is limited by it.

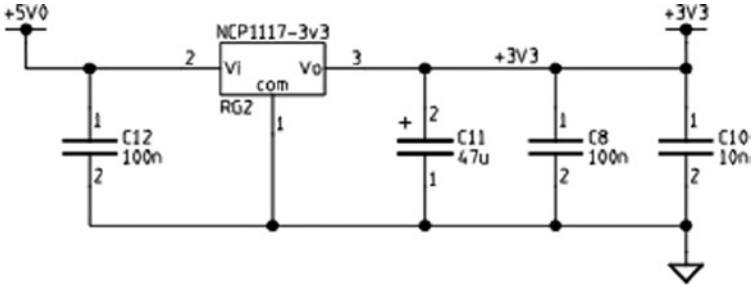

Figure 2-3. *3.3 V power*

Model A Input Power

Like the Model B, the Model A receives its power from the micro-USB port. The Model A power requirement is 300 mA, which is easily supported by a powered USB hub or desktop USB 2 port. A USB 2 port is typically able to supply a maximum of 500 mA unless the power is divided among neighboring ports. You may find in practice, however, that not all USB ports will deliver 500 mA.

As with the Model B, factor the power required by your USB peripherals. If your total nears or exceeds 500 mA, you may need to power your Model A from a separate power source. Don't try to run a wireless USB adapter from the Model A's USB port if the Pi is powered by a USB port itself. The total current needed by the Pi and wireless adapter will likely exceed 500 mA. Supply the wireless adapter power from a USB hub, or power the Pi from a 1.2 A or better power source. Also be aware that not all USB hubs function correctly under Linux, so check compatibility if you're buying one for that purpose.

3.3 Volt Power

Since the 3.3 V supply appears at P1-01, P1-17, and P5-02, it is useful to examine Figure 2-3 (shown previously) to note its source. This supply is indirectly derived from the input 5 V supply, passing through regulator RG2. The maximum excess current that can be drawn from it is 50 mA; the Raspberry Pi uses up the remaining capacity of this regulator.

When planning a design, you need to budget this 3 V supply carefully. Each GPIO output pin draws from this power source an additional 3 to 16 mA, depending on how it is used. For more information about this, see Chapter 10.

Powered USB Hubs

If your power budget is stretched by USB peripherals, you may want to consider the use of a *powered* USB hub. In this way, the *hub* rather than your Raspberry Pi provides the necessary power to the downstream peripherals. The hub is especially attractive for the Model A because it provides additional ports.

Again, take into account that not all USB hubs work with (Raspbian) Linux. The kernel needs to cooperate with connected USB hubs, so software support is critical. The following web page lists known working USB hubs:

http://elinux.org/RPi_Powered_USB_Hubs

Power Adapters

This section pertains mostly to the Model B because the Model A is easily supported by a USB 2 port. We'll first look at an unsuitable source of power and consider the factors for finding suitable units.

An Unsuitable Supply

The example shown in Figure 2-4 was purchased on eBay for $1.18 with free shipping (see the upcoming warning about fakes). For this reason, it was tempting to use it.

Figure 2-4. *Model A1265 Apple adapter*

This is an adapter/charger with the following ratings:

- *Model*: A1265
- *Input*: 100–240 VAC
- *Output*: 5 V, 1 A

When plugged in, the Raspberry Pi's power LED immediately lights up, which is a good sign for an adapter (vs. a charger). A fast rise time on the power leads to successful power-on resets. When the voltage was measured, the reading was +4.88 V on the +5 V supply. While not ideal, it is within the range of acceptable voltages. (The voltage must be between 4.75 and 5.25 V.)

The Apple unit seemed to work fairly well when HDMI graphics were *not* being utilized (using serial console, SSH, or VNC). However, I found that when HDMI was used and the GPU had work to do (move a window across the desktop, for example), the system would tend to seize up. This clearly indicates that the adapter does not fully deliver or regulate well enough.

■ **Caution** Be very careful of counterfeit Apple chargers/adapters. The Raspberry Pi Foundation has seen returned units damaged by these. For a video and further information, see www.raspberrypi.org/archives/2151.

E-book Adapters

Some people have reported good success using e-book power adapters. I have also successfully used a 2 A Kobo charger.

Best Power Source

While it is possible to buy USB power adapters at low prices, it is wiser to spend more on a high-quality unit. It is not worth trashing your Raspberry Pi or experiencing random failures for the sake of saving a few dollars.

If you lack an oscilloscope, you won't be able to check how clean or dirty your supply current is. A better power adapter is cheaper than an oscilloscope. A shaky/noisy power supply can lead to all kinds of obscure and intermittent problems.

A good place to start is to simply Google "recommended power supply Raspberry Pi." Do your research and include your USB peripherals in the power budget. Remember that wireless USB adapters consume a lot of current—up to 500 mA.

■ **Note** A random Internet survey reveals a range of 330 mA to 480 mA for wireless USB adapter current consumption.

Voltage Test

If you have a DMM or other suitable voltmeter, it is worthwhile to perform a test after powering up the Pi. This is probably the very first thing you should do, if you are experiencing problems.

Follow these steps to perform a voltage test:

1. Plug the Raspberry Pi's micro-USB port into the power adapter's USB port.
2. Plug in the power adapter.

3. Measure the voltage between P1-02 (+5 V) and P1-25 (Ground): expect +4.75 to +5.25 V.
4. Measure the voltage between P1-01 (+3.3 V) and P1-25 (Ground): expect +3.135 to +3.465 V.

■ **Caution** Be very careful with your multimeter probes around the pins of P1. *Be especially careful not to short the +5 V to the +3.3 V pin*, even for a fraction of a second. Doing so will zap your Pi! If you feel nervous or shaky about this, leave it alone. You may end up doing more harm than good. As a precaution, put a piece of wire insulation (or spaghetti) over the +3.3 V pin.

The left side of Figure 2-5 shows the DMM probes testing for +5 V on header strip P1. Again, be very careful not to touch more than one pin at a time when performing these measurements. *Be particularly careful not to short between 5 V and 3.3 V*. To avoid a short-circuit, use a piece of wire insulation, heat shrink tubing, or even a spaghetti noodle over the other pin.

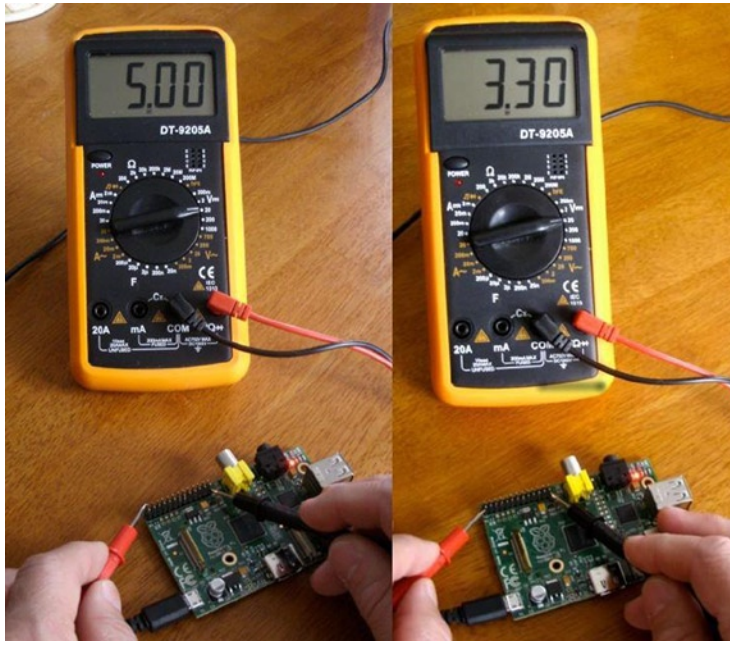

Figure 2-5. *Measuring voltages*

The right side of Figure 2-5 shows the positive DMM probe moved to P1-01 to measure the +3.3 V pin. Appendix B lists the ATX power supply standard voltage levels, which include +5 ± 0.25 V and +3.3 ± 0.165 V.

Battery Power

Because of the small size of the Raspberry Pi, it may be desirable to run it from battery power. Doing so requires a regulator and some careful planning. To meet the Raspberry Pi requirements, you must form a power budget. Once you know your maximum current, you can flesh out the rest. The following example assumes that 1 A is required.

Requirements

For clarity, let's list our battery power requirements:

- Voltage 5 V, within ± 0.25 V
- Current 1 A

Headroom

The simplest approach is to use the linear LM7805 as the 5 V regulator. But there are some disadvantages:

- There must be some headroom above the input voltage (about 2 V).
- Allowing too much headroom increases the power dissipation in the regulator, resulting in wasted battery power.
- A lower maximum output current can also result.

Your batteries should provide a minimum input of 5+2 V (7 V). Any lower input voltage to the regulator will result in the regulator "dropping out" and dipping below 5 V. Clearly, a 6 V battery input will not do.

LM7805 Regulation

Figure 2-6 shows a very simple battery circuit using the LM7805 linear regulator. Resistor R_L represents the load (the Raspberry Pi).

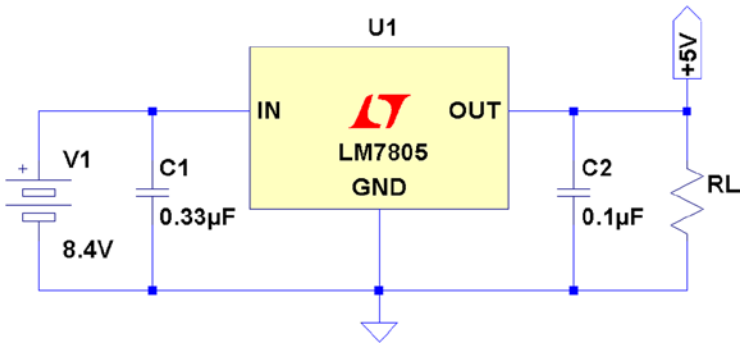

Figure 2-6. *Regulated battery supply*

The 8.4 V battery is formed from seven NiCad cells in series, each producing 1.2 V. The 8.4 V input allows the battery to drop to a low of 7 V before the minimum headroom of 2 V is violated.

Depending on the exact 7805 regulator part chosen, a typical heat-sinked parameter set might be as follows:

- *Input voltage*: 7–25 V
- *Output voltage*: 1.5 A (heat-sinked)
- *Operating temperature*: 125°C

Be sure to use a heat sink on the regulator so that it can dissipate heat energy to the surrounding air. Without one, the regulator can enter a thermal shutdown state, reducing current flow to prevent its own destruction. When this happens, the output voltage will drop below +5 V.

Keep in mind that the amount of power dissipated by the battery is more than that received by the load. If we assume that the Raspberry Pi is consuming 700 mA, a minimum of 700 mA is also drawn from the battery through the regulator (and it could be slightly higher). Realize that the regulator is dissipating additional energy because of its higher input voltage. The total power dissipated by the regulator and the load is as follows:

$$\begin{aligned} P_d &= P_L + P_R \\ &= 5\ \text{V} \times 0.7\ A + (8.4\ V - 5\ V) \times 0.7\ A \\ &= 3.5\ W + 2.38\ W \\ &= 5.88\ W \end{aligned}$$

The regulator must dissipate the difference between the input and the output voltages (2.38 W). This additional energy heats up the regulator with the energy being given away at the heat sink. Because of this, designers avoid using a high input voltage on linear regulator circuits.

If the regulator is rated at a maximum of 1.5 A at 7 V (input), the power maximum for the regulator is about 10.5 W. If we apply an input voltage of 8.4 V instead of 7, we can derive what our 5 V maximum current will be:

$$\begin{aligned} I_{\max} &= \frac{P_{\max}}{V_{in}} \\ &= \frac{10.5\,W}{8.4\,V} \\ &= 1.25\,A \end{aligned}$$

From this, we find that the 8.4 V battery regulator circuit can provide a maximum of 1.25 A at the output, without exceeding the regulator's power rating. Multiply 8.4 V by 1.25 A to convince yourself that this equals 10.5 W.

DC-DC Buck Converter

If the application is designed for data acquisition, for example, it is desirable to have it run as long as possible on a given set of batteries or charge cycle. A switching regulator may be more suitable than the linear regulator.

Figure 2-7 shows a very small PCB that is about 1.5 SD cards in length. This unit was purchased from eBay for $1.40, with free shipping. At these prices, why would you build one?

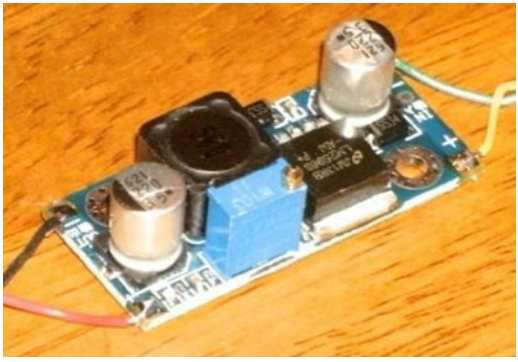

Figure 2-7. *DC-DC buck converter*

They are also simple to use. You have + and – input connections and + and – output connections. Feed power in at one voltage and get power out at another voltage. This is so simple that you'll forgive me if I omit the diagram for it.

But don't immediately wire it up to your Raspberry Pi, until you have calibrated the output voltage. While it *might* come precalibrated for 5 V, it is best not to count on it. If the unit produces a higher voltage, you might fry the Pi.

The regulated output voltage is easily adjusted by a multiturn trim pot on the PCB. Adjust the pot while you read your DMM.

The specifications for the unit I purchased are provided in Table 2-1 for your general amusement. Notice the wide range of input voltages and the fact that it operates at a temperature as low as –40°C. The wide range of input voltages and current up to 3 A clearly makes this a great device to attach to solar panels that might vary widely in voltage.

Table 2-1. *DC-DC buck converter specifications*

Parameter	Min	Max	Units	Parameter	Min	Max	Units
Input voltage	4.00	35.0	Volts	Output ripple		30.9	mA
Input current		3.0	Amps	Load regulation	±0.5	%	
Output voltage	1.23	30.0	Volts	Voltage regulation	±2.5	%	
Conversion efficiency		92	%	Working temperature	–40	+85	°C
Switching frequency		150	kHz	PCB size		45×20×12	mm
				Net weight		10	g

The specification claims up to a 92% conversion efficiency. Using 15 V on the input, I performed my own little experiment with measurements. With the unit adjusted to produce 5.1 V at the output, the readings shown in Table 2-2 were taken.

Table 2-2. *Readings taken from experiment*

Parameter	Input	Output	Units
Voltage	15.13	5.10	Volts
Current	0.190	0.410	Amps
Power	2.87	2.09	Watts

From the table we expected to see more power used on the input side (2.87 W). The power used on the output side was 2.09 W. The efficiency then becomes a matter of division:

$$\frac{2.09}{2.87} = 0.728$$

From this we can conclude that the measured conversion efficiency was about 72.8%.

How well could we have done if we used the LM7805 regulator? The following is a best case estimate, since I don't have an actual current reading for that scenario. But we do know that at least as much current that flows out of the regulator must flow into it (likely more). So what is the absolute best that the LM7805 regulator could theoretically do? Let's apply the same current draw of 410 mA for the Raspberry Pi at 5.10 V, as shown in Table 2-3. (This was operating without HDMI output in use.)

Table 2-3. *Hypothetical LM7805 power use*

Parameter	Input	Output	Units
Voltage	7.1	5.10	Volts
Current	0.410	0.410	Amps
Power	2.91	2.09	Watts

The power efficiency for this best case scenario amounts to this:

$$\frac{2.09}{2.91} = 0.718$$

The absolute best case efficiency for the LM7805 regulator is 71.8%. But this is achieved at its *optimal* input voltage. Increasing the input voltage to 12 V causes the power dissipation to rise considerably, resulting in a 42.5% efficiency (this calculation is left to the reader as an exercise). Attempting to operate the LM7805 regulator at 15.13 V, as we did with the buck converter, would cause the efficiency to drop to less than 33.7%. Clearly, the buck converter is much more efficient at converting power from a higher voltage source.

Signs of Insufficient Power

In the forums, it has been reported that ping sometimes doesn't work from the desktop (with HDMI), yet works OK in console mode.[42] Additionally, I have seen that desktop windows can freeze if you move them (HDMI). As you start to move the terminal window, for example, the motion would freeze part way through, as if the mouse stopped working.

These are signs of the Raspberry Pi being power starved. The GPU consumes more power when it is active, performing accelerated graphics. Either the desktop freezes (GPU starvation) or the network interface fails (ping). There may be other symptoms related to HDMI activity.

Another problem that has been reported is resetting of the Raspberry Pi shortly after starting to boot. The board starts to consume more power as the kernel boots up, which can result in the Pi being starved.[43]

If you lose your Ethernet connection when you plug in a USB device, this too may be a sign of insufficient power.[44]

While it may seem that a 1 A power supply should be enough to supply a 700 mA Raspberry Pi, you will be better off using a 2 A supply instead. Many power supplies simply don't deliver their full advertised ratings.

The micro-USB cable is something else to suspect. Some are manufactured with thin conductors that can result in a significant voltage drop. Measuring the voltage as shown previously in the "Voltage Test" section may help diagnose that. Try a higher-quality cable to see whether there is an improvement.

No Power

If your Pi appears dead, even though power is present at the input, the input polyfuse could have blown. If this was a recent event, allow the unit to cool down. The polymer in the fuse recrystallizes, but this can take several hours. If you think the F3 poly fuse is permanently destroyed, see the Linux wiki page[45] for how to test it.

CHAPTER 3

Header Strips, LEDs, and Reset

In this chapter, an inventory of the Raspberry Pi header strips, LEDs, and reset button connections is covered. These are important interfaces from the Pi to the outside world. You may want to use a bookmark for Table 3-3, which outlines the general purpose input/output (GPIO) pins on header strip P1.

Status LEDs

The Model A Raspberry Pi has a subset of the Model B LED indicators because it lacks the Ethernet port. The Model B has three additional LEDs, each showing the network status. Table 3-1 provides a list of LED statuses.

Table 3-1. *Status LEDs*

LED	Color	Model A	Model B	Comment
ACT	Green	OK	ACT	SD card access activity
PWR	Red	Yes	Yes	Power supply
FDX	Green	N/A	Yes	LAN: Full duplex
LNK	Green	N/A	Yes	LAN: Link
100	Yellow	N/A	100	Labeled incorrectly on Rev 1.0 as 10M: 10/100 Mbit link

OK or ACT LED

This green LED indicates SD card I/O activity. This active low LED is internally driven by the kernel on GPIO 16 (see the kernel source file bcm2708.c in arm/mach-bcm2708).