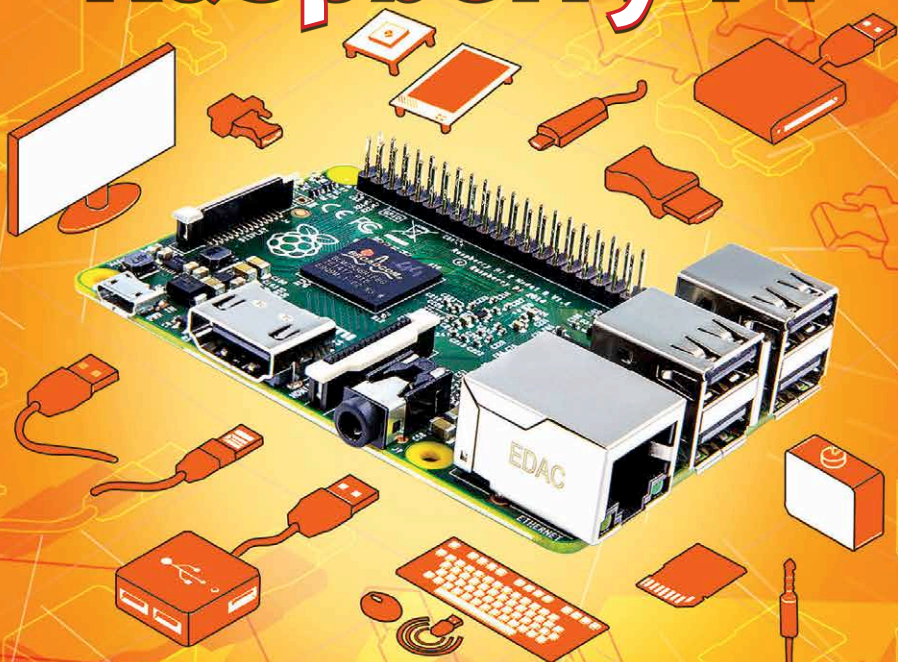


# di*git* FastTrack

YOUR HANDY GUIDE TO EVERYDAY TECHNOLOGY

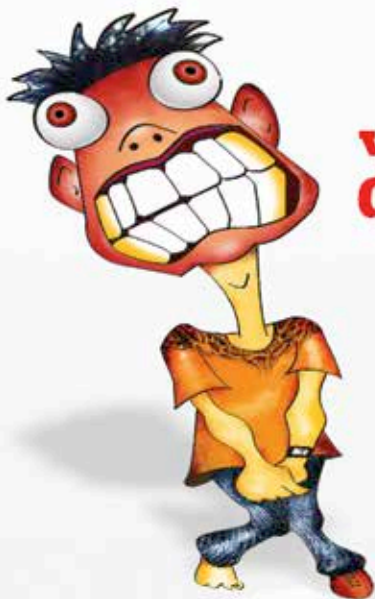
## To Raspberry Pi



- Introduction to RPi
- Evolution of RPi
- Competing boards
- Crazy RPi contraptions
- Getting started with RPi



- DIY Entertainment
- DIY Education
- DIY Home Automation
- DIY Environment
- Future of RPi



**www.  
digit.in/forum**

Join the forum to  
express your views  
and resolve your  
differences in a more  
civilised way.

**digit.in  
FORUM**

Post your queries  
and get instant  
answers to all  
your technology  
related questions



One of the most active online technology forums  
not only in India but world-wide

**JOIN  
NOW**

**digit.in**



# RASPBERRY PI

---

powered by

**digit**

YOUR TECHNOLOGY NAVIGATOR

# CHAPTERS

**RASPBERRY PI**

APRIL 2015

06

PAGE

## Have some Pi

The humble Raspberry Pi had a humble beginning. Learn about the people behind the project.

11

PAGE

## Finger in every Pi

There are many Raspberry Pi models, which one's right for you? That's not all, take a look at the choice of Operating Systems for the Pi.

20

PAGE

## A popular Pi

Definitely not the first of its kind the Raspberry Pi must've pulled a miracle to become this popular. How different is it compared to others?

27

PAGE

## Certainly no humble Pi

It may be the size of a credit card but it sure packs a wallop! Here are some of the awesome creations people have come up with using the Pi.

38

PAGE

## Baking your own Pi

Getting started with the Raspberry Pi is really simple. We even take it a step forward so that you can create your own OS for the Pi.

CREDITS

The people behind this book

### EDITORIAL

#### Executive Editor

Robert Sovereign-Smith

#### Assistant Editor

Siddharth Parwatay

#### Writers

Samir Alam

Vishal Patil

#### Contributing Copy Editor

Infancia Cardozo

### Manager Test Center

Jayesh Shinde

### Reviewers

Mithun Mohandas,

Anirudh Regidi

Abhijit Dey

### DESIGN

#### Sr. Creative Director

Jayan K Narayanan

#### Sr. Art Director

Anil VK

### Associate Art Director

Anil T

### Sr. Visualisers

Shigil Narayanan

Sristi Maurya

### Visualiser

Baiju NV

### Sr. Designer

Pradeep G Nair

52  
PAGE

## DIY Entertainment

One of the most common uses of the Raspberry Pi is to use it as a home entertainment system. We show you how and more.

58  
PAGE

## DIY Education

Teaching kids how to tinker with code was and continues to be the primary driving force behind the development of the Pi. Here are some simple DIYs for you to start your learning journey.

71  
PAGE

## DIY Home Automation

Fancy a fully automated home like the one Tony Stark has in Iron Man? Start small with a few useful DIYs and soon you'll have your own Jarvis.

83  
PAGE

## DIY Environment

Doing your part for the environment isn't an uphill task. Check out some really simply DIYs to monitor your environment.

92  
PAGE

## Ain't no Pi in the sky

Clearly, this is just the beginning for the Raspberry Pi. We consult the crystal ball to see what the future holds for the Raspberry Pi.

### © 9.9 Mediaworx Pvt. Ltd.

Published by 9.9 Mediaworx

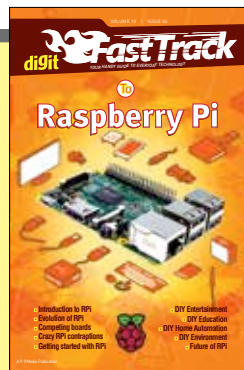
No part of this book may be reproduced, stored, or transmitted in any form or by any means without the prior written permission of the publisher.

### April 2015

Free with Digit. If you have paid to buy this Fast Track from any source other than 9.9 Mediaworx Pvt. Ltd., please write to [editor@digit.in](mailto:editor@digit.in) with details

### Custom publishing

If you want us to create a customised Fast Track for you in order to demystify technology for your community, employees or students contact [editor@digit.in](mailto:editor@digit.in)



COVER DESIGN: PETERSON P

# Introduction

Welcome to one the most exciting Fast Tracks you will ever read. This month we dive deep into the world of the Raspberry Pi microcomputer. The latest model of the device - the Raspberry Pi 2 - was released in February 2015 and we happened to get our hands on a few of these beauties. After weeks of tinkering around and exploring the various unique projects, we are happy to present the Fast Track to Raspberry Pi.

We start off this month's issue by explaining what a Raspberry Pi is exactly and clarifying a few misconceptions regarding its usage. We explain how the Raspberry Pi isn't a conventional computing device and justify how it needs to be critiqued against a unique benchmark. We trace the history of its origins and present a detailed overview of its hardware evolution - demystifying how such a versatile and powerful device can be so cheap.


This edition also delves into the reasons as to why the Raspberry Pi has achieved such unprecedented popularity despite alternatives like the BeagleBone and Intel Galileo being more powerful. This eventually leads us into the innovation philosophy and broader goals of the Raspberry Foundation.

We then take a step-by-step journey to guide the first time Raspberry Pi user from unboxing to complete set-up. The chapters also include the various options that come with the Raspberry Pi with respect to its scalability and range of operating systems. This chapter also wets readers creative appetite by taking a closer look at some really interesting projects that have been implemented using the Raspberry Pi - from a weather camera to a supercomputer.

We hope to aid the reader in starting off with some really simple but useful Do-It-Yourself projects using the Raspberry Pi through a series of chapters. There were a tonne of DIYs to choose from and we've broadly

classified them into entertainment, education, home automation and environmental projects. And finally we discuss the potential that cheap microcomputers and the Raspberry Pi have in influencing the future of technology across the world.

We really look forward to you enjoying this edition of Fast Track and welcome your feedback at: [editor@thinkdigit.in](mailto:editor@thinkdigit.in)

Happy tinkering! 

**Note:** The Raspberry Pi 2 is relatively new and quite a lot of the projects that work on the Raspberry 1 are yet to be fully functional on the newer model owing to the fact that certain software packages are still being re-written for the new hardware. We've put together a small appendix to help you in this aspect. You can check it out here : <http://digit.in/PiAppendix>



# HAVE SOME PI

What is the Raspberry Pi and how does it taste? We demystify the concept of the pocket sized computer that has taken the world by storm.

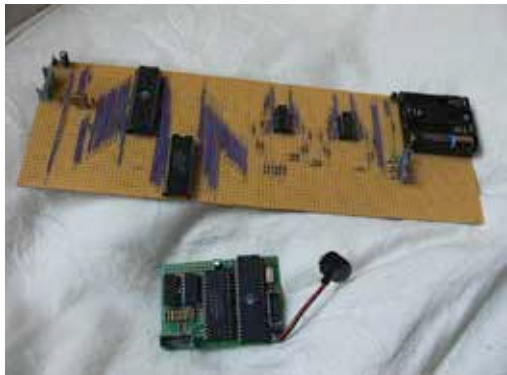
**T**he early days of the computer age were riddled with young kids exploring these new and fascinating devices. But as computing became a ubiquitous phenomenon in everyday life, hidden behind the facade of games and software, the mystery was lost. The number of kids (and adults) who would tinker around with a computer's code or electrical board sharply declined. People simply became users and the computer an appliance. The introduction of the Raspberry Pi pocket computer was in response to this phenomenon and it has given a new lease to "makers" and DIY enthusiasts.



In its simplest form, the Raspberry Pi is a series of pocket sized single board computers developed by the Raspberry Foundation in the United Kingdom. The vision behind its creation was to foster awareness and training of fundamental computer knowledge within the education system. But as with all good fundamental inventions, the Raspberry Pi has captured the imagination of millions across the globe in its applications and utility. Since its 2012 launch the Raspberry Pi has sold nearly five million units worldwide and has already launched its new version - the Raspberry Pi 2 - which improves upon the original version with significant updates. The computer board has accomplished this by ensuring that it remains highly affordable to even the most casual users, with three price brackets of USD20, USD25 and USD35 which is equivalent to ₹1,256, ₹1,570 and ₹2,197. The core reason behind its mass appeal is the degree of freedom and customisation it allows for in the hands of an enthusiastic user who is willing to learn and experiment.

## The Master Chefs

In early 2005, the inklings of the Raspberry Pi started taking shape in the United Kingdom at Cambridge. A trio of alumni from the University of Cambridge were concerned that the calibre of applicants in the computer sciences had begun lacking. The three visionaries who would eventually become the originators of the Raspberry Pi foundation were Eben Upton, Pete Lomas and David Braben. All three remembered the early days of home computing when set-ups like the BBC



Original idea was for users to be able to make their own board.

Micro B, C64 and the Spectrum systems allowed home users to experiment and educate themselves in the basic computing skills necessary to become an effective programmer. They simply wanted kids to take an interest in learning how to program.

Eben Upton spent many years trying to design prototypes of his own even as he continued working with global chip designer Broadcom. In his attempts to make a viable product he reached out to his university friend Pete Lomas who worked in hardware design at Norcott Technologies and David Braven, who was the co-creator of the old-school BBC Micro game Elite. The three of them joined their forces to create the Raspberry Pi Foundation that was tasked with the precise goal of making affordable pocket sized home computing set-ups that would engage and educate kids in computer programming. Within three years the Foundation had gone into mass production with the original Raspberry Pi Model B and followed it up with the cheaper Model A.

## An Original Recipe

The conceptual basis of the Raspberry Pi was rooted in the early 1980 models of the BBC Micro computer systems. The 8-bit BBC Micro was created over 30 years ago by Acorn Computers for the British Broadcasting Corporation in its quest to promote computer literacy. As a part of the BBC Computer Literacy Project, the Micro was responsible for being an introduction to generations of children in Britain such as the creators of the Rasp-

berry Pi. And while the BBC Micro was sleek for its time it still had various drawbacks which the Raspberry Pi wished to improve upon for the current age. In order to achieve its goal of educating and making computing more accessible to children, the Raspberry Foundations turned to its potential users.



The BBC Micro served as inspiration for the Pi.

By bringing together an array of teachers, academics and programming enthusiasts, they were able to get as many insights into the needs of their future invention.

The core essentials of the project were always centred on affordability. Unlike the BBC Micro device of the 1980s, which was priced in hundreds

of pounds sterling, the Raspberry Pi set its goal at a fraction of the cost. In order to do this the computer system was stripped of all peripherals and accessories. The product only supplied the basic electronic hardware and the programming environment which would allow it to interface with other required devices.

## Early Prototypes

The early prototype was based off an Atmel ATmega 644 microcontroller board in 2006. The next functional ARM version was in fact as small as the size of a pen drive with only two ports - one HDMI and one USB. The Atmel microcontroller was only clocked at 22.1 MHz with 512K SRAM for storage and buffer memory. The video signal capable on this set up was at a measly 320x240 resolution and could only generate simple graphics output. Within a couple of years, as mobile processors become cheaper and more affordable the project moved onto using ARM chips.

The first design on the Raspberry Pi was the Model B. In order to realise the goal for the Model B, Eben Upton approached RISC OS Open Community in 2011 to help develop the ports for the device. Eventually, Broadcom's Adrian Lees took on the challenge and even went on to contribute to the boards graphics capabilities. By August 2011, the first functional alpha boards of the Model B were in production but were larger in form factor due to hardware issues. Despite the size problem the boards were capable of running Quake 3 at Full HD(1920x1080) and processing Full HD video streaming over HDMI.

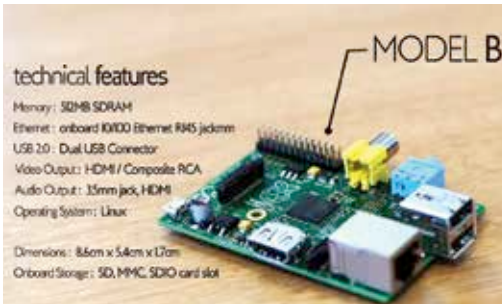
By October 2011, RISC OS 5 had been developed and was expected to be ported on to the Raspberry Pi within the year. Over the course of that year, multiple limited production runs were carried out for the Pi. The first Model B Beta boards came out of production in December 2011 but suffered from a single hardware error that was immediately rectified. Upon completion the Beta boards proved successful at playing Full HD video and measured up on the OpenGL ES benchmarking for graphics processing. As the final polish and fixes were implemented the launch data for the Pi drove ever closer. As a charitable and promotional step, 10 boards were put on auction in January 2012. The auction raised over GBP 16,000 on a retail value of GBP 220.

Just before the sales date the Raspberry Pi was fitted with the ability to run an operating system off an SD card in February 2012. The initial OS in play was the Debian 6.0 or Squeeze edition with the LXDE desktop, Midori

browser and an assortment of programming tools. With all the preparations finished, the six year journey of the Raspberry Pi's development led to the sale of the Model A on 29 February 2012. The Model A which was initially planned to have only 128 MB of RAM was quickly augmented before the sale to provide 256MB of RAM. In certain cases there were over four million pre-order enquiries and over 100,000 confirmed pre-orders for the first run of the boards.

## The Raspberry Pi Menu


Now the Raspberry Pi and Pi 2 comes in a variety of models. The most affordable model from the first series is the Model A+, priced at USD 20 with the following specifications: Broadcom 2835 SoC (System on a Chip) which

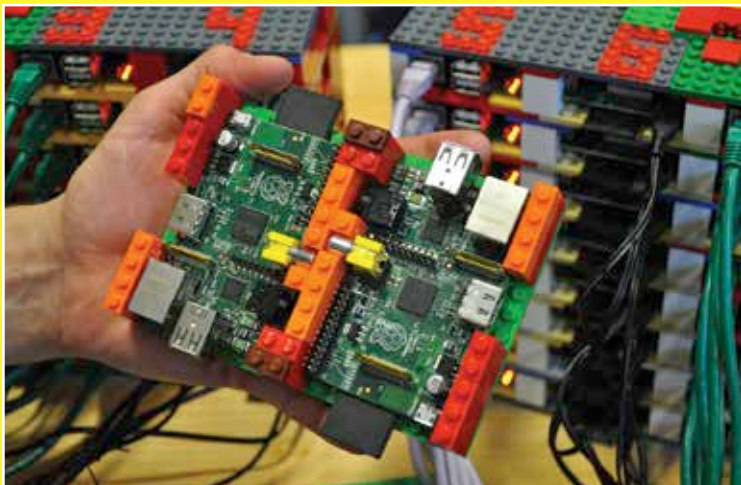


Model B remains the most popular version of the Raspberry Pi.

includes the CPU, GPU, DSP, SDRAM and one USB port, a 700 MHz single core ARM processor, a Broadcom VideoCore IV processor clocking at 250 MHz with 24 GigaFlops of OpenGL ES 2.0 operational speeds, 256 MB SDRAM

which is shared with the GPU, a 15-pin MIPI CSI connector, a HDMI PAL/NTSC compatible output with a 3.5 mm audio jack and a MicroSD slot. The Model A+ was powered by a 5 V source via the MicroUSB port or the GPIO header and is approximately 2.5-inches on each side weighing in under 23 grams.

The only non-consumer model that was created in 2014 was the Compute Module which used the same SoC from Broadcom with a 512 MB RAM to be used as a part of embedded systems. These modules are available for purchase in batches of 100 and are priced at USD 30. The Model A, Model B, Model B+ and the new Raspberry Pi 2 had small variations, in departments such as processor power, RAM capacity and Ethernet ports. But the biggest differences were in their prices which still kept the newest and most powerful options - the Model B+ and Pi 2 - affordable at just USD 35. We will take a closer look at the variations in the next chapter. 



# A POPULAR PI

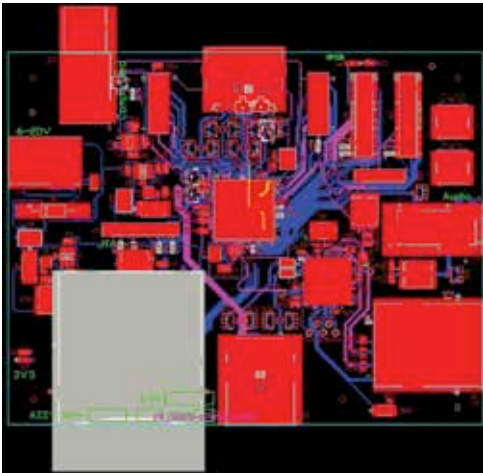
The Raspberry Pi isn't the only pocket sized computer on the market. But what makes it the most appealing to consumers?

Since the launch of the Raspberry Pi in February 2012 we have seen the popularity and the variety of its uses grow at a radical pace. The original Raspberry Pi series of microcomputers was limited to only a handful of models which is a tradition it continues even now. In total, there are only five Raspberry Pi products, of which one is the Compute Module. In February 2015, the company announced the release of their new model the Raspberry Pi 2 Model B which incrementally advances the capabilities of the product line. The road to this new design has been a slow and meditated one which has kept true to the original ideals of the project that was envisioned back in 2006. We discuss how the company has evolved its devices and its market place.

## Prototype Pi to Present Day Pi

As we know the creators of Raspberry Pi were motivated by the lagging programming literacy in Britain. As alumni of the prestigious University of Cambridge, and as professionals in the computer industry, they recognised that this diminishing aptitude for core computing skills needed to be fixed. The purpose of the Pi was to make computing and programming an early development skill which could be used to educate children. The core features of the product they imagined was to be an open learning environment which could be bought for cheap by any school and parent. With that in mind they set about looking at the current landscape and decided to build a microcomputer which could form the basis of this learning experience.

The design and philosophy of the device was borrowed from 1981's BBC Micro computer made by Acorn Computers. But with the advantages



The Alpha design schematic is a work of art in itself.

of the miniaturisation of components accessible it was decided in advance that the Raspberry Pi would be about the size of a credit card. The lead designer of the project was Eben Upton, who was a chip architect at Broadcom, and was working to bring that form factor to life.

The early 2006 concept design was made using raw materials in a homemade

setting. It was initially hoped that users could assemble their own very cheap microcomputers at home. Using a veroboard base, Upton employed the Atmel ATmega644 microcontroller which clocked only 22.1 MHz with 512K of SRAM for data and frame buffer storage. The Atmel microcontroller only had 32 GPIOs, of which 19 were used to drive SRAM address bus needed to generate the video output. The video resolution was limited to 320x240 which the microcontroller juggled while performing other operations. Given

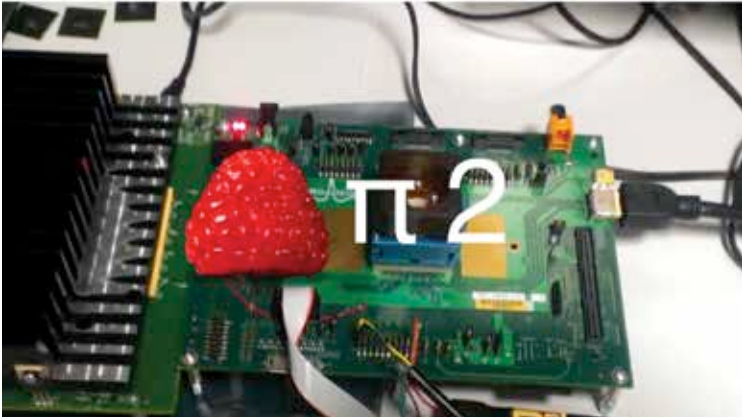
the limitations of this particular design, home assembly was abandoned in favor of higher performance necessary to run an operating system.

In the next couple of years the boom in smartphones indicated that an affordable processor unit could form the backbone of the Raspberry Pi design. As custom microprocessors were being developed at Upton's own employer - Broadcom - affordability and performance were both becoming realistic for the Pi. The next iteration of the device abandoned all homemade elements like veroboards, DIP chips and through-hole components and looked to fine crafted manufacturing assembly. The first alpha design board was sent for manufacturing in July 2011, just months before its February 2012 launch. The board was identical to what would be the final release with exceptions in its slightly larger size and more layers. It used an ARM processor along with an Ethernet controller. These alpha boards were used primarily to run software applications testing. Within a month of the alpha board testing the Raspberry Pi team was successfully running Quake 3 Deathmatch on a set of networked devices in the office.

The next major development was with refining the power distribution layout of the board's design. The challenge remained to keep the power consumption stable while ensuring a stable route for all components. This problem could have been easily solved if the team was willing to spend more money on the board designs. But given their commitment to keeping the price point low they attempted to figure out another solutions. Eventually, the team arrived at the solution of using microscopic laser made holes in the circuit board to evenly distribute board space for different electrical components. The wiring channels or micro vias would only increase the per unit price by pennies when done at a large scale using lasers. The ability to wire the printed circuit boards or PCBs at different levels of its thickness allowed the final design to become fully functional.

The basic layout and design of the Raspberry Pi remained fairly unchanged for the bulk of its development. Once the team had discovered the right low cost ARM processors for their needs, the rest of the board layout and design was easy to execute. The real challenge was towards mass production and business partnerships which would allow them to leverage economies of scale and allow their price points to remain unchanged. The final beta test boards for the Model B were wrapped up in December 2011 and ten of them were auctioned off online. The final production line for the Model B started off in January 2012, while the first root filesystem disc image was developed for Debian release in February. The preorders for the Rasp-





Pi board can be customised in function as well as form.

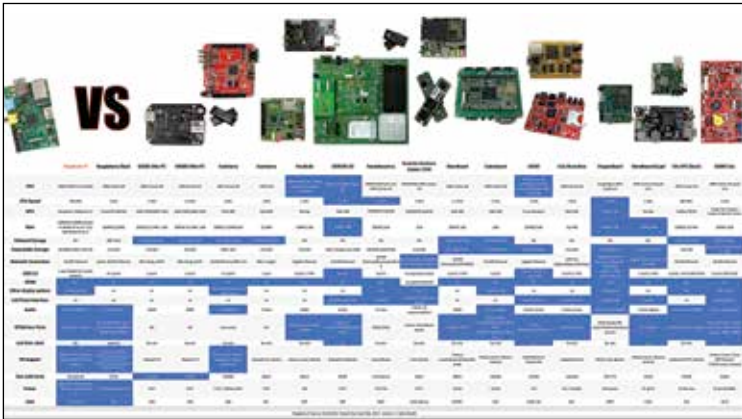
Raspberry Pi had already crossed 100,000 and forced multiple manufacturing plants to be engaged in ensuring timely delivery. The subsequent Model A, A+ and B+ were released before the end of 2012, and the latest Raspberry Pi 2 was released on 2 February 2015.

### **Pi 2 or Not Pi 2: What is the difference?**

Every decision that the Raspberry Pi team has made with the release and development of their microcomputers has been rooted in price. At no point have they given in to the temptation of spending a few dollars more per unit in order to gain any advantage. The same is true of the new release that they introduced with the Raspberry Pi 2. Since the original launch of the Pi, the prices for key components like processors and memory have come down, which means a higher degree of performance can be achieved at the same costs. With this in mind, the team decided to take their Model B+ variant and give it a makeover. The new Raspberry Pi 2 is now equipped with an ARMv7 Cortex-A7 processor that clocks 900 MHz and has a memory enhancement which doubles it to a total of 1GB. That's it. Such a small upgrade may appear to be too modest to worth mentioning, but in the world of the microcomputer such changes are epic - specially when the prices don't change.

In terms of performance, these minor upgrades pave a whole new level of implementation including the operation of high level operating systems like Microsoft Windows 10. This is primarily to the Broadcom BCM 2836 which





There really is no comparison between the Pi and its contemporaries. (<http://dgit.in/PiVsAll>)

replaces the former Broadcom 2835 SoC (System on a Chip) foundation. In addition to which the former single core 700 MHz ARMv6 processor is replaced with the new quad-core 900 MHz ARMv7 Cortex-A7 processor. While this doesn't radically alter the physical design of the chip there is a minor power consumption uptake and subsequent heating issue if the chip is overclocked.

Despite the changes that come with a new board, such as ARMv7 modules and kernel, it will not obstruct compatibility with ARMv6 user space binaries and will run existing Linux distributions for the original Raspberry Pi. The only flaw of note is an amusing one - the Raspberry Pi 2 is "camera shy". This fault was discovered when the microcomputer was being photographed with a Xenon flash. In certain specific cases, the high intensity of the flash can cause a hiccup in the voltage flow causing it to reboot. This problem is due to a photoelectric effect, which is when metals give off electrons when hit by light. This problem is easily fixed by putting a tiny opaque cover on the U16 chip between the power supply USB and the HDMI port. Clearly this flaw hasn't lessened the excitement of Raspberry Pi's core and dedicated following since the initial stock of the Pi 2, which was of 150,000 units was sold out in no time. And the company continues to manufacture 20,000 computers per day to meet the increasing demand.

## The Cost of Keeping Costs Low

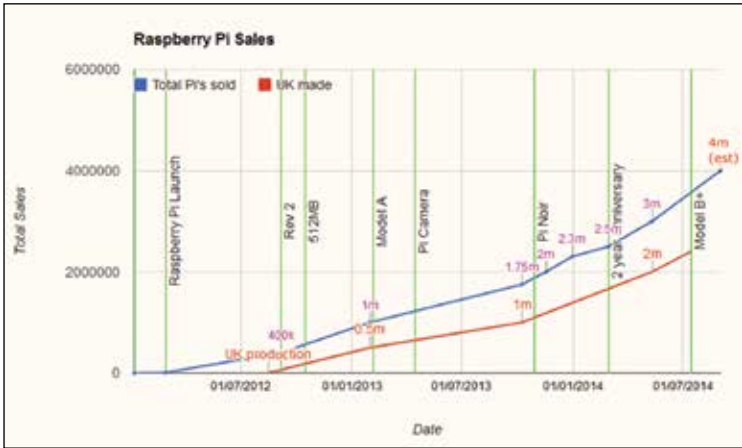
There are two broad criticisms of the Raspberry Pi - they're not that cheap

when you add up all the peripherals and they're not that powerful when you compare them to the competition. If we accept these criticisms at face value, we have to ask ourselves - where did they cut corners? Lets begin with accepting the first criticisms - Raspberry Pi's are not that cheap - true. Apart from the top-tier USD 35 dollar expense on the new Pi 2, when you begin adding up the costs of USB keyboard, mouse, HDMI-screen, WiFi dongle, Ethernet cable, HDMI cable, Pi case, speakers, external storage and a USB hub, you can easily end up spending nearly USD 300 on putting together your Pi computer system. The second criticism is also true - the power and capacity of the fully assembled Pi computer system can easily be outdone by something as basic as a generation old Chromebook. But both these criticisms are irrelevant because they miss the point of the Pi as a different type of computer. It would be like criticizing the Space Shuttle for giving really bad mileage on Indian national highways.

The reason for the cheapness of the Pi and its spartan setup isn't a deficiency, it is a feature. The Raspberry Pi never claimed to be an alternative to home computers like laptops and tablets. First and foremost, it is worth remembering that the company that designs and releases the Pi - Raspberry Foundation - is a not-for-profit educational organisation based in the United Kingdom.

This means that they do not hoard and profit-share the revenue from the millions of units they sell. The Foundation is rooted in its vision statement, which is to make computer literacy within the reach of children and schools. By limiting the functionality and components of their product they are able to keep the costs low, and allow integration of their device with existing IT infrastructure in schools and universities. However, the fame of the Raspberry Pi has come from well-funded independent projects by people across the world who find appeal in its form-factor and versatility. There "makers" use their ingenuity to make unique, customised and useful projects on the Raspberry Pi. Many inventors like the convenience of using the Raspberry Pi for quickly and cheaply prototyping their projects. All of this exists outside the primary domain of the mandate that the Raspberry Foundation started out with.

The status of the Raspberry Foundation as an educational non-profit is another reason they are able to keep the costs low, since they do not have to bear expenses which private technology firms endure. They, in fact, enjoy the support from companies like Broadcom (where a co-founder works) to get significantly lower rates on their SoCs and chips. This partnership



The Pi is expected to double sales in the next two years.

was only created because of the overwhelming demand for the project. It is worth remembering that the initial product run of the Pi was planned to be around 10,000 throughout its lifetime.

It surpassed that in pre-orders by a factor of 10 when people started taking an interest. In order to meet the production requirements necessary to produce millions of units, they partnered with private companies like Broadcom and Sony to ensure affordable assembly and components. This required a certain tradeoff which still restricts them from making the hardware design open source, but it is a cost worth paying to ensure the availability of the device to any young child looking to learn programming and computer sciences. In short, the Raspberry Pi doesn't cut corners just because it is cheap - it is a precision tool for educational purposes. And while its fame comes from its off-label usage in maker projects, that isn't their primary audience. They have even said that the best thing they hope for is that some Chinese company cracks their design and starts selling versions that are cheaper, since it will make it more accessible.

## Unofficial Accessories

The Raspberry Pi can be used in thousands of different ways. But to truly unlock its advanced features it helps to have add-ons. Apart from the basic accessories such as multiple micro USB cables, at least 4 GB mini SD storage card, USB keyboard, USB mouse and an HDMI compatible screen there are

many more add-ons. These accessories not only enhance the versatility of the device but in some cases make the final product, whatever it may be, fun as heck.

**Gertboard:** The Gertboard is an extension microcontroller for the Pi designed by one of the original co-designers of the Raspberry Pi. Compatible with Arduino projects, the Gertboard allows for projects that require more connections and performance than a standard Pi can give, such as those needed in robotics and motor-based designs. Not for the casual user.

**Pi Camera Module:** The camera for the Raspberry Pi allows for 1080p30, 720p60 and 480p60/90 video and 5 megapixel photography. The camera board is very small with 25 mm x 20 mm x 9 mm size at only 3 grams. It connects to the Raspberry Pi's CSI bus and supports high rate data transfers. The quality of the footage is impressive.

**Slice of Pi:** This is an expansion board which once assembled offers interesting features such as a custom development environment, wireless modules and onboard soldering points for more sturdy prototypes and IoT based systems.

**4D LCD Screen:** This custom designed LCD screen is made for the Raspberry Pi comes in sizes from 2.4-inches to 4.3-inches with 65K true life colours. The TFT screen resolution is 480x272 and is also a resistive touch screen suitable for touch or stylus. It connects directly with the Pi for an assortment of projects like a personal iPod.

**Piboards:** The Piboard is a programmable control relay that works to control external hardware via the Raspberry Pi. The board support multiple programming languages like Python, C and Scratch.


**Storage Cases:** There are a variety of interesting storage cases for a Raspberry Pi which can be found suitable for different processes. You can even commission artists to customise and 3D print cases on demand.

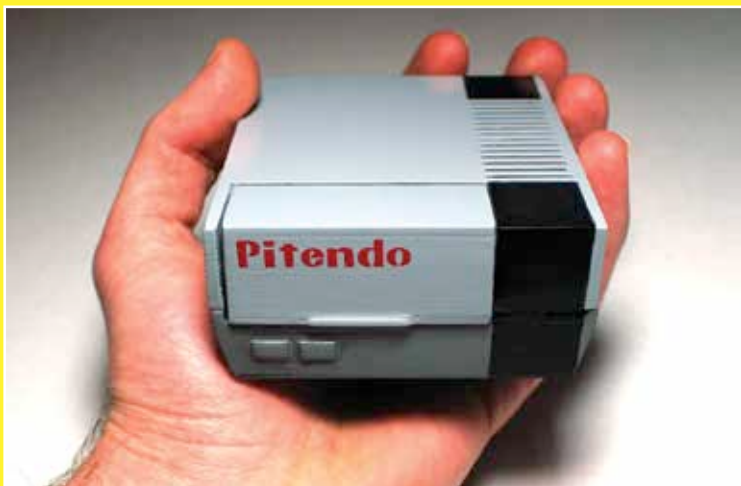
**Pi LITE:** This is a Kickstarter funded basic LED add-on display for the Raspberry Pi which can be used in many interesting ways. It's a basic display tool which uses a grid of 14x9 pixels made from 126 LEDs to display Pi programmed text. It can range from a timer to a Twitter feed. It's all up to the imagination of the user.

**XLoBorg Sensor:** Another one of the big boy toys, this sensor allows motion and directional information to be fed to the Pi. It adds a 3-axis accelerometer along with a 3-axis magnetometer data input to your projects. It helps to detect motion, position and orientation related information to the project being run on the Pi. Great for robotics or smartphone related projects.

**PicoBorg:** For motorised projects that require programmable control, the PicoBorg is a small DC motor controller. It interfaces with the Pi to provide control over up to four devices. It can also be used to control fans and solenoids.

**RadioBlocks:** This is an IEEE 802.15.4 protocol based wireless modem that connects to the Pi to enhance its connectivity features. Using standard internet protocols it can be also be paired with extra-microcontrollers and be activated from any embedded development platform including the Arduino and BeagleBone.

**BrickPi:** This is the funnest of all accessories since it is used to combine the Raspberry Pi with LEGO Mindstorm systems. By combining the two worlds users can create dynamic and motorised LEGO made projects. The BrickPi supports up to three NXT motors and uses the default LEGO Mindstorm batteries. 



# A FINGER IN EVERY PI

We explore the path that led to the Pi's evolution from prototype to the Pi 2. How do they do it?

**M**iniature consumer grade computers are not a new thing. As far back as 1968 we find examples of highly customisable calculators made by Hewlett-Packard that would fit the definition. Over the next couple of decades the numbers increased with Datapoint 2200 being the first official microcomputer to be released in 1970. Since then the changes in microcomputer technology have tracked changes in the rest of the computer industry. With products getting smaller, faster and more powerful in their capabilities. However, their biggest value has been the ability for computer enthusiasts to use them for learning

programming and implementing their own applications.

This attribute sadly fell by the wayside during the 1980s after the release of Apple's enterprise applications for the Apple II microcomputer - which then began to redefine the appeal of microcomputers for modern day consumers. Within the next couple of decades the idea of a computer enthusi-



Early microcomputers were larger than today's consoles.

ast microcomputer was replaced by products like mobiles, tablets, handheld game consoles and laptops. No longer were consumers engaging with the technology, now it was all about applications. It was only with the release of the Raspberry Pi that the world began to take notice of the latent demands in the market for old-school enthusiast driven computing. This by no means implies that Raspberry was the first microcomputer of the current era - in fact, there remain many vital products that appear to compete with Raspberry Pi, such as the Beaglebone, Beagleboard, Intel Edison and Intel Galileo. But in the face of such competition the Raspberry Pi has remained very popular going on to sell millions of units. We explain why.

## Clarifying the Competition

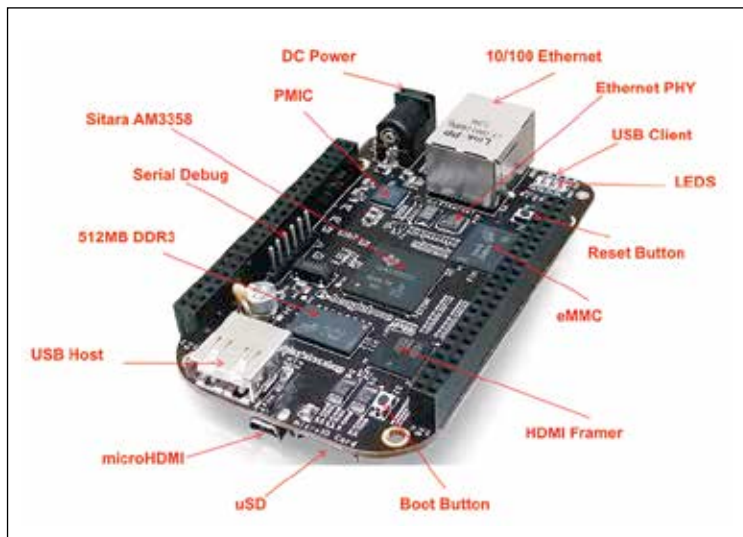
Before taking a look at the Raspberry Pi's closest direct competitor we should clarify certain misconceptions. Many people think that some Arduino boards or certain Intel products can be compared to the Raspberry Pi in terms of function. Nothing could be further from the truth. There is a fundamental difference between the Arduino and the Pi - the former is a microcontroller, while the latter is a micro computer. The microcontroller is only a compo-

nent of the microcomputer. They are fundamentally different products for different types of projects.

The Arduino and the Intel Edison fall under the category of microcontrollers. While the Intel Galileo is a product of the partnership between Arduino and Intel, it is oriented towards open source hardware users, who wish to have control over elements such as the processor. In contrast, the Intel NUC (Next Unit of Computing) is a directly comparable to the Pi but it is designed keeping in mind the Internet of Things framework for wireless projects. It is also priced at nearly four times that of a Pi.

## Raspberry Pi vs. Everything Else

Many people harbour the misconception that the Raspberry Pi is a new form factor for a home PC - it isn't. The processing power and memory on the Pi is at its best equivalent to a mid-range smartphone. The device doesn't even come with a storage unit - it needs to be purchased separately, just like the display, control interface, wireless devices and pretty much everything else. The Raspberry Pi is an educational device which is why comparing it to other similar products is tricky. We clarify this point with comparisons between Raspberry Pi and its only other competitor with regards to its educational and hobbyist ideal



Competitors to the Pi provide more powerful features yet lag in sales.

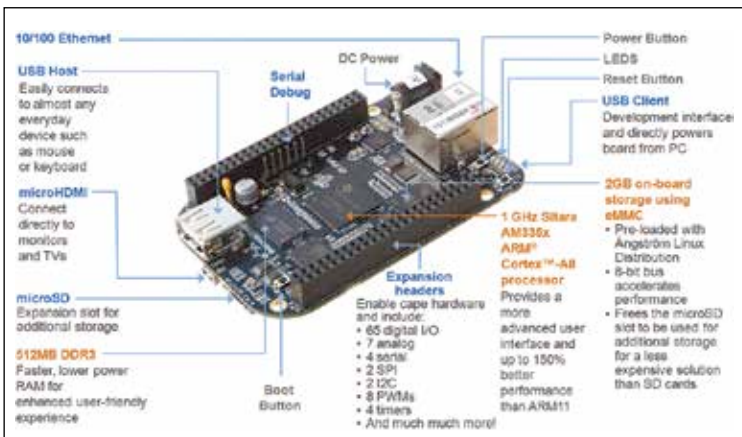


## The Beaglebone

The original Beagleboard was launched late to the party in 2008. Only recently has its Beaglebone Black model from 2013 made serious strides towards sales and popularity. The original BeagleBoard was far more powerful in terms of its configuration while maintaining the small form factor. It allowed for advanced usage by hobbyists and makers for complex projects. However, the cost of the original BeagleBoard is at USD 125 with the BeagleBoard-xM variant at USD 145. This price barrier alone has prevented the BeagleBoard from finding a mass consumer base. The BeagleBone Black model is a stripped down version of the original device at the price of USD 45.

The BeagleBone has a faster ARM processor, DDR3 RAM, 2 GB storage, higher GPIO capability and Linux OS support. These upgrades alone more than make up for the ten dollar price difference between the two products. However, the BeagleBone Black only provides for a micro HDMI, while the Pi has HDMI and a composite display output. Another major deficiency in the BeagleBoard Black is its lack of Full HD resolution limited to a 1440x900 display. The BeagleBoard also lacks a separate audio out, unlike the Pi's 3.5 mm audio jack and draws 50 percent more power than the Pi to be operational.

Overall, the BeagleBoard does have its advantages. It is far easier to set-up than the Pi since it comes with basic peripherals such as a micro-USB cable and mini SD memory card. The Pi also lacks a pre-installed OS which isn't a problem in BeagleBoard given its onboard storage. The BeagleBoard only



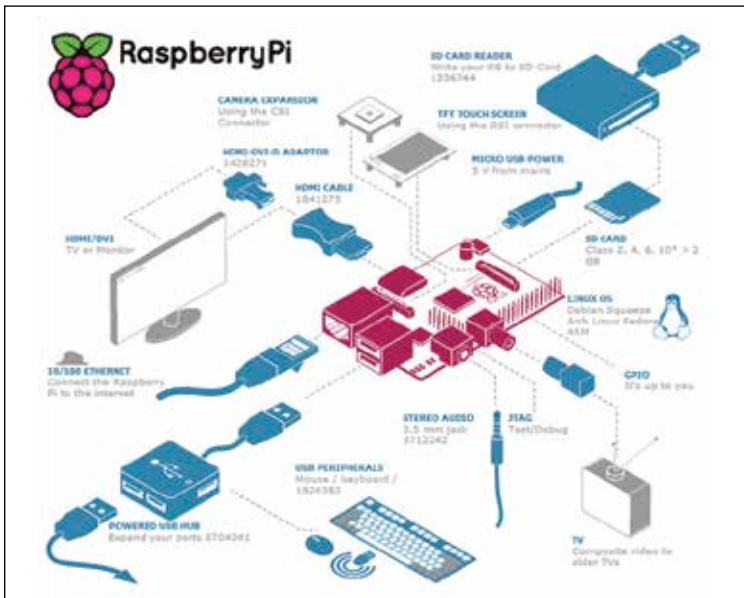
The Beaglebone is a powerhouse of features for advanced makers.

has one USB port which can be problematic and requires a USB hub to make it fully functional with mouse, keyboard and other accessories. But purely from a developer's perspective, the BeagleBoard is far superior given its vastly higher number of connection ports (92 at last count) which allow for great expansions. But the bottom line winner has to be the Raspberry Pi which thanks to its Arduino-supported add-ons, allows cross-compatibility between the Pi and over 300 Arduino Shields.

As newer models come out it isn't inconceivable that the BeagleBoard Black will advance its specifications to match up to the Pi. But in terms of pure geek-quotient, it is necessary to say that the BeagleBoard is a better product for a variety of uses. But as long as the online community is heavily skewed in favour of Pi related DIY projects, we can safely expect to see the Raspberry Pi dominating the market.

## Pi for the Growing Brain

The world renowned astrophysicist and educator Neil deGrasse Tyson has a very elegant idea about how best to foster science literacy among the population. To paraphrase, children are driven by curiosity and naturally



The form factor of the Pi allows for seamless hardware integration.


seek out experiments in their environment. Their persistent drive to ask the question “why” can, however, drive many parents to frustration at the best of times. And even for those parents who wish to support the self-education of young children the costs can be discouraging. The damage done by children in their exploration of the world also has a monetary cost which all parents can’t afford. So in the case of computing and programming, even if a parent wants to encourage their child to explore and discover, a home computer worth tens of thousands of rupees may not be worth the risk. But what if it wasn’t that expensive? And what if it was scalable in cost over time? A teaching tool that grows with the child’s imagination - that is what the Raspberry Pi proves to be.

The success of the Raspberry Pi isn’t because it is the fastest or the most powerful or the most easy to use microcomputer. It’s because it is the blank canvas that teaches people how to paint. While it was originally intended to foster an interest in programming and computing among children, it has proven that the broader population of adults is just as curious when it comes to computers. While almost everyone familiar with a computer has used it - either as mobile devices or PCs - they haven’t ever had the opportunity to explore what makes it work. The computer has always been a magic box with a beautiful facade through which user experience is oriented. But as people are discovering with the Raspberry Pi, computers can be reduced to understandable concepts which can be physically worked upon to achieve a customised results.

The explosion in Do-It-Yourself or DIY projects around the Raspberry Pi has demonstrated that with a barebones setup, people of different means and interests, can create their own learning environments. The highest priority of needs with a learning tool isn’t its novelty but its utility - and that is where the Raspberry Pi is able to differentiate itself from its competitors. While other micro pocket-sized computers come in packaged boxes, with wireless features and higher specifications, they take away the learning opportunities regarding how to implement wireless systems independently. The Raspberry Pi puts together the basic set-up of a computer and leaves it in the hands of the user to do with as they please. And to help them is a vast array of open source DIY projects available to explore - from simple ones that allow for a computer in your TV to something as creative as a robotic assistant.

The killer feature of the Raspberry Pi in this context is the low buy-in cost. For even the most financially conservative user, the price tag of the cheapest

Raspberry Pi (A) model is worth the expense. Once a user begins advancing in their ability to program using code and refines his/her understanding of how computer circuitry works, they may decide to move things to a higher gear. In this case, they can purchase a more advanced model or in some cases manually augment the existing Pi board with the help of their newly acquired know-how. In other cases, the low cost of the Raspberry Pi works great towards building computer powered prototypes in other projects by experienced professionals. Need to build a motion tracking camera rig - no problem. Want a home based supercomputing powers - done. All of this at small incremental costs that suit most people's technology budget while allowing massive customisation.

The Raspberry Pi may be built on the philosophy of the open source movement using the cheapest hardware available but that is what is required when the goal is to make core computing technology accessible to everyone. Amongst all its competitors that we have previously outlined - we find that the three features that distinguish the Raspberry Pi are - low cost, creative scalability and an enthusiastic open sourced community. In some contexts, the success of the Pi is a riddle the likes of the chicken and the egg - which came first? While other devices and products have more advanced features at proportional prices, they haven't been able to build as diverse and imaginative a community as the Raspberry Pi. And one of the reasons the community has been so responsive is mainly because the Pi is cheap, easy to learn by any age group and capable of being upgraded to individual desires - in other words it is a technology that allows creative and financial freedom to its users. 



# CERTAINLY NO HUMBLE PI

We walk you through from the basic setup to the advanced possibilities of the Raspberry Pi.

**W**e've taken a deep look at what all went into making the Raspberry Pi the phenomenon it is today. After seven years of ideation and three years of mass consumption one thing is clear - the Raspberry Pi is a force to be reckoned with - by casual and advanced users alike. While many other microcomputer products have arrived since the Raspberry Pi, they have all yet to make as sizeable an impact on the imagination and creativity of users. We devote this chapter to explaining the basic user setup of the Raspberry Pi and taking an overview of the range of projects you can begin experimenting with right away.

## Out of the Box

The Raspberry Pi is a no-frills piece of computer hardware. It is sold as is - which means there are no accessories of any kind in the sold product. The other essentials like an HDMI-compatible display, speakers, mouse, keyboard, micro USB power adaptor and SD storage card need to be either repurposed from present setups or purchased separately. It is recommended that even though only a 2 GB storage card is required, users invest in a 4 GB SD card with a Class 4 or higher grade. The higher class of the cards ensure that speed and stability aren't an issue.

The storage card needs to be formatted so it can be used to install the native operating system from the Raspberry Foundation - Raspbian. There are in fact an assortment of operating systems to choose from which we will discuss in later sections. The default OS can be downloaded at the Rasp-



Raspberry Pi needs a started kit to get it going out of the box.

to set up the hardware of their Raspberry Pi. To accomplish this, a micro USB cable is needed to power the 700 mA at 5 Volts that the card needs. In addition, a USB mouse and keyboard is also required. Depending on the model of the Raspberry Pi that is being used, it may be handy to have a good quality USB hub on stand-by just in case. The Pi computer needs to be connected to a display interface using either the recommended HDMI port and cable, or the analogue video output port. If the display being used has built-in audio hardware then a separate out for audio isn't necessary, however if required the Raspberry Pi comes with a 3.5 mm audio jack output. Only in the Model B range will users have the option for an Ethernet port which

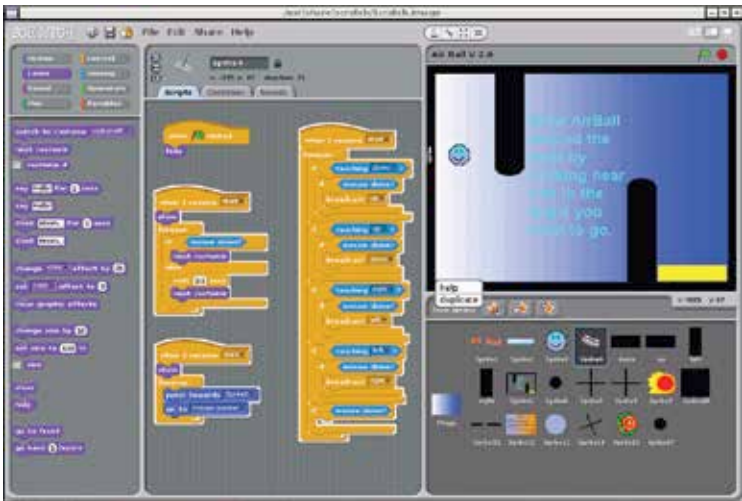
berry Pi website and needs to be installed on the memory card. The installation of the OS on to the memory card is quick and easy on all platforms, you can even download and use NOOBs from the official web site to do all the work for you.

Once the OS has been installed on the card users just need

they can use to connect to their modem. Finally, for those who wish to keep everything polished and good looking, it may be worth investing in a Pi case.

## Talking Pi

The next stage to explore is the programming language of the device. Remember, the Raspberry Pi isn't an appliance like computer which has all the applications pre-programmed, it is a computer learning device. For this purpose it is worth knowing how best to use the programming languages it employs. The Raspberry Pi is supplied with two languages by default - Scratch and Python. But not to worry - Raspbian - like other OSes



Scratch program on the Pi is a great way to learn programming principles.

for the Pi, has a fully functional graphic user interface that will be familiar to Windows and Apple users alike. Users just need to type “StartX” at the command line to launch the desktop view for the first time. The option to toggle the programming language options appear as desktop shortcuts.

Python is a well known programming language by users of all stripes. It initiates with the Python Shell, where commands are read and implemented one line at a time. The language is perfect for scripting purposes since it supports high level commands and data structures. The version of Python can vary depending on the popular default release of your operating system, but there are more than enough development IDEs available

for usage. The icons for Python read as IDLE which stands for Integrated Development Language.

The other language is Scratch which is a very useful beginner language for first time users. Scratch was evolved by the Lifelong Kindergarten Group at MIT as a part of its Media Labs division. The purpose of Scratch is to be the first step in learning programming. The interface of Scratch uses tile-based commands which can be stitched together without any concern over syntax. The tile commands are accessible in related categories which are intuitive and easy to understand. They work on a drag and drop principle, and can be used for multimedia projects as well. There is a vibrant Scratch online share community where projects can be uploaded and shared.

### Garnishing Your Pi

Once the Raspberry Pi is setup and running with its operating system, the next thing to do is to populate it with the necessary applications. All these applications can be found at the Pi Store which can be accessed online with ease directly from the Pi OS. Like any other app store, the Pi



UI on the Pi is familiar and easy to use by anyone.

Store comes with a wide array of apps, games, tutorials, tools and media. Users can even subscribe to the community magazine that tracks and discusses Raspberry Pi related material.

Users are required to create an account to download material from the store, but many options can be run directly from within the store without any need to download. The majority of the

options on the Pi Store are available free, but a few items may be worth the money if you need them. Users are encouraged to share and distribute their unique creations with the community and remixing projects is a common activity. With enough expertise and innovation, users can even consider selling material on the Pi Store by signing up as developers.

### Official OS for the Pi

Thanks to the ARM architecture of the Raspberry Pi's design it can run many different types of operating systems, including the upcoming Windows 10



with the Pi 2. The dominant operating system types for the Raspberry Pi are driven from the open source community and are one form or another of Linux. As most computer users know, Linux is an open source, free-to-use, customisable and powerful operating system. There are hundreds of different variations available for Linux, many of which work on the Raspberry Pi. These versions are called as distributions or distros.

## Raspbian OS

As we mentioned earlier, the default OS for the Pi is called Raspbian which is actually a distro of the Debian OS, which comes bundled with many applications and development tools. It is the ideal OS for first time users of the Pi since it is designed with the Raspberry Pi hardware limitations in mind. This custom fit ensures superior performance and gets the most out of the hardware. It's also convenient to start with since its installation process is intuitive and fast. In addition to which, the bundled packages of applications allow the Raspberry Pi to be good-to-go with all essential tools ready at the time of initialisation. The OS also comes with full documentation and an active online community.

The Raspbian OS was created and developed with the Lightweight X11 Desktop Environment or LXDE and Openbox, as the windows manager, which makes it efficient and smooth. This setup ensures that the OS uses only the minimum resources allowing for faster and smoother performance. At its core, the Raspbian OS remains committed to the core ideals of education by the Raspberry Foundation. This is why the OS also comes with the basic tools like Scratch, which educate and train users in programming. But in function, it is as fully functional as any operating system, making it perfect for a desktop computing solution.

## Arch Linux ARM

The Arch Linux distro is a far more advanced operating system for the Raspberry Pi. It has been in development for over a decade and is brilliant for its efficient use of minimal resources. The elegance of the OSes design and speed are lauded web-wide, such as its less than 10 second boot up rate. The Arch Linux ARM is a port designed specifically for the Raspberry Pi and offers the user full control over the operating system and is very light on the hardware. But with all this control and speed, it sacrifices user friendliness.

In its default state, the Arch Linux ARM doesn't come with a graphical user interface. It presents only a command prompt on booting from which

users are expected to install all the features they require. While this may be a problem for novice users, it is in fact great for advanced users who wish to have things highly customised interfaces or in fact wish to build a personalized operating system from scratch.



The Pi is a great platform for gaming if you have retro tastes.

## RISC OS

RISC OS was the first operating system that was tested on the prototype of the Raspberry Pi. It was designed and released in 1987 in Cambridge. RISC OS and ARM actually share a concurrent development history as well as common team members. In fact, RISC OS was a direct evolution of the operating system that was used on the BBC Micro - the device that inspired the Raspberry Pi. The version used for the Raspberry Pi is called RISC OS Open and is a fairly popular choice.

RISC OS on the Raspberry Pi has a slightly old-school design and layout with a very Windows 95 feel to it. The simple graphic design and artwork scheme actually give the efficient OS a retro styling. The interesting thing here is that



Raspbian OS uses the familiar UI like Windows/Linux/Mac.

it would be a new experience for users of Windows, Linux and Apple, since it is its own system. It isn't based in the popular open source movement which include Linux and Unix but performs just as well on the Raspberry Pi.

## Alternative OS Options

While there are no hard and fast rules about “official” and “unofficial”

systems on the Raspberry Pi, it is clear that some work better than others. The alternative options that we discuss have also been developed with the Raspberry Pi in mind but still haven't found the stability we would expect. Using the operating systems isn't undesirable, some are simply glitchy since they haven't yet been perfected. They are ideal for users who wish to experiment with the operating system interface and wish to learn on their own with no community resources. Due to the relative inexperience of these operating systems there aren't very many guidelines or communities to turn to for help. Definitely not recommended for first time users.

## **Android OS**

Everyone has used Android operating system in one form or another. From their smartphones to tablets to even appliance tools like digital cameras - the Android OS is one of the most popular mobility device systems available. The idea of the Android OS on the Raspberry Pi makes sense given the hardware similarities between the Pi and mobile devices. As a part of the Google family of products, the Android OS is actually a Linux distro designed for mobile systems. It is very versatile and fast with thousands of applications and tools all ready to be used. The Android OS enjoys a vibrant community of developers keep improving and evolving the platform. However, due to its design for smartphones, it can prove to be overwhelming for the Raspberry Pi.

As we discussed in the early chapters - the Raspberry Pi is actually less powerful in hardware terms, when compared to most smartphones and tablets on the market. This makes the task of redesigning the Android OS for a far more light-weight device sort of a challenge. The installation is painless but the stability on the Pi is still uncertain. The development work on the Android OS is dormant or slow at best, with no major push for finding an effective port to the Pi. But it is still an interesting choice in certain multi-Pi board projects which collectively can support the demands of the operating system.

## **Pidora**

The Pidora OS is a reworked Fedora distro for the Raspberry Pi which uses features from Fedora 20. The latest version of Pidora is customised for the ARMv6 architecture, with a ARMv7 version due for release. Pidora has a well designed user interface which works well on the Pi with all its features in tact. Unfortunately, it is not as fast as Raspbian or Arch Linux due



The Pidora OS offers great widgets and interface control.

to its different desktop environment which is based on Xfce instead of LXDE. In addition to which, many applications on the Pidora are originally designed for Fedora which is a fully fledged desktop system and has higher requirements. Overall, it still

lacks the variety of applications that would work well on the Pi and isn't used by the majority of the online community. While worth exploring it isn't likely to be any beginners default choice of operating system.

## Chromium OS

Also designed by Google, the Chromium OS is another Linux distro that is a product of the open source movement. It is also the name of the operating system that is used by Google on their Chromebooks. The last couple of years have seen renewed interest in finding a way to deploy it in Raspberry Pi systems. We only mention it for those extremely advanced users who wish to take on a promising project that has fallen by the wayside. It's a partial work-in-development which was started by web-renowned developer Hexxeh and could use a helping hand.

## Amazing Implementation of the Pi

So, after all the choices and work put into setting up a Raspberry Pi, it is worth wondering - what can it really do? Can something less powerful than a smartphone actually accomplish something as amazing as its popularity would demand? Well, the answer is a resounding yes - from weather stations to game arcades, the Raspberry Pi has proven to be a formidable tool in the hands on an imaginative user. We take a look at some of the most impressive implementations of the Pi so far.

## Pi Arcade

Possibly one of the most fun creations with the Pi is to build a home arcade system. The first of these was made on a large scale using a real life JAMMA cabinet that requires a coin to operate it. The maker used the Multiple Arcade

Machine Emulator application to run ROM games off the Raspberry Pi. The final appearance of the rig was like that of an old school retro gaming arcade and could play a variety of games off its local storage. The arcade machine even had a 32 button joystick between two players for a full on gaming experience. Interestingly enough, other makers have made miniature versions of the arcade Pi game machine that can fit in the palm of your hand.

## Pi Supercomputer

The idea for a Pi supercomputer is fairly intuitive - what would happen if you combined a number of Pi boards together? Would it equal the power of a supercomputer? Well yes and no. So far the largest ever Pi based computer has been using 64 Raspberry Pi modules. This project was conducted at the University of Southampton, where a professor used Lego bricks to house and organise his Pi boards. These 64 boards were connected using their Ethernet ports and used a message passing interface to link up the processors over 32 nodes, allowing for efficient task allocation.



Lego bricks are a great way to set up a multi-Pi system.

The performance boost recorded was linear on the 700 MHz ARM processor per board and a total of 1TB of memory, with remarkable stability, limited only by the quality of the Ethernet network switches used. The official benchmark results of this project indicated a peak performance of 10.13 GFLOPS, which may not seem supercomputer grade today, but its worth remembering that the first supercomputer - the Cray-2 in 1985 clocked only 1.9 GFLOPS at the price of USD 16 million. The Pi supercomputer cost only about USD 3,500.

## Gameboy Pi

Another gaming related project was the creation of a Nintendo Gameboy. The device used a discarded gameboy body to host the Pi and the software.

The device plays emulated games which work surprisingly well on the now refurbished and amped up mobile console. This project is feasible by intermediate users, with the exception of some soldering work, which is necessary to interface between the Gameboy hardware and the Pi hardware. The final product can actually play games from many providers including Sega, Game Gear and NES, as well as Linux based games.

## Be a PiBorg

We are already in the age of wearable computing, although that age is advancing at a very slow pace. We have barely started seeing wearables like watches entering the market, with the highly anticipated visors still



The Pi has already helped make wearable computing prototypes.

some time away. But not in the world of Pi. A wearable Raspberry Pi project was successfully implemented wherein a pair of glasses and a wrist sized keyboard were the only things required. The glasses used were made using the inside part of the MyVu Crystal Glasses, a battery recharger, mini keyboard and trackpad,

and a video cable. The nearly wireless wearable cyborg-esque set up cost less than USD 5,000. Slightly more if you want to go full wireless.

## PiCloud

The need to have a personal cloud is increasing day by day. The idea that all your private content is stored hardwired to your location instead on a Google server can be reassuring, not to mention cheap. The Pi, when using an always active internet connection and external hard disks, can create a personal cloud without any problems. This effectively free storage can easily be scaled up by buying new hard disks, with only the cost of the internet being a constant. This personal cloud allows for all your personal mobile information to have a secure back up, not to mention have access to all your digital content at a moments notice.

## Pi in the Sky

One of the most imaginative and educational experiments done by students - the Pi in the Sky device - used the Pi, a USB camera and a weather balloon to take photographs from high above the Earth. The USB port on the Pi allows for the use of a cheap and easy to




The Pi in the Sky project allowed users perspectives in real time at the lowest costs.

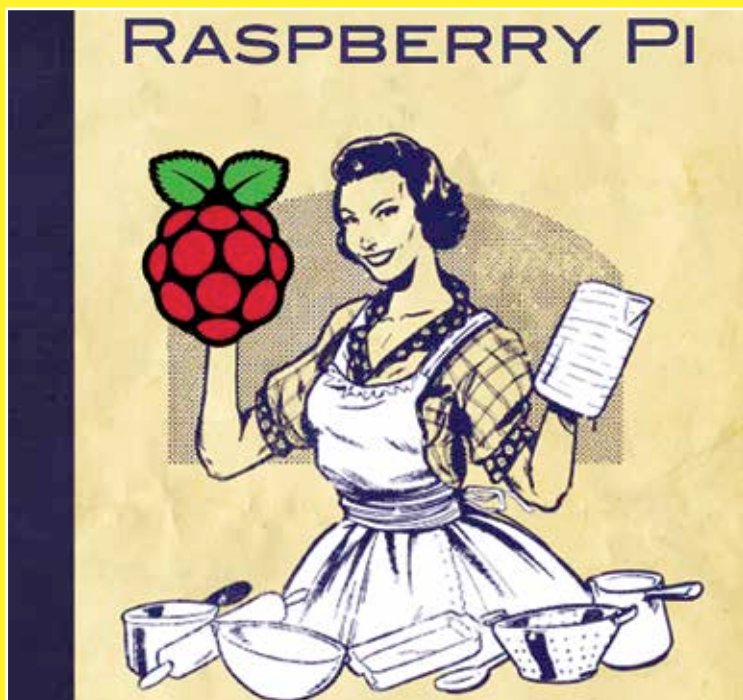
setup webcam, which could then send live images (Slow Scan Digital Video) directly to the receiver. This allowed for a real-time viewing of the atmosphere as well as real time recordings of other weather data.

## Pi Media Center

One of the simplest and most essential projects is to have a home media control centre using the Raspberry Pi. The Pi is connected to the TV using its HDMI port and can produce a much better experience than owning an Apple TV. If users want even more control over their entertainment system they can opt for custom media entertainment based operating systems like RasPlex and OpenELEC on their Pi boards. The media centre allows for remote access to content over WiFi and can be controlled using smartphone apps. The project can even allow direct streaming of online video to the mobile devices. The end result is a tiny and compact replacement for a bulky entertainment centre, at a fraction of the cost.

## Pi Tor Node

As we know, Tor networks help keep users anonymised on the internet. By using relayed network nodes across multiple locations, the user becomes untraceable by any one looking to invade privacy. A simple project uses Tor on the Raspberry Pi to create a relay node which ensures that all of the users internet activities can't be traced back to its origins. 



# BAKING YOUR OWN PI

Now to business. We explain how you can begin your own projects using Raspberry Pi.

**T**his is where the real fun begins. This chapter and the ones following it are devoted to the hands on experience you can have with the Raspberry Pi microcomputer. We begin your exciting journey with a few core tutorials and checklists. In this chapter



we will give a detailed overview of the hardware and a list of necessary tools you will require for the Do-It-Yourself projects we have suggested. As a geek bonus, we also outline how you can create your own operating system on the Raspberry Pi. So brew up a pot of coffee and settle in for a fun and educational ride.

## Hardware Components of the Pi

The Raspberry Pi and the newest Raspberry Pi 2 are fairly similar only in the top range of their models. For the original Raspberry Pi the Model B and Model B+ are at the top of the line, while there is only one of Model B for the Raspberry Pi 2

which has been released so far. There are only two points of difference between the two - the new and improved 900 MHz quad-core ARM Cortex-A7 CPU and the enhanced 1 GB of RAM. Apart for these two upgrades the



The Pi no bigger than a credit card in its form factor.

Raspberry Pi 2 is identical in every respect. Thanks to the new processor the microcomputer can now run advanced operating systems like Snappy Ubuntu Core and Microsoft Windows 10.

The key hardware components for the Raspberry Model B and B+ are the same. They come equipped with a microSD slot, 40 GPIO pins, four USB ports, micro USB port, 3.5 mm audio port, Ethernet port, CSI connection for camera interface, display interface, HDMI port and a VideoCore IV 3D graphics core. All these components come as a part of the System on a Chip design with the Broadcom BCM2835 and BCM2836 for Raspberry Pi and Pi 2 respectively.

## First Time Booting

In earlier chapters we have already discussed how you can easily install an operating system like Raspbian or Pidora onto your Pi. Now we will learn the basic environment in which the Pi user interface exists and how best to make the most of its features.

After you have booted up the Pi to your operating system you will arrive at the Config tool by default. This is the central hub from which you can

select and alter many features with ease. In order to bring up the Config Tool later on all that is needed is the “sudo raspi-config” in the command prompt. You can ensure that this config menu isn’t the default boot up screen by checking the Enable Boot to Desktop option. The next step to when starting up for the first time is to expand the root partition on your storage. Since the default settings don’t give access to the whole memory card you need to go to the Expand Filesystem option. You can use this to gain access to the whole of the memory card’s storage.



The boot interface is easy to navigate and implement.

Next you need to change the default username and password from “pi” and “raspberrypi” respectively to whatever you wish. This is accomplished by using the “Change User Password” guide. You can also

set up Localisation Option such as location, date, timezone etc and custom language and keyboard layouts. Once you are satisfied with all the steps of the start up so far you need to finalise changes and reboot. To do this just click on Finish in the Config Tool and you will be prompted to reboot. Say yes.

Now to connect to the internet you need to get your network details. These can be obtained from your default Windows PC which is on the network and selecting Start>Run>“cmd” - this will bring up the Command Prompt. Here, enter the command “ipconfig” and look at the output - it will display your IP address and Default Gateway details. Note these down as they will be needed for the Raspberry Pi.

Coming back to the Pi you need to open the Terminal window by entering the command: `sudo nano/etc/network/interfaces`. Here change the starting line of “eth0” to “static” in the place of “dhcp”. The last three digits (octet) of the IP address should be different from what you noted down in Windows - and it shouldn’t be used by any other device. Next you need to test your connection between the Windows PC and the Raspberry Pi. To do this you need to start the Command Prompt in Windows and run the command: `ping [Pi IP Address]`. If the connection is correct you will see a reply with no connection timeout errors.

Once the operating system on the Pi has been successfully interfaced with the internet, you can begin with your very first DIY. Brush up on Linux commands and syntax since you'll be using the command line quite often.

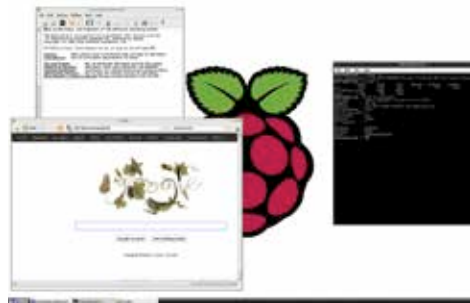
## Geek Cred: OS Tutorial

So you want to earn some legit geek cred? Fair enough. But it won't be easy. For most users who are familiar with core hardware programming and operating systems architecture this section may come across as too basic, but for the yearning tinkerer among us this section is a great way to understand how computer systems are designed and manipulated. We will accomplish these lessons by guiding you through the process of creating your very own operating system for the Raspberry Pi. While this operating system is in no way like graphic rich user interface you may be used to on Linux, Raspbian or Windows, it follows all the same rules as they do. And these rules are what need to be learnt if advanced projects like robotics are what you wish to explore later on.

These instructions are based on educational guidelines from the University of Cambridge's Raspberry Pi development curriculum and intended for mid-teens. But they can also be used by younger, less experienced users with some guidance and help. This section presumes no prior knowledge of programming and computer systems, and is intended for the most basic users. It is assumed that you have already obtained your Pi and the necessary peripherals required to make it functional, along with another active computer which can read/write the SD card. You will need to download the GNU Compiler Toolchain software and the OS Template files which are available at the following URL: [<http://tinyurl.com/yougottalovethepi>]

## What is an Operating System?

We have all used an operating system of one kind or another. From the rich graphical user interface of Windows or Mac, to the simple barebones ones in a digital camera - operating systems are



Coding your own OS leads to infinite possibilities.

in nearly all modern devices that use an integrated circuit. In essence the operating system is just another program, albeit vastly more complicated. The function of the operating system is to organise the rest of the programs used on a computer and manage the hardware resources in relation to those programs.

The operating system is the first boundary of contact between user commands and the hardware that executes them. It allows users to interact and manipulate the hardware to provide desired results. Its important to realise that every click of the mouse and tap of the keyboard is an interaction with the hardware, which we see depicted on the visual display in many appealing styles. The only way an operating system program can interact with the hardware is if they speak the same language, this is accomplished with the use of “drivers”. The concept of the driver was invented so that different operating systems could interface with different types of hardware without any problems. Drivers are nothing but lines of code that can be deployed or removed based from the operating system to alter its interaction with the hardware. These drivers can vary radically from system to system - so we will only discuss the specific ones for Raspberry Pi.

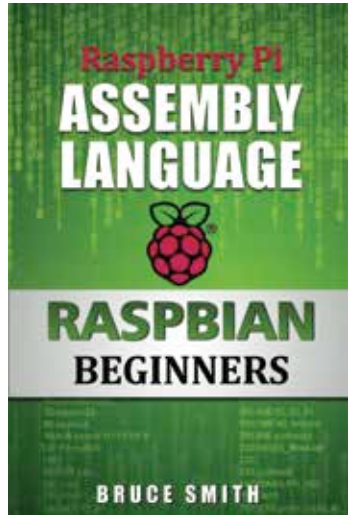
## **What is Assembly Code?**

In any standard computer there is a processor and some memory called RAM (Random Access Memory). The processor is the tool that performs simple functions like math, while the RAM is used to store numbers. The processor is tasked by the programmer to execute a series of instruction and interact with the hardware, this process causes the numbers to be changed in the RAM. These instructions are then translated to readable text for humans thanks to assembly code. Assembly code is the nearest and closest computer language that we can use to instruct the hardware. The reason that it is only an approximate representation of the communication is because of how computers are designed at a fundamental level in a mathematical language.

In usual programming practice, code is written in an assortment of languages such as C, C++, Java and Basic. After the programmer has written code in any of these languages it needs to translated into assembly code. The tool used to accomplish this translation is known as the compiler. The mathematical language of the computer is actually in binary which is very difficult for humans to read. Assembly code on the other hand is more readable but limited in its expressions but cleanly understandable

by computers. Every line of command written in assembly code goes directly to the computer's processor for execution. This is why at its very basic level all these commands are actually very simple in design. Only when we see them all together in millions of line do we feel that it is an alien language.

It is important to know that different processors require different assembly code languages. So the assembly code known for an Intel processor won't work on an ARM processor and vice versa. However, thanks to an operating system we are able to sidestep the hassle of rewriting assembly code for each processor and are able to have it executed and translated via the operating system to its required version. Thankfully, since most operating systems are written in C or C++, with assembly code used only in the most necessary sections.

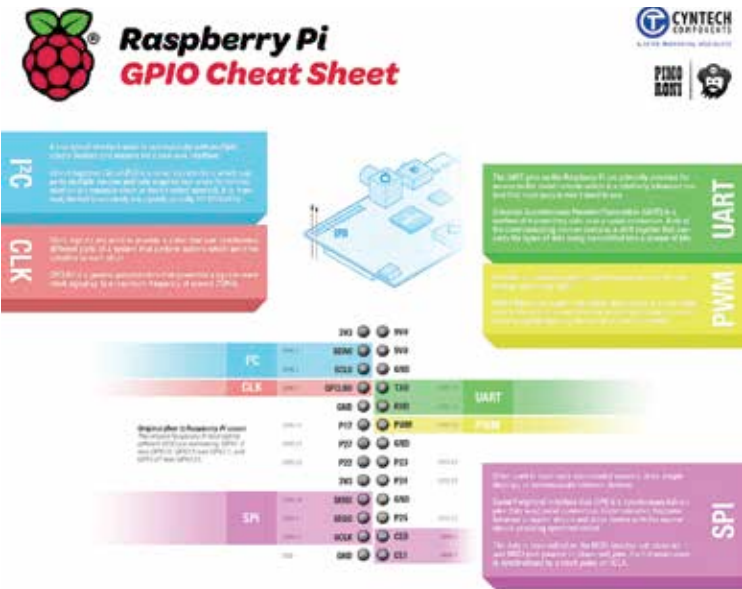


Learning assembly language is a must for advanced projects.

## Baking Your Pi - OS Development

The process of writing your own operating system on the Raspberry Pi is called “baking”. Typically, the Pi is used with a pre-made operating system like Raspbian or Pidora, however, if your project requires a set of custom instructions for the hardware, as in the case of robotics, game emulation or internet relays, then a baked operating system is best. As of now there are hundreds of thousands of different baked operating systems available online which can accomplish many project goals. We will be discussing the process by which you can write a very simple operating system with one simple purpose - to activate a LED light on the Raspberry Pi. The LED is located near the RCA and USB port. This activation operation is known as the “OK” or “ACT” command for the LED because the light was originally labelled as OK and then changed to ACT in the revised boards.

Once you have visited the URL link provided early in the chapter and downloaded the GNU Toolchain and OS Templates we are ready to begin.



Learning to control the hardware features is a must for hardware driven projects.

These downloaded files need to be run on a functional computer other than the Raspberry Pi. The files need to be extracted to a folder using WinRAR or another similar application. A new file needs to be created in the “source” directory called “main.s”. The “.s” file extension is commonly used for all forms of assembly code, in case of the Raspberry Pi it is meant for the ARM processor. The “main.s” file will be used to carry the code for the new operating system. The “build” directory should be empty, the “main.s” should be the only file in the “source” directory and there should be no other additional files in the extracted folder other than “kernel.ld”, LICENSE and Makefile.

### Coding the Compiler

Next you have to open the “main.s” file using a text editor. The assembly code used for the Raspberry Pi and Pi 2 is based on the ARMv6 and ARMv7 processors respectively. The first code in the text editor should be the following:

```
.section .init
.global _start
_start:
```

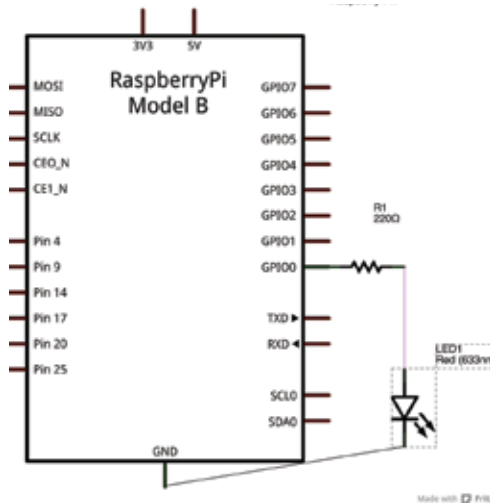
These commands are instructions for the assembler itself so that it translates code correctly. The first line of the code tells the assembler where to place the code itself. The downloaded template you have instructs the code in the section “.init” to be at the beginning of the output. We give this instruction so that the our code is first to be executed, otherwise the default setting would cause the code to be executed in an ascending alphabetical order. The “.section” command informs the assembler which section the code belongs in until we give it a new “.section” location or the code comes to an end. The last two lines are only to stop a warning message and don’t require that much attention.

Once the instructions to the assembler have been coded we now move on to the real programming. All new instructions in assembly code need to be in a new line so that the processor moves on progressively from line to line. The next line of code is:

**ldr r0,=0x20200000**

This is the first instruction to the processor. This line tells the processor to store the number 0x20200000 into the

register r0. In this case, a register is a tiny allocated memory component in the processor where the processor can store the numbers it requires in its active program. There are many registers that have different meanings and purposes which we won’t be discussing now. We are using the simplest of these registers called General Purpose, which range from r0 to r12. They can be used for any calculations required of the processor. We can use any General Purpose register but for sake of simplicity we are using the r0 here. As long as we maintain consistency in our programming it doesn’t matter which register we use.

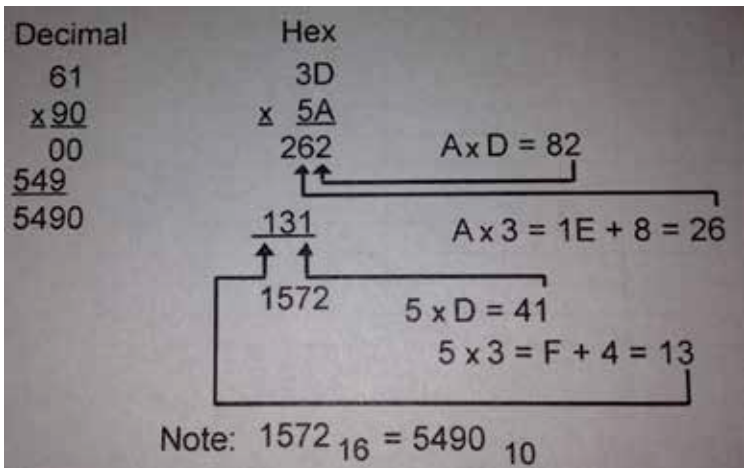


Assembly coding is the way to control the hardware reactions of any processor board.

## Registers and Hexadecimal Addresses

Its interesting to note that a single register can store any integer number between 0 and 4,294,967,295 on the Raspberry Pi. Translated to memory this only makes up 32 binary bits. While 0x20200000 may not appear to be a normal number it very much is, except that it is being written in the hexadecimal notation. The hexadecimal system recalculates the factors of numbers in base 16 and used alphabetic series from letter 'a' through letter 'f' to express six other unique values in hexadecimal form. We typically express numbers in base-10, for example 567 is actually 567<sub>10</sub> which is said as 567 to the base 10 which is the decimal system. In hexadecimal, 567 is written as 237 to the base 16 or 237<sub>16</sub> which can also be written as 0x237. This means that 0x20200000 is 20200000 to the base 16 or 538968064 to the base 10.

Our first command has been to place the number 0x20200000 in the r0 registry. This is the first basic step towards building our single purpose operating system. It is important to realise that computing involves allocating swaths of memory and devices using code. Each of these memory sets and devices has a specific address that is used to identify them. In the world of computers the numbers are just these addresses, and in the case of the Raspberry Pi the number 0x20200000 happens to be the address of the GPIO controller. This address location is decided by the makers of the Raspberry Pi and doesn't have any greater meaning or pattern. The



Understanding numerical expressions is a must to code detailed instructions.



only common feature in these numbers is that they tend to be large well rounded hexadecimal numbers.

## Activating Output

In order to accomplish our objective of activating the LED lights on the Raspberry Pi board we need to send two messages to the GPIO controller. These messages are transmitted using the following commands: `mov`, `lsl` and `str`. For ARM assembly code almost all instructions are written as three letter codes which are mnemonic and hint what the command does. In this case, “`mov`” is for move, “`lsl`” logical shift left, “`str`” is store register and the one we used in an earlier section, “`ldr`” is for load register. The lines of code that need to be added to our ongoing assembler code file are as follows:

```
mov r1,#1
```

```
lsl r1,#18
```

```
str r1,[r0,#4]
```

These commands are used to activate an output to the 16th pin of the GPIO which is wired to the OK/ACT LED. The two main instruction in the code are to get a value into the `r1` register. We could use the “`ldr`” command as we did before but as a matter of programming habit it is wise to deduce the value from a formula rather than enter it directly. To activate the LED pin we need to get the right value in the `r1` registry. The first line is a “`mov`” command and places the number 1 to the base 10 into the `r1` registry. The “`mov`” command is faster than the “`ldr`” command since it doesn’t involve a memory interaction which is what the loading “`ldr`” command does. But keep in mind that the “`mov`” command can only be used to load certain values.

The second line of code uses the “`lsl`” or logical shift left command. This instructs the assembler to shift the binary value representation for the first instruction left by the second instruc-



Every component on the Pi is labeled.

tion. Here the binary representation of 1 to the base 10 is the same as 1 to the base 2 (for binary), and is moved left by 18 places which makes it 100000000000000000 to the base 2 which is 262144 to the base 10. It is helpful to read up and educate yourself on the way binary language works so that you're fluent enough for more advanced projects. We only know these specific values for the LED project due to their documentation in the Raspberry Pi manual.

The storage location of the value in the registry are arrived at by understanding the settings of the GPIO. We know that the GPIO controller has a set of 24 bytes which control the pin. The first four of these bytes are connected to the first 10 GPIO pins, the next four bytes are connected to the next 10 pins and so on, totally 54 GPIO pins, which is six sets of 4 bytes. Each four byte section has three bits related to one of the ten GPIO pins. So in order to arrive at the 16th GPIO pin we need the second set of four bytes which controls pins 10 through 19. And since we need the sixth set of the three bits we arrive at the number 18 (which is 6 times 3) and insert it in the logical shift left instruction. The last line of instruction code therefore is "str" which stores the value from the first instruction in register r1 to the address calculated from the last instructions. This successfully executes the first of our two messages to the GPIO controller telling it to put the 16th pin on stand-by.

## Final Activation

Now that the LED is on stand-by we need to give it the activation command. In the case of the Raspberry Pi the messaging system is slightly inverted, where the off instruction actually activates the LED. This means that as a matter of core design, the LED light turns on when the GPIO pin is turned off. Again, this is a matter of designer preference and follows no particular reason or pattern. The final lines of code are:

```
mov r1,#1
```

```
lsl r1,#16
```

```
str r1,[r0,#40]
```

Using all the familiar commands as before we execute the final instructions. Do not worry about understanding the values ascribed to the above commands - try and puzzle it out based on our discussion of GPIO design, binary language and hexadecimal numbering. The first line puts the number 1 in the r1 registry, the second line shifts the binary value representation of the number 1 in the r1 registry left by 16 places. Since we want the "off"

signal we use the number 1 from the binary code in the 16th pin of the GPIO. This is stored in the registry r1 at the address 40 to the base 10 added to the GPIO controller address. This location happens to be the address which would turn the pin off and activate the light.

## Signing Off Code

Although we have successfully activated the light on the LED the processor still needs to be told the final steps. The processor is designed to be active as long as it is powered and needs to be given instructions on what to do next otherwise it will crash. In order to prevent this we need to give it an infinite task which is accomplished by the following code:

**loop\$:**

**b loop\$**

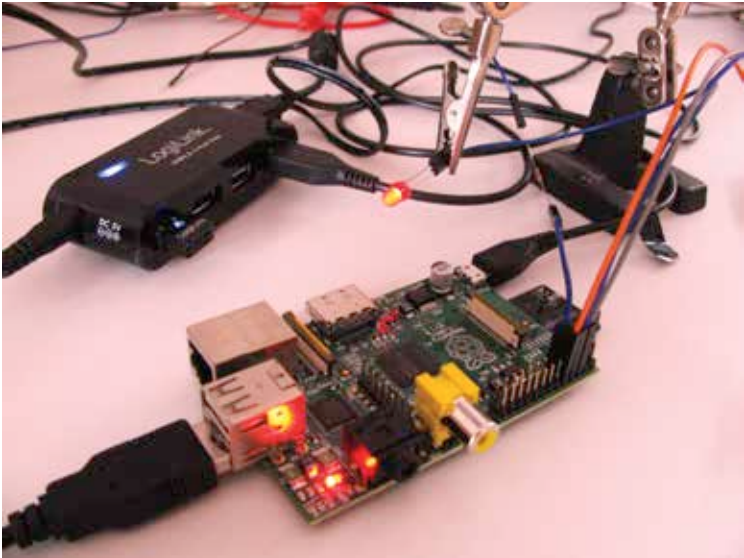
This code isn't a command but a label which names the next line "loop\$" giving the line a name. Labels are used only for code reading purposes by humans and are not converted into binary. As per protocol, the dollar symbol is used to denote labels which are only relevant to the local block of code as opposed to the whole file. We have therefore labeled the first line of code as "loop". The "b" command in the second line or the branch command causes the line of code to be executed to be the one that has the "loop" label instead of any line written after it. This causes the processor to go back to the first line which was named "loop" and the cycle goes on forever in the processor. Keep in mind that the last line in the assembler code should be a blank line since the GNU Toolchain tool expects assembly code files to end in a blank line. Otherwise the software assumes the code isn't finished and it was an accidental gap prompting you for action.

## Loading the OS

Believe it or not, if you have followed all the instruction so far, you have successfully written your first operating system. Now all that remains is to load it on to the Raspberry Pi. From within GNU Toolkit go to the parent directory of the source directory where the code is written. Type "make" and press enter - this should generate three files - kernel.img, kernel.list and kernel.map. The first is the compiled image of your operating system, with the second being the listing of the assembler code and the final one being the map of where all the labels are placed.

To install this image file of your operating system you need to take the SD card for your Raspberry Pi. It is assumed that the SD card in use

already has a version of some other operating system installed on it. The card should be browsed or searched to find the location of the kernel.img which needs to be renamed. It can be named anything as long as you remember it was the original file of the operating system on the SD card. Once renamed you need to copy the kernel.img made using GNU Toolkit in to the directory of the SD card where the original kernel file was located.



The initial set-up can seem messy but is easy to arrange if its well organised.

Presto! The operating system is successfully on the card which just needs to be entered into the Raspberry Pi and activated. The result should be the OK/ACT LED light switching on. Success! Your very first operating system that works!

### **That's It?**

Well no. Not even close. This example was just useful to illustrate the many tools, commands and hardware intricacies that the Raspberry Pi enjoys. If everything in the above section felt easy to understand and execute than you will have no problems exploring and figuring out the Pi. And for general hardware controlled tinkering it is a must that you read the Raspberry Pi manual which explains its hardware design and components in great

details. With the familiarity of assembler coding and the layout of the board's electronics, you can literally command it to execute any command which can be written. This can range from remote controlling a drone to creating a home entertainment system. The only limit is your willingness to learn, develop and innovate coding for the operating system required to bring your dreams to life. 🍌

# DIY ENTERTAINMENT

In this chapter, we'll look at a DIY that will take you back to the good ol' days of gaming or introduce some of you to the games of a bygone era. Other DIYs in this section include smart TVs and stationary exercise bicycles in simulated environments featuring the Raspberry Pi

### Retro Pi

This can also be called the Pi video game, and it will definitely remind some of you of your childhood days that were spent with friends playing Super Mario. The school vacation and classroom chats sometimes filled with words like "I killed the boss/monster in Stage 4" will immediately come rushing back to your mind.

Without further ado, let's have a look at the steps that go into building a Raspberry Pi-based retro video game station.



8-bit retro gaming – Mario

### Hardware needed:

- ◆ Raspberry Pi
- ◆ HDMI cable
- ◆ Keyboard and mouse
- ◆ Game controller – Xbox controller or USB game controller

1. To begin with, you'll have to download the OS images for your version of Raspberry Pi.
2. Download the Win32 disk imager from <http://dgit.in/19BYOYg> to install the downloaded image file from the previous step to the SD card.
3. Next, connect your monitor, keyboard, ethernet cable, game controller and SD card to the Raspberry Pi.
4. Now, power up your Raspberry Pi by connecting the micro USB cable.
5. In the menu that appears, first choose the 'Expand Filesystem' option.
6. Then, choose the 'Finish' option to exit setup and restart.



Win 32 disk Imager to install OS to micro SD card



Raspberry Pi OS start-up configuration

### Install the game controller

When the Raspberry Pi restarts, in the terminal type the following commands:

To install the Xbox game controller, type:

```
sudo apt-get install xboxdrv
```

To install any joystick, type:

```
sudo apt-get install joystick
```

This will complete the part of installing the driver for the Retro Pi.

Now to configure the Xbox controller, type the following command in the terminal:

```
sudo nano /etc/rc.local
```

In the nano text editor that appears in the terminal, scroll down to find “fi”. Between “fi” and “exit 0”, enter the following lines of code:

```
xboxdrv --trigger-as-button -i 0 -l 8 --deadzone 4000 -s
```

The above line of code will add one Xbox controller. To add a second controller, you can use the following command:

```
xboxdrv --trigger-as-button -i 1 -l 9 --deadzone 4000 -s
```

Next, press [Ctrl] + [X] then [Y] and then [Enter].

## Set up the gaming environment

You need to set up the game environment using Git. In the terminal, type

```
sudo apt-get update
```

This will update packages and will take some time to complete.

To install some necessary packages, type:

```
sudo apt-get install git dialog
```

To download the files for emulator installation, type:

```
git clone --depth=0 git://github.com/petrockblog/RetroPie-Setup.git
```

After it's complete, in the terminal, type

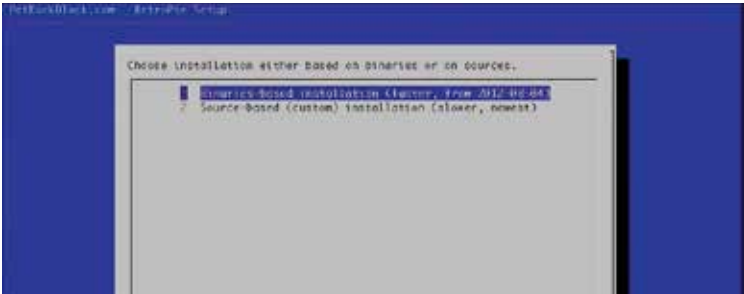
- `cd RetroPie-Setup`
- `chmod +x retropie_setup.sh`
- `sudo ./retropie_setup.sh`

In the screen that appears, choose ‘1] Binaries based installation’

You may choose ‘2] Source based installation’, but it will take many hours to compile and install the files. We, therefore, recommend using binaries-based installation.

Finally, proceed through the screen and select ‘Start EmulationStation on boot?’





Retro Pi game installation set-up screen

## Configure the game pad

Now open the terminal and type the following commands:

- `cd RetroPie/emulators/RetroArch/tools`
- `./retroarch-joyconfig >> ~/RetroPie/configs/all/retroarch.cfg`

Add the following lines at the end to add a 'Game Exit' key:

### For joystick:

- `input_enable_hotkey_btn = "A"`
- `input_exit_emulator_btn = "B"`

### For Xbox:

- `input_enable_hotkey_btn = "X"`
- `input_exit_emulator_btn = "Y"`

Depending on the key on your game controller, you can modify the keys from 'A' and 'B' to some other value.

## Add a second joystick

Now open the file manager and navigate to the following path:

```
RetroPie > configs > all
```

Open the file 'retroarch.cfg' in a text editor. Scroll down until you find lines starting with "input\_player1\_" and some lines like "input\_player1\_joypad\_index = "0" ". Copy all lines containing the text "input\_player1\_" and paste them below. Now in the newly pasted lines, replace all "input\_player1\_" with "input\_player2\_". This will complete the process of adding the second controller.

## Use the ROM

Download the ROM for your favourite games on your PC and copy those files to a USB drive. Open the file manager and copy the ROM files from the USB drive. Navigate to the following path: RetroPie > roms and paste them in the Pi user folder.

Restart your Raspberry Pi and you're ready to go.

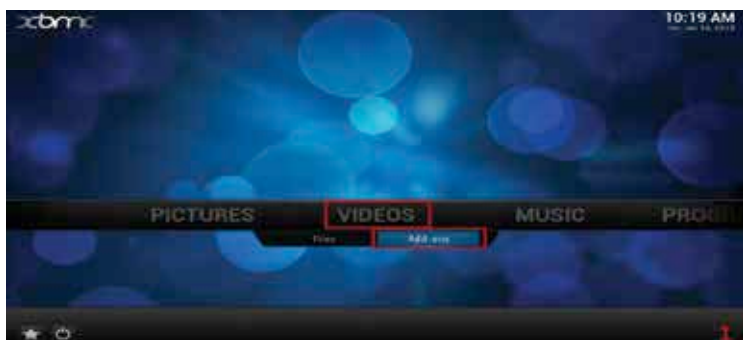
## Raspberry Pi media center

If you don't have a smart TV, you can make one that's not only smarter than any other TV available in the market but also costs substantially less than a smart TV. All you need is a Raspberry Pi to amp up the intelligence of a dumb TV.

Hardware:

- ◆ Raspberry Pi
- ◆ HDMI cable

Check out the DIY instructions at <http://dgit.in/1xAT76K>



Raspberry Pi Media Center screenshot

## Raspberry Pi exercise cycle

If you have an age-old cycle lying around in your house and want to ride it without having to leave your house, the Raspberry Pi-based bicycle is just for you. It has a simulated environment to make stationary exercise fun.


Hardware needed:

- ◆ Old bicycle
- ◆ Raspberry Pi



Riding a Raspberry Pi exercise bicycle

- ◆ Soldering iron
- ◆ Hall effect sensors
- ◆ Wire

This project is very useful for people who can't ride a bicycle outside their homes due to health problems, but yet wish to cycle. 

# DIY EDUCATION

Pick from one of these interesting projects to experiment with the capabilities of the Raspberry Pi

**I**n this chapter, we'll take a look at a few DIYs that simple to create and can be further extended to build much larger DIYs. We won't be covering basics like installing Python since it comes pre-installed in most OSes but we will have a look at extending the Python library for some Raspberry PI specific applications. In the last DIY, we go a step further and repurpose some of the techniques from the first DIY to build a treasure box.

Electronic devices are very easy to control. All you need to do is switch the voltage level on the pin to which the electronic device is connected.

### **What are the possible applications?**

You can switch various electronic devices ON and OFF using this method. These devices include TVs, refrigerators and air conditioners, among other appliances. Using a particular pattern of switching ON and OFF, programmers have also developed a remote control that allows you to change TV channels using the Raspberry Pi.

The increased processing power of the Raspberry Pi 2 as compared to the Raspberry Pi model B+ has broadened its range of capabilities. Despite being cheap, it's a powerful piece of hardware in the hands of a person building a working prototype of an idea.

Let's look at some of the possible do-it-yourself applications using the Raspberry Pi.

## Detect faces using Raspberry Pi

The added processing power in the Raspberry Pi 2 can be very well exploited by image processing algorithms. Image processing algorithms perform complex calculations on an image to recognise various features in an image, and are among the most computationally expensive algorithms.



Raspberry Pi webcam

### How do we do it?

To simplify the process, you can use the PiCam – a camera specially made for use with the Raspberry Pi – using the guide at <http://dgit.in/1HOGcUb>. Or you can connect a USB web camera available in market. For the purpose of this DIY, we'll be using a USB web camera.

To use a USB webcam, you'll need to install the `fswebcam` package from the Raspberry Pi repository. To install the same, open the terminal and enter the following commands:

- `sudo apt-get update`
- `sudo apt-get install fswebcam`

To check if your webcam works correctly, use the following command:

- `fswebcam image.jpg`

You can specify the resolution of the image using the `-r` flag along with the required resolution as:

- `fswebcam -r 1280x720 image2.jpg`

Before we continue, we need to install the OpenCV (Open Computer Vision) library that can help detect faces. To install the library, open the terminal and enter the following commands:

- `sudo apt-get install libopencv-dev`

- `sudo apt-get install Python-opencv`

If you're unable to install the OpenCV library, you can check the source installation method at <http://dgit.in/1FA6TNu>

Create a Python script that will help detect your face from the webcam input. Open the terminal and type:

- `nano face_detect.py`



Famous Lemma image for face detection

Next, enter the following code in the editor that appears on the screen.

- `import os`
- `import`
- `cv2cv2.NamedWindow('image')`
- `while True:`
- `os.system('fswebcam -r 1024x768 -S 3 --jpeg 50 --save /home/pi/Desktop/image.jpg')`
- `frame = cv2.imread("/home/pi/Desktop/image.jpg")`
- `gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)`
- `faces = faceCascade.detectMultiScale`
- `(gray,`
- `scaleFactor=1.1,`
- `min Neighbors=5, minSize=(10, 10),`
- `flags = cv2.cv.CV_HAAR_SCALE_IMAGE )`
- `for (x, y, w, h) in faces:`
- `cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 79, 255), 2)`
- `cv2.ShowImage('image',frame)`
- `k=cv2.WaitKey(0);`
- `if (k == 'ESC'):`
- `break`
- `cv2.DestroyWindow('image')`

After entering the code, press [Ctrl] + [X] and then [Y] and finally the [Enter] key. This will save the code.

Code	Explanation
<code>import os import cv2</code>	Imports the required libraries.
<code>cv2.NamedWindow('image')</code>	Opens a window to show the image.
<code>while True:</code>	
<code>os.system('fswebcam -r 1024x768 -S 3 --jpeg 50 --save / home/pi/Desktop/image.jpg')</code>	Reads the image using the fswebcam utility and saves it to the desktop as image.jpg.
<code>frame = cv2.imread("/home/pi/Desktop/ image.jpg")</code>	Loads the image captured from the webcam.
<code>gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)</code>	Converts the image to grayscale.
<code>faces = faceCascade.detectMultiScale(     gray,     scaleFactor=1.1,     minNeighbors=5,     minSize=(10, 10),     flags = cv2.cv.CV_HAAR_SCALE_IMAGE )</code>	Detects all faces in the image.
<code>for (x, y, w, h) in faces: cv2.rectangle (frame, (x, y), (x+w, y+h), (0, 79, 255), 2)</code>	For every face detected, it draws a light blue border to show every face detected.
<code>cv2.ShowImage('image',frame)</code>	Displays the image.
<code>k=cv2.WaitKey(0);</code>	Waits to see if the user presses a key.
<code>if(k == 'ESC'): break</code>	Exits the loop if the user presses the [Escape] key.
<code>cv2.DestroyWindow('image')</code>	Closes the window.

Next, in the terminal, type:

- `Python face _ detect.py`

This should open a window that detects faces.

## Get an LED to blink

In this section, to get started, we'll use a simple circuit that can switch an LED ON or OFF. You can take the help of an electronics expert to extend this project to turn appliances on and off in your home.

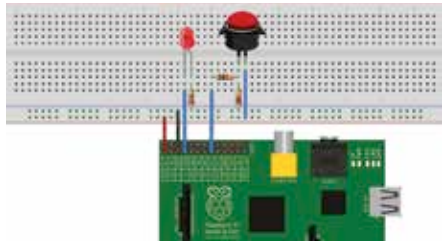
The Raspberry Pi has pins called ‘GPIO [General Purpose Input Output] pins’ that allow switching ON and OFF. To use the GPIO pins, you need to install the GPIO Python library that allows you to switch the voltage level on a particular GPIO pin. This change in voltage level can switch appliances ON or OFF by using relays.

#### Software needed:

To control the voltage of the GPIO pin, you’ll need to install the GPIO library that allows voltage control using the Python programming language.

To install the GPIO Python library, open the terminal and type the following commands:

```
• sudo apt-get install Python-dev
• sudo apt-get
install Python-rpi.
gpio
```



#### Hardware required:

- ◆ Raspberry Pi
- ◆ Breadboard
- ◆ Wires
- ◆ LED
- ◆ 1K resistor – Three     Circuit for blinking LED in number
- ◆ Switch

Connect the circuit as shown in the given figure. If you’re new to electronics, we advise taking the help of someone with some knowledge of electronic circuits.

For the first part, we’ll just switch the LED ON and OFF alternatively.

We’ll write the Python code for the same as follows:

```
• import RPi.GPIO as GPIO
• import time

• GPIO.setmode(GPIO.BCM)
• GPIO.setup(4, GPIO.OUT)
• while True:
```



- `GPIO.output(4,True)`
- `time.sleep(1)`
- `GPIO.output(4,False)`
- `time.sleep(1)`

Code	Explanation
<code>import RPi.GPIO as GPIO</code> <code>import time</code>	The GPIO library allows use of GPIO pins. The Time library has the functionality of waiting for a specified amount of time.
<code>GPIO.setmode(GPIO.BCM)</code> <code>GPIO.setup(4, GPIO.OUT)</code>	Sets pin 4 on the Raspberry Pi as an output pin, since it will output a voltage that will switch the LED ON and OFF.
<code>while True:</code>	We want the LED blinking forever, so we use the 'while' loop with the condition to be 'True' – a hack used by the electronic device for running programs forever without halting.
<code>GPIO.output(4,True)</code>	This command switches ON the LED by giving a high voltage at pin 4.
<code>time.sleep(1)</code>	Waits for one second.
<code>GPIO.output(4,False)</code>	This command switches OFF the LED by giving a low voltage at pin 4.
<code>time.sleep(1)</code>	Waits for one second.

The code does the following things:

- ◆ Switches ON the LED
- ◆ Waits for 1 second
- ◆ Switches OFF the LED
- ◆ Waits for 1 second
- ◆ Repeats again from Step 1

Once the entire process is complete, the program will jump to the beginning of the 'while' loop and restarts the process from the 'GPIO.output(4,True)' command.

**Code 2:** This code reads the input from the switch connected at pin 22 and switches the LED ON and OFF accordingly.

- `import RPi.GPIO as GPIO`
- `import time`
- `GPIO.setmode(GPIO.BCM)`
- `GPIO.setup(4, GPIO.OUT)`
- `GPIO.setup(22,GPIO.IN)`
- `while True:`
- `c = GPIO.input(22)`
- `GPIO.output(4,c)`
- `time.sleep(1)`

Command	Function
<code>GPIO.setup(22,GPIO.IN)</code>	Sets pin 22 on the Raspberry Pi as an input pin, since it will read the voltage at the switch to check if it has been pressed.
<code>c = GPIO.input(22)</code>	Reads the input on pin 22 and stores this in the variable 'c' to check if the switch has been pressed.
<code>GPIO.output(4,c)</code>	The data in variable 'c' contains the state of the switch i.e. informs whether it has been pressed or not. This command switches the LED on pin 4 ON or OFF depending on the state of the switch.

These snippets of code will look at the incremental change over the previous piece of code.

## Translate with Speech Recognition and Playback

You don't have to rely on your phone to translate from one language to another anymore; you can build a translator for yourself using the Raspberry Pi.

Hardware required:

- ◆ Raspberry Pi
- ◆ USB headset

This DIY uses the Microsoft Translator service to convert speech to text,

however, you could also use the Google Translate service. Check the DIY instructions at <http://dgit.in/1beD7mu>.



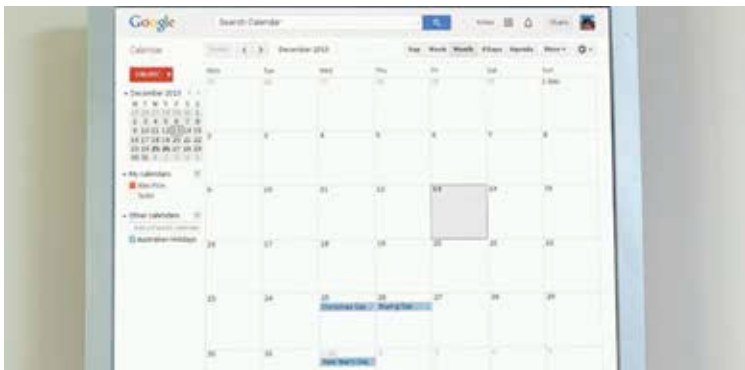
Carry your translator along with you on your travels!

## Wall-Mount Google Calendar

The calendar is the go-to tool for every productive person looking to schedule their daily activities. This DIY uses the Google Calendar service as your daily calendar and enables you to see which task is coming up next. Since Google Calendar also has a built-in reminder, it will remind you of the next task/meeting in your planned schedule. This is especially good for checking the availability of meeting rooms and knowing whether they've been booked at a given date and time.

Hardware needed:

- ◆ Home network
- ◆ Raspberry Pi



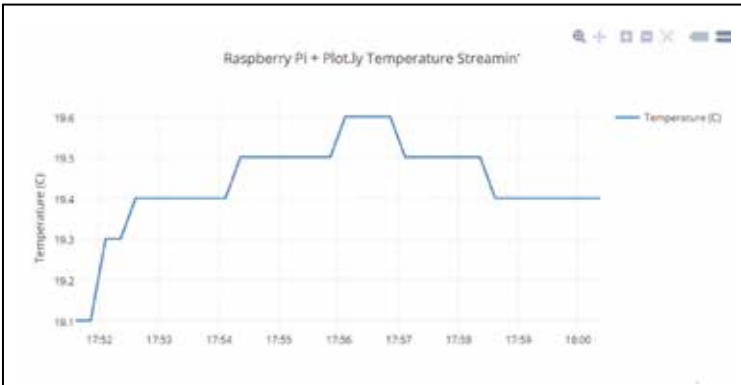
Interactive Calendar using Raspberry Pi

- ◆ AC adaptor
- ◆ USB keyboard and mouse
- ◆ HDMI cable
- ◆ Wall mountable HDMI capable monitor
- ◆ Wall bracket for your monitor

Check the DIY instructions at <http://dgit.in/1xdGHHw>.

## Visualise data with real-time graphs

Let's say you want to keep a log of your home temperature or the temperature of the chemical reaction that you kept overnight at the lab. It can be an intimidating task to look at numbers displaying the time and temperature to check for erratic behaviour. The best way to visualise such data is by looking at a graph of the data. This DIY records the data available at the GPIO pin and passes it to an online graph plotting service called plot.ly. So now, you can check the temperature at your home or the temperature of that chemical reaction you left at the lab from any place in the world.



Real-time web graph using 'Plotly'

### Hardware needed:

- ◆ Raspberry Pi
- ◆ MCP3008 DIP-package ADC converter chip
- ◆ Analog Temperature Sensor TMP-36
- ◆ Breadboard
- ◆ Wire

Check the DIY at <http://dgit.in/1xdGGmU>



Internet radio using Raspberry Pi and Arduino

## Create an internet radio player

Online radio stations are a good option when you're bored of FM channels playing the same music over the past few days. However, its alternative, playing music from your PC, isn't very feasible either since you'll have to keep your PC switched on just to play music. This DIY helps play music running on a particular online radio station.

- ◆ Arduino UNO
- ◆ LCD/Keyboard shield
- ◆ Raspberry Pi
- ◆ USB and Ethernet cables

By default, this DIY uses the Radio Paradise radio station, but it does support RTE Radio, Monkey Radio, The Smooth Lounge, Radio Nova and Newstalk.

To add your favourite radio station, all you have to do is put in its URL. Check out this DIY that can help you create a Internet streaming radio at <http://dgit.in/1BORTZO>.

## Install a server at home

If you wish to install a server at home that can act as a web server running a full Linux desktop or supporting Apache, MySQL, PHP stack, SAMBA file-sharing network or port forwarding, but don't wish to dedicate a power-



A Raspberry Pi home server

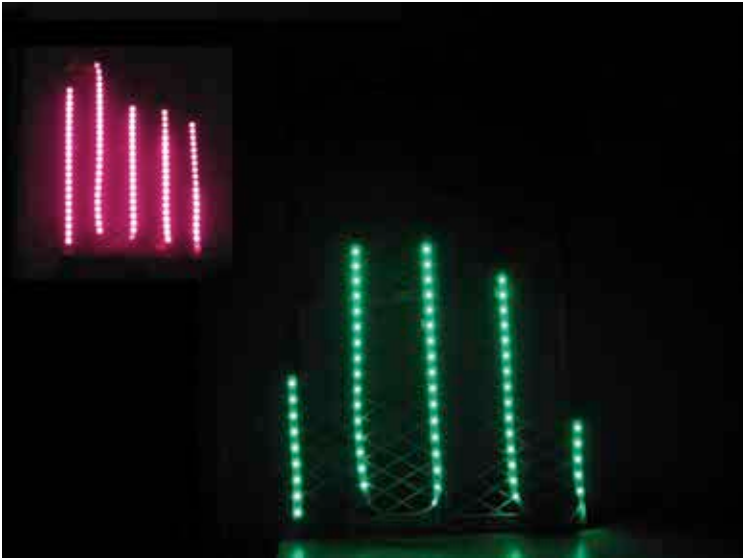
hungry PC for the same, you can use the Raspberry Pi to run a server. So it's definitely a great option for web designers who want to develop a website from any place around the world and yet keep the code for it safe on their device.

These are the possible uses of the server:

- ◆ A web server
- ◆ SQL instance
- ◆ Custom cloud network
- ◆ Managing users for the cloud network
- ◆ SAMBA file-sharing network, integrated into Windows-based home or office network
- ◆ Understanding the principle of the port forwarding function on a home router
- ◆ A web interfaced bit torrent manager

Hardware needed:

- ◆ Raspberry Pi
- ◆ External USB 2.5" HDD
- ◆ Raspberry Pi case of your choice
- ◆ Ethernet cable



Spectrum Analyser using Raspberry Pi

- ◆ USB cable for the hard drive
- ◆ SD card
- ◆ Relevant power supplies for the Pi and HDD

Check out the DIY instructions at <http://dgit.in/1Gngjuq>.

## Create a spectrum analyser with RGB LED strip and Python

You might have seen a spectrum analyser on your desktop media player showing the bars of varying intensity at various frequencies for audio tracks. A DJ uses it to analyse various components in the sound and change the output, so as to create a great music experience when you're at a club. This DIY is an attempt to create a spectrum analyser using LED lights.

Hardware needed:

- ◆ Raspberry Pi
- ◆ RGB LED strip
- ◆ 10A 5V power supply to drive the LED
- ◆ HDMI monitor and cable

Check out the DIY instructions at <http://dgit.in/1ADNKoC>.



Create a treasure box using Raspberry Pi


## Secure your treasure box with face recognition

An extension to the face detection earlier mentioned is face recognition. This tutorial will enable the device to precisely identify you using a connected webcam. This DIY will help you build a basic treasure box or safe that recognises your face. A Raspberry Pi computer connected to the webcam captures the image. When you appear in front of the webcam it detects your face and opens the treasure box that's keeping all your belongings safe.

### Hardware needed:

- ◆ Raspberry Pi
- ◆ Raspberry Pi camera
- ◆ Servo or lock solenoid for the locking mechanism
- ◆ 10 kilo-ohm resistor
- ◆ Power supply for the system
- ◆ Wood, wood glue and fasteners for building a latch mechanism and frame
- ◆ Wires

It can also capture the face of the person who tries to open the box without your knowledge. Beware, however, that someone can also use your photograph to open the box.

Check the DIY instructions at <http://dgit.in/1F2IT5G>. 



# DIY HOME AUTOMATION

Learn to automate your home with this bunch of DIY projects that will help secure your home while saving electricity

**W**ashing machines started the automation process in our homes by automatically washing clothes. With time, our houses will only keep getting better. Why not try to develop our ideal future home today? Raspberry Pi will let you automate many more things in your house. It offers the required level of flexibility to kickstart the development of future homes. Let's try to peek into some of the possibilities.

Home automation is not just about automating the menial tasks in your house. It also involves optimising the home to automatically make it adjust to the user's need while saving electricity, water and many other resources without disturbing the residents in that house.



Automating and controlling homes remotely

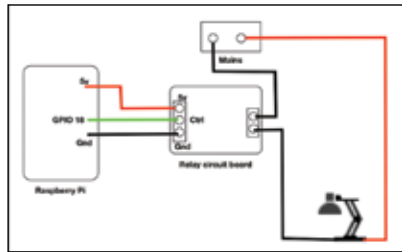
To start with, let's try to control the lights and move further to add more complexity.

## Remotely control light

Turning on the lights and turning them back off are things we do throughout the day. Many a time, we forget to switch off electronic appliances when we leave home. It's all good until you forget to switch off your air conditioner or television set, which can effortlessly waste electricity and add to light bill costs. For the purpose of this DIY, we'll look at a simple circuit and code that will help you switch a lamp on and off. This is an extension to the LED blinking program that we saw in the previous chapter.

Hardware needed:

- ◆ Raspberry Pi
- ◆ Relay circuit
- ◆ Lamp for circuit testing



Basic circuit for switching appliances ON and OFF

If you're a beginner, we recommend using a relay circuit from the market. You could also build your own relay circuit using a motor driver or a transistor. In case you want to build your own relay circuit, please take the help of a friend who has adequate knowledge about electronics. Alternatively, you can follow an online tutorial such as the one available at <http://dgit.in/1O98eyK>.

Be careful when building the relay circuit or connecting the relay circuit. A wrong connection may damage your Raspberry Pi board, harm you or switch off the lights of your house. Please connect with caution and under expert supervision.

First, install the Python GPIO wrapper by using the following command in the terminal:

- `sudo apt-get install rpi.gpio`

Connect the circuit as shown in the figure. Now, in the terminal enter the following command to create a Python script that will control the appliances:

- `nano basic_switch.py`

In the new text editor that appears in the terminal, type the following code:

- `import RPi.GPIO as GPIO`
- `import time`
- `GPIO.setmode(GPIO.BCM)`
- `GPIO.setup(18, GPIO.OUT)`
- `GPIO.output(18, True)`
- `time.sleep(120)`
- `GPIO.output(18, False)`

Let's analyse the code.

Code	Explanation
<code>import RPi.GPIO as GPIO</code>	Imports the GPIO library.
<code>import time</code>	Imports the time library that contains the 'sleep' function.
<code>GPIO.setmode(GPIO.BCM)</code> <code>GPIO.setup(17, GPIO.OUT)</code>	Uses pin 17 as an output pin.
<code>GPIO.output(17, True)</code>	Turns ON the lamp.
<code>time.sleep(120)</code>	Waits for 120 seconds or 2 minutes.
<code>GPIO.output(17, False)</code>	Turns OFF the lamp.

Now that you're able to control a lamp, let's try to control it from a web interface, so that you can control the light from some remote location using the 'GET' interface. The circuit for this piece of code remains the same; all you need to do is add the server parameters.

We'll use the Twisted web server to run this piece of code. Let's start by installing the web server. Open the terminal and type the following command:

- `sudo apt-get install python-twisted`
- `from twisted.web.server import Site`
- `from twisted.web.resource import Resource`

```

• from twisted.internet import reactor

• import RPi.GPIO as GPIO
• GPIO.setmode(GPIO.BCM)
• GPIO.setmode(18, GPIO.OUT)

• class lampAPI(Resource):
•     def render_GET(self, request):
•         if 'bulb_status' in request.args:
•             if request.args['bulb_status'][0]
== "off":
•                 GPIO.output(18, False)
•                 return "Bulb has been
turned OFF"
•             if request.args['bulb_status'][0]
== "on":
•                 GPIO.output(18, True)
•                 return "Bulb has been
turned ON"

• root = Resource()
• root.putChild("interface", lampAPI())
• factory = Site(root)
• reactor.listenTCP(80, factory)
• reactor.run()

```

Now press [Ctrl] + [X] together, then [Y] and then press the [Enter] key to save the file.

Code	Explanation
<pre> from twisted.web.server import Site from twisted.web.resource import Resource from twisted.internet import reactor </pre>	Imports the Twisted Python server libraries.
<pre> import RPi.GPIO as GPIO GPIO.setmode(GPIO.BCM)  GPIO.setmode(18, GPIO.OUT) </pre>	Imports the GPIO libraries which are like the code for LED blinking.

<code>class lampAPI(Resource):</code>	Define a new class
<code>def render_GET(self, request):</code>	Define render_GET function
<code>if 'bulb_status' in request.args:</code>	Checks if a variable 'bulb_status' exists in the received URL.
<code>if request.args['bulb_status'][0] == "off":</code>	Checks if the parameter for the 'bulb_status' variable has been set to 'off'.
<code>GPIO.output(18, False)</code>	Switches OFF the light if the user sends the 'off' parameter.
<code>return "Bulb has been turned OFF"</code>	Informs the user that the bulb has been switched OFF.
<code>if request.args['bulb_status'][0] == "on":</code>	Checks if the parameter for the 'bulb_status' variable has been set to 'on'
<code>GPIO.output(18, True)</code>	Switches ON the light if the user sends the 'on' parameter.
<code>return "Bulb has been turned ON"</code>	Informs the user that the bulb has been switched ON.
<code>root = File("myserver")</code>	Specifies the server files storage folder as 'myserver'.
<code>root.putChild("pi_auto", lampAPI())</code>	Creates a child of resource to handle requests.
<code>serve = Site(root)</code>	Initialises the 'twisted' object.
<code>reactor.listenTCP(80, serve)</code>	Sets the server to handle request on port 80.
<code>reactor.run()</code>	Keeps listening on port 80 forever until turned OFF manually.

To turn ON and turn OFF, simply type the following URL in the browser.

To turn ON the light:

[http://<ip-address>/pi\\_auto?bulb\\_status=on](http://<ip-address>/pi_auto?bulb_status=on)

To turn OFF the light:

[http://<ip-address>/pi\\_auto?bulb\\_status=off](http://<ip-address>/pi_auto?bulb_status=off)

Now type the following command in the terminal:

- `mkdir myserver`

This will create a directory with the name 'myserver' as specified in the program with 'res\_var = File("myserver")'. Now, in terminal, type the following to change the current directory to the newly created folder 'myserver':

- `cd myserver`

In the newly created 'myserver' directory, create an HTML file that will be accessed by the user to turn ON or OFF the lights by merely entering:

- `nano index.html`

Now type the following code in the text editor to create a webpage:

- `<!DOCTYPE HTML>`
- `<html>`
- `<head>`
- `<title>Light control using Raspberry Pi</title>`
- `</head>`
- `<body>`
- `<h1> Bulb control </h1> <br>`
- `<a href="http://<ip-address>/pi_auto?bulb_status=on">`
- `<h2> Turn ON </h2> </a> <a href="http://<ip-address>/pi_`
- `auto?bulb_status=off"> <h2> Turn OFF </h2> </a>`
- `</body>`
- `</html>`

Now press the [Ctrl] + [X] keys together, then the [Y] key and then press [Enter] to save the file.

Now enter the following command to run the light control Python script:

```
python basic_switch.py
```

## Bulb control

[Turn On](#)

[Turn Off](#)

After writing this program you can access the webpage using the IP address of the Raspberry Pi in the browser (Chrome/Firefox) as:

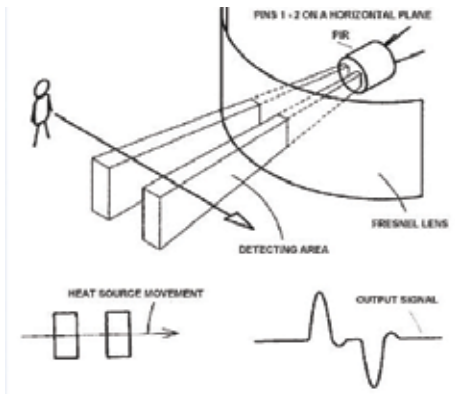
<http://<ip-address>/>

This will open a webpage with the option to turn ON and turn OFF the light as shown in the screenshot.

## Detect intruders

When you're out of your house, an intruder may try to gain entry into it. You don't need to install a camera to constantly monitor for break-ins. Leave the task to sensors called 'PIR sensors', which we'll be using in a while to make an intruder detector. Sometimes called 'PID' (Passive Infrared Detector), the PIR sensor works by detecting infrared rays. You might have heard of or seen an infrared camera that shows the body of a human in red, blue and green colour. The camera is able to capture this because the human is emitting infrared rays, which is, in fact, heat that a human body radiates.

Let's see how a PIR sensor works. Refer to the image showing the working of a PIR sensor.

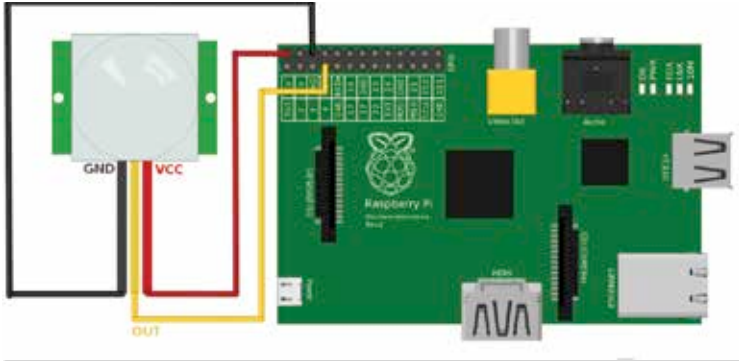


Working of a PIR sensor

Every PIR sensor has two slots in it, each containing an infrared radiation detector. A special lens called 'Fresnel' lens (that's used in lighthouses near coasts) is placed in front of the two sensors to increase its area of reach. In the absence of a human, both sensors detect the same amount of infrared radiation. When a human passes in front of the sensor, the first sensor detects heat and the second doesn't detect any heat, leading to a difference in output of the two sensors. When the human moves to the other side, the second sensor detects heat, but the first doesn't detect any heat then. Thus, from the change in heat patterns, a PIR sensor detects the presence of a human. To know how PIR sensors work in detail, visit this link: <http://dgit.in/1H2Q64b>.

**Hardware needed:**

- ◆ Raspberry Pi
- ◆ PIR sensor
- ◆ Wires for connection



PIR sensor to Raspberry Pi connection

Connect the circuit as shown in the circuit diagram. Now, open the terminal and enter the following command in the terminal to create the Python script that will detect an intruder:

- `nano intruder _ detect.py`
- `import RPi.GPIO as GPIO`
- `import time`
- `GPIO.setmode(GPIO.BCM)`
- `GPIO.setup(7, GPIO.IN)`
- `print ("CTRL+C to exit")`
- `print ("Intruder Detector Ready")`
- `while True:`
- `if GPIO.input(7):`
- `print ("Motion Detected!")`
- `time.sleep(10)`

Now press [Ctrl] + [X] together, then the [Y] key and then press [Enter] to save the file.



Let's look at the explanation of the code.

Code	Explanation
<code>import RPi.GPIO as GPIO</code>	Imports the GPIO library for using the GPIO pins.
<code>import time</code>	Imports the time library for using the sleep function.
<code>GPIO.setmode(GPIO.BCM)</code>	
<code>GPIO.setup(7, GPIO.IN)</code>	Uses pin 7 on Raspberry Pi for PIR sensor detection.
<code>print ("CTRL+C to exit")</code>	Instructs the user to press [CTRL]+[C] to exit.
<code>print ("Intruder Detector Ready")</code>	Informs the user that the module is ready for operation.
<code>while True:</code>	Checks forever to see if there's an intruder.
<code>if GPIO.input(7):</code>	Checks if heat signature is detected by the PIR sensor.
<code>print ("Motion Detected!")</code>	If a heat signature is detected by the PIR sensor, it warns the user.
<code>time.sleep(10)</code>	Waits for 10 seconds

Now execute the Python script for intruder detection. On detecting an intruder, it will display a warning for the same.



Hook up a flashlight to the camera and you could catch that sneaky animal that's been prowling in your backyard.

What more, you can extend the above code to take a picture of the intruder, using the `fswebcam` or `picam` utility mentioned in the previous chapter. All you need to do is change the 'while' loop, after you've connected the webcam and installed the `fswebcam` package (if you're using a regular camera) or the `picam` package (if you're using the official Raspberry Pi camera).

The Python code for the same to take a photo of the intruder will be as follows:

```
• import RPi.GPIO as GPIO
• import time
• GPIO.setmode(GPIO.BCM)
• GPIO.setup(7, GPIO.IN)
• print ("CTRL+C to exit")
• print ("Intruder Detector Ready")
• while True:
•     if GPIO.input(7):
•         print ("Motion Detected!")
•         filename=$(date -u
+ "%d%m%Y_%H%M%S").jpg
•         fswebcam -d /dev/video0 -r
640x480 $filename
•         time.sleep(10)
```

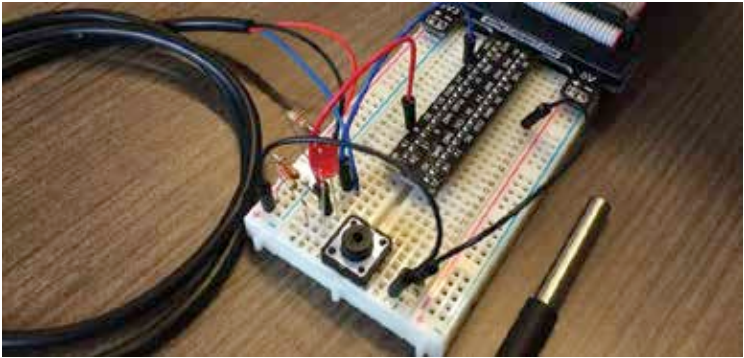
Thus, using the above code, we'll be able to take a photo of the intruder along with the time of the photo capture.

Code	Explanation
<code>filename=\$(date -u + "%d%m%Y_%H%M%S").jpg</code>	Creates the name of the image file using the date command in the format 'Date-month-year_hour-minutes-seconds'
<code>fswebcam -r 1024x768 \$filename</code>	Saves the image with the given file name.

What we saw above are some basic projects that can help you get started. There are many more home automation projects that can be created with the Raspberry Pi.

## Stream your home temperature

This is a simple project that will only stream the current temperature of your home using the Raspberry Pi.



Streaming temperature circuit for Raspberry Pi

Hardware needed:

- ◆ Raspberry Pi
- ◆ DS18B20 temperature sensor
- ◆ Breadboard
- ◆ 4.7K to 10K resistor
- ◆ Wires

To check out the DIY, head over to <http://dgit.in/1x40Y1M>

## Remotely open your garage door

People owning a car will want their garage door to be opened without getting down from their car when they reach home. This DIY does exactly that. After implementing this, all you have to do is enter the IP address of the Raspberry Pi in the browser to open the garage door via the web interface.


Hardware needed:

- ◆ Raspberry Pi
- ◆ BreadBoard
- ◆ Wires
- ◆ 5v Relay
- ◆ Transistor
- ◆ Diode
- ◆ 1K resistor
- ◆ Bell Wire (or similar)

Check out the interesting DIY at <http://dgit.in/1x41blL>



Garage door opening interface

You could also use this concept to lock the door of your personal room from possible intruders and alert you if someone has opened it. Also, you could take it further to take a pic of the intruder as in the intruder detection DIY. To sum up this chapter, we've seen a few DIY projects that can help you get started with automating your home. We began by telling you about how to control a light source remotely and went on to talking about intruder detectors and eventually spoke about opening doors or closing them using the Raspberry Pi. With so much automation in place, you could easily save some electricity and feel safe even when you're out of your home. Good luck automating your home. 

# DIY ENVIRONMENT

Learn how to create various Raspberry Pi based projects that will help monitor and manipulate the environment around us

Everyday, we humans damage the environment by indirectly harming all plants, animals and birds. As inhabitants of this planet, we need to focus on sustainable living and minimise our negative influence on the environment around us. Every pollution control act is just useless law, unless followed by citizens judiciously. To scientifically study our impact on the environment, we need to attach a numeric value to it. By measuring quantities, we can therefore measure our impact on the planet's ecosystem. This can help us choose better options to reduce our negative impact on the environment. In this chapter, we'll look at a few DIY projects that will help you monitor various parameters of the environment like temperature, humidity to name a few.

To preserve our environment, we need to plant more trees and take care of these plantations. Part of taking care of plants is ensuring that they're watered on time. Trees need an adequate amount of water before they establish their roots in the ground. So let's start the first DIY with a plant watering system.

## Control a plant watering system using the internet

In this DIY, we'll make a web-enabled plant watering system. This will allow you to water the plants in your home, building or even colony while sitting in another part of the world. For this purpose, we'll create a web server that will listen to a request for switching the motor ON and OFF.

Hardware needed:

- ◆ Raspberry Pi
- ◆ Breadboard
- ◆ Water pump / Solenoid valve
- ◆ Solid State Relay
- ◆ N3904 Transistor
- ◆ 4.7 k ohm Resistor

The above mentioned list includes all the electronic parts of the complete setup. You'll also need the required piped connections for distributing water.

Let's get started by adding logic to the Raspberry Pi. First, install the Python GPIO wrapper by using the following command in the terminal

```
• sudo apt-get install
  rpi.gpio
```

Install the Python server Twisted, using the command:

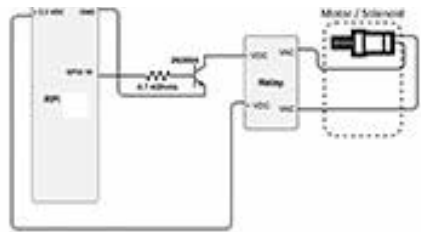
- ◆ **sudo apt-get install python-twisted**

Connect the circuit as shown in the figure. Now, in the terminal enter the following command to create a Python script that will control the motor or solenoid:

```
• nano basic_switch.py
```



Plant watering system using Raspberry Pi



Connection for plant watering system.

In the text editor that appears in the terminal, enter the following piece of code:

```

• from twisted.web.server import Site
• from twisted.web.resource import Resource
• from twisted.internet import reactor
•
• import RPi.GPIO as GPIO
• GPIO.setmode(GPIO.BCM)
• GPIO.setmode(18, GPIO.OUT)
•
• class lampAPI(Resource):
•     def render_GET(self, request):
•         if 'motor_status' in request.args:
•             if request.args['motor_status']
[0] == "off":
•                 GPIO.output(18, False)
•                 return "Motor / Solenoid
has been turned OFF"
•                 if request.args['motor_status']
[0] == "on":
•                 GPIO.output(18, True)
•                 return "Motor / Solenoid
has been turned ON"
•
• root = Resource()
• root.putChild("pi_auto", lampAPI())
• serve = Site(root)
• reactor.listenTCP(80, serve)
• reactor.run()

```

Now press [Ctrl] + [X] together, then [Y] and then press [Enter] to save the file.

Now let's create the webpage that will be used to control the motor / solenoid. Type the following command in the terminal:

```
• mkdir myserver
```

This will create a directory with the name 'myserver' as specified in the program with 'res\_var = File("myserver")'. Next, in the terminal, type the following command to change the current directory to the newly created

Code	Explanation
from twisted.web.server import Site from twisted.web.resource import Resource from twisted.internet import reactor	Imports the Twisted Python server libraries.
import RPi.GPIO as GPIO GPIO.setmode(GPIO.BCM) GPIO.setmode(18, GPIO.OUT)	Imports the GPIO libraries which contain code similar to the code for LED blinking
class lampAPI(Resource):	
def render_GET(self, request):	Define render_GET function
if 'motor_status' in request.args:	Checks if the variable 'motor_status' exists in the received URL.
if request.args['motor_status'][0] == "off":	Checks if the parameter for the 'motor_status' variable has been set to 'off'.
GPIO.output(18, False)	If the user sends the 'off' parameter, this command switches OFF the motor / solenoid.
return "Motor / Solenoid has been turned OFF"	Informs the user that the Motor / Solenoid has been switched OFF.
if request.args['motor_status'][0] == "on":	Checks if the parameter for the 'motor_status' variable has been set to 'on'.
GPIO.output(18, True)	If the user sends the 'on' parameter, this command switches ON the motor / solenoid.
return "Motor / Solenoid has been turned ON"	Informs the user that the Motor / Solenoid has been switched ON.
root = File("myserver")	Specifies the server files storage folder as 'myserver'.
root.putChild("pi_auto", lampAPI())	Creates a child to handle requests.
serve = Site(root)	Initialises the 'twisted' object.
reactor.listenTCP(80, serve)	Sets the server to handle request on port 80.
reactor.run()	Keeps listening on port 80 forever until turned OFF manually.



folder 'myserver':

- `cd myserver`

In the newly created myserver directory, create an HTML file that will be accessed by the user to switch the motor or the solenoid ON or OFF by entering.

- `nano index.html`

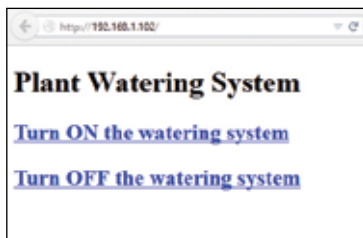
Now type the following code in the text editor to create a webpage:

- `<!DOCTYPE HTML>`
- `<html>`
- `<head>`
- `<title>Plant watering using Raspberry PI</title>`
- `</head>`
- `<body>`
- `<h1> Plant Watering System </h1> <br>`
- `<a href="http://<ip-address>/pi_auto?motor_status=on">`
- `<h2> Turn ON the watering system</h2> </a> <a`
- `href="http://<ip-address>/pi_auto?motor_status=off"> <h2>`
- `Turn OFF the watering system</h2> </a>`
- `</body>`
- `</html>`

Now press [Ctrl] + [X] together, then [Y] and then press the [Enter] key to save the file.

You could use the circuit in Step 10 of the DIY at <http://dgit.in/1Bamn4R> to build an automatic watering system.

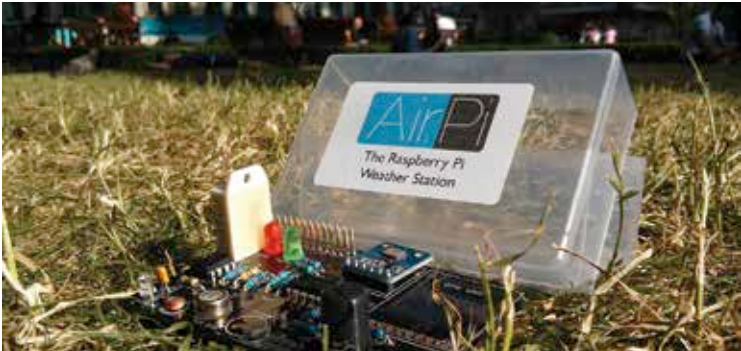
Next, open the browser on your PC and type the IP address of the Raspberry Pi to open the web interface for watering your plants.



Webpage for plant watering system control

## Create a weather monitoring system – AirPi

This DIY isn't a no-brainer and needs some serious knowledge about electronic hardware as, unlike the previous DIY, it has a slightly bigger circuit and needs some understanding of pin numbering and circuits. We'll be



Air Pi for monitoring weather

building a complete weather monitoring system, that can also be used to get the amount of toxicants and other gases in your home. Thus, you can combine the data from this circuit with your home automation system to open windows when the carbon-dioxide level in the room increases beyond the permissible limit.

#### Hardware needed:

- ◆ Raspberry Pi
- ◆ Breadboard
- ◆ Wires
- ◆ Resistors
- ◆ ADC

#### Weather sensors:

- ◆ DHT22 - Humidity & Temperature
- ◆ BMP085 - Pressure & Temperature
- ◆ UVI-01 - Ultraviolet Radiation

#### Gas sensors:

- ◆ TGS 2600 - General Air Quality
- ◆ MICS-2710 - NO<sub>2</sub> Concentration
- ◆ MICS-5525 - CO Concentration

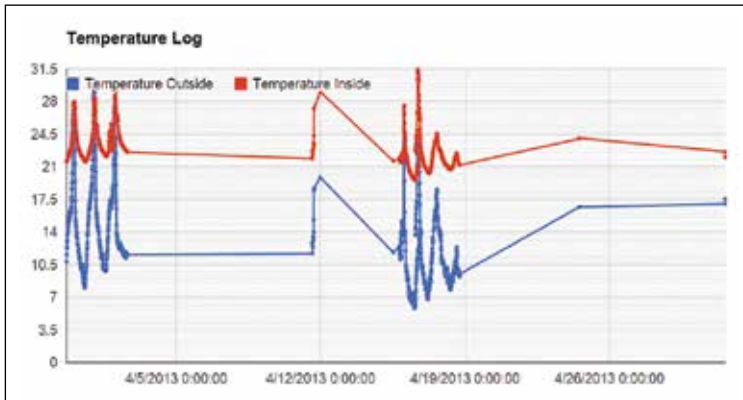
The AirPi is actually a kit that can be bought. The AirPi is a Raspberry Pi shield kit, capable of recording and uploading information about tem-

perature, humidity, air pressure, light levels, UV levels, carbon monoxide, nitrogen dioxide and smoke level to the internet.

Check out the DIY instructions at <http://dgit.in/airpies>.

## Log temperature in Google Docs

With this tutorial, you'll be able to record the temperature of your house directly to your Google Docs spreadsheet. Why record the data to Google Docs? The given code will record the data to a spreadsheet along with a time stamp. Technical analysis can be performed using this data. Further, many other scientific tools including statistical analysis tools can help you develop better ways to regulate temperature, thereby increasing the efficiency of your air-conditioning system. For instance, by using outside environmental conditions to complement the temperature regulation.



Temperature analysis in Google Docs

Hardware needed:

- ◆ Raspberry Pi
- ◆ DS18B20 temperature sensor
- ◆ 4.7 k resistor

The DIY uses the 'gspread' Python library for saving data to Google Docs spreadsheet.

To install the gspread library, use the following command at the command line:

- `wget https://pypi.python.org/packages/source/g/gspread/`

```
gsread-0.0.15.tar.gz
```

- `tar -zxvf gsread-0.0.15.tar.gz`
- `cd gsread`
- `sudo python3 setup.py install`

This will download the libraries and install them. Check out the DIY instructions at <http://dgit.in/logtempPi>.

## Monitor humidity

This DIY is very important from the point of view of farming, cricket, sports and residents of humid areas like Mumbai. Human productivity and the motion of the ball in cricket and football is linked to humidity. Agricultural yield is also largely dependent on humidity. Thus, it's very important to monitor humidity and this DIY enables you to do just that.



Humidity monitoring using Raspberry Pi

### Hardware needed:

- ◆ Raspberry Pi
- ◆ Temperature / Humidity sensor
- ◆ Wires
- ◆ Resistors

Check out the DIY instructions at <http://dgit.in/1BEF4b3>


## Monitor dissolved oxygen

Finding out oxygen levels is especially important for the owner of fish tanks. Also, people owning fish and prawn farms need to monitor the level of dissolved oxygen in their tanks. By monitoring the level of oxygen, a greater yield can be obtained from fish farms. Environmentalists and researchers use the dissolved oxygen monitoring system to monitor the level of oxygen in water bodies and thus predict the activity of fishes such as their habitat and migration patterns, among other things.

Hardware needed:

- ◆ Raspberry Pi
- ◆ Atlas Dissolved Oxygen sensor

Check out the DIY instructions at <http://dgit.in/1EA7RI7>.

To sum up this chapter, we can use the Raspberry Pi along with various sensors to sense the environment around us. Using the environmental data thus obtained, researchers and experts can monitor changes in environment. Also, it can be used to improve living conditions like that of the fish in tank example of oxygen monitoring. Further, since Raspberry Pi is a web-enabled device, web-based monitoring helps to remote control all parameters. 

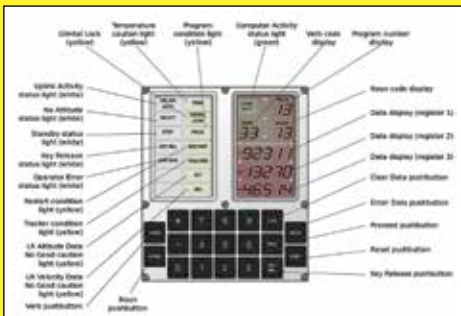
# AIN'T NO PI IN THE SKY

## What does the future hold in the world of the Raspberry Pi?

The Raspberry Pi still poses a mystery for a lot of computer users. As people learn and know more about the basic specifications of the Pi, they shrug and wonder why is it so popular. How can something that has the hardware specifications of a mid-to-low range smartphone be so amazing? This reaction is normal since we know that the public at large still doesn't understand computing and the role of computer hardware in our world.

## Processors, processors everywhere

The modern world is populated with billions of microprocessors all around us. From the cores of our cell phones to our washing machines - computer processors are at the heart of almost all modern day electronics. The power of processors to make computable



Users can build anything – from calculators to smartphones.



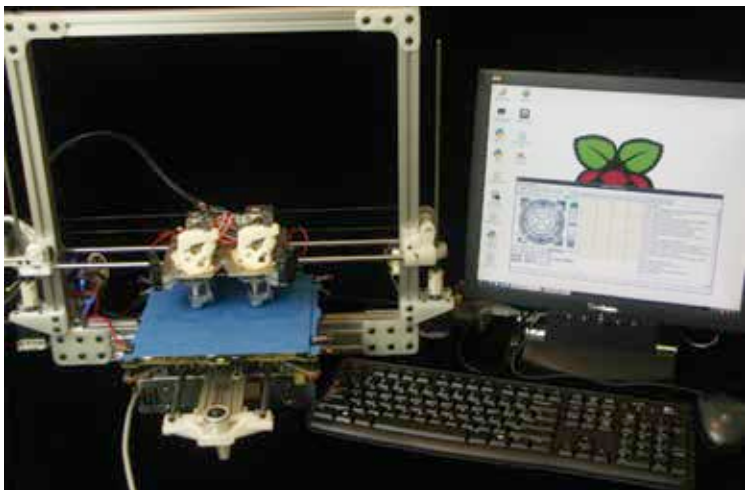
The Pi 2's processor is faster than before at the same price.

operations fast, reliable and cheap is what has led to their explosive usage. But due to the slick packaging and product differentiation of technology most consumers don't really have a benchmark for what a good processor can do.

The latest technology advertisements tell consumers that the latest chip from Intel is the best thing for their needs. But most consumers don't even realise what their needs are. The Apollo 11 guidance computer in 1969 used only a 1.024 MHz CPU, while an average computer today easily has a 1.2 GHz CPU power. Are we really doing that much on our computers? Well, the answer is yes and no. A huge part of our computing has become multi-tasking oriented in the past 30 years, which places huge demands on our computers. But no as well, because we are no longer doing epic teraflop consuming tasks anymore with the computers we have. The everyday home computer has become as under utilised as a 9-to-5 government employee.

## Return to the Age of Innocence

The loss of wonder in the modern computer age has only just found respite thanks to the Raspberry Pi. This is not because of its power or form factor or any other superficial hardware feature - it is because users can now learn and experiment with serious processing power towards experiments that weren't worth busting a smartphone or laptop over. Prior to the Raspberry Pi the everyday user couldn't get their hands on a reasonably fast SoC if they wanted to build a processor powered device at home. Such a project would involve major investment of time and money in order to obtain a suitable processor or it would require repurposing an already available CPU from



Pi machines can be used to build 3D printer set ups as well.

a laptop or desktop. Both these options were tedious and expensive, but with the Pi they're almost an impulse purchase.

The world of tomorrow in the wake of the Raspberry Pi is going to be one where people have a more direct relationship with their technology. The knowledge of computer programming, coding, building and assembly will be common knowledge to those who wish it to be. Since the cost barrier to entry will be so low, anyone interested in learning about these technologies will be free to do so. The Pi serves to remind people that great and grand projects have human origins and that one imaginative idea only needs some skill and training to become real. Just in the past three years the online community which has thrived around the Raspberry Pi is evidence of this trend.

The biggest change we can expect to see is perhaps in terms of more options. While the Raspberry Foundation seems the best suited body to ensure that costs remain low and that the Pi standard is used throughout, there is nothing wrong in new players such as BeagleBone and others from growing. Growth through competition is one of the defining features of technology and if a new device is able to make the process of using microcomputers better for the users, then everything will change even more radically.

As concurrent technologies like 3D printing and hands-on electrical tinkering becomes more commonplace, the use of microcomputers will shift as well. Today the key purpose of the Pi is to enhance computer literacy by

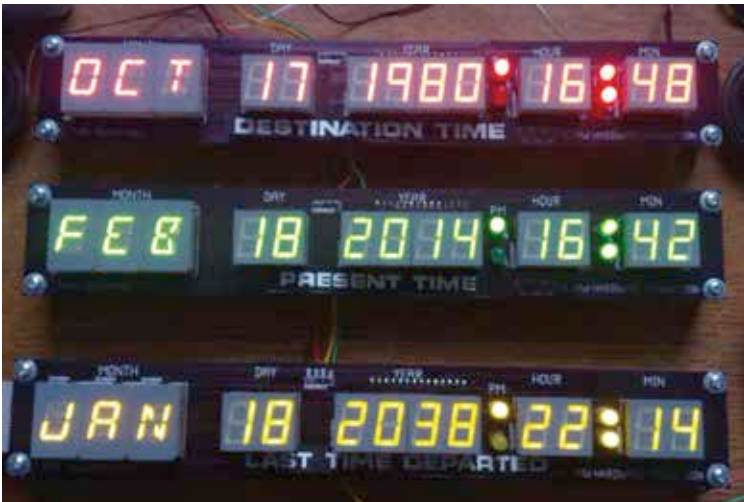


making it fun and exciting to learn programming, coding and assembly. But as these skills become a staple part of education, in the future the landscape of personalised and customised technology may also see a sharp rise. Users will no longer be slave to pre-installed programs that standardise and restrict the usage of any computer based tools. Just as people now pick up a screwdriver to tighten loose screws, microcomputer uses will hack their televisions and computers to get a performance they want.

The fundamental shift in our culture will be that people will no longer be mystified by technology. Geeks and nerds will no longer be relegated to the niche realms of pop culture. Tech support will no longer be something that people dread, and perhaps something that doesn't exist. Because in a future where building you own mini-supercomputer is cheaper than a gaming laptop we can expect to see great ideas come from the most surprising quarters.

## Dual Berry

The only aspect of the Raspberry Pi and its future trajectory that is uncertain is how long it will capture the world's imagination. Recently, the Pi 2 came out which was faster and with more memory, but in other ways was the same. It was advised that it be used for school and education purposes while the original Model A series be used for personal projects.



Fun home projects are easy to learn with online tutorials. Like this Back to the Future display.

Ultimately, the Raspberry Pi is two things at once – its a microcomputer which can run traditional operating systems and a powerful control board which can be programmed to fully enforce its processing power. While both these usages have their benefits - the former is great for kids using Scratch to learn how to code, it is the latter which truly unleashes the potential of the tool. The use of the Raspberry Pi in personal projects is a multi-disciplinary skill, since users not only need to have the coding skills necessary to program their Pi device but also the imagination, design, art and innovation to collaborate and create a brand new projects.

The world of pure IT development hasn't been as promising in the household as it is now. The quick availability of core technology, falling prices of hardware and the easier access to training is unprecedented. By making such a combination of features possible we know that a generation of children who prefer a Raspberry Pi instead of a Sony Playstation might actually be in our future. After all why play a game when you can build one? We highly encourage you to continue the educational goals of the Raspberry Pi. We advise you to become part of the online community devoted to this project since it can help, support and ignite developing interest in the broad range of skills that go into making the Raspberry Pi truly the launchboard to great ideas. It is the seed from which a million ideas can bloom. 



All this and more in the  
world of Technology

VISIT  
NOW

▶ **digit.in**

# Join 500K+ members of the digit community



<http://www.facebook.com/thinkdigit>

**facebook**

**digit.in**

**Digit** 1,100 likes

Your favourite magazine on your social network. Interact with thousands of fellow Digit readers.

- Wall
- Info
- Check In
- Current News
- Subscribe
- Open to All
- Control T
- Photo
- Video

<http://www.facebook.com/IThinkGadgets>

**facebook**

**I Think Gadgets** 1,100 likes

An active community for those of you who love mobiles, laptops, cameras and other gadgets. Learn and share more on technology.

- Wall
- Info
- Check In
- Photo
- Video
- Links
- Groups
- Questions

<http://www.facebook.com/GladtobeAProgrammer>

**facebook**

**Glad to be a Programmer** 1,100 likes

If you enjoy writing code, this community is for you. Be a part and find your way through development.

- Wall
- Info
- Check In
- Photo
- Video
- Links
- Groups

<http://www.facebook.com/devworx.in>

**facebook**

**devworx** 1,100 likes

devworx, a niche community for software developers in India, is supported by 9.9 Media, publishers of Digit

- Wall
- Info
- Check In
- Photo
- Video
- Links
- Groups