



circuit cellar

Inspiring the Evolution of Embedded Design

DRONE VIDEO TECH ASCENDS TO NEW HEIGHTS



- ▶ **Product Focus: 32-Bit Microcontrollers** ▶ **Low-Power Wireless Comms** | **IoT Monitoring for Refrigerators** | **Build a Prescription Reminder** | **High-Voltage Differential Probe** | **Microchip PIC32-Based Recording Studio**
- ▶ **The Art of Current Probing** | **Software Quality** | **Bluetooth Mesh (Part 2)** | **Building an All-in-One Serial Terminal** ▶ **The Future of Robotics**

Together, we are making security implementation easier for you.



Industry-leading development tools and ground-breaking security technology

IAR Systems and Secure Thingz are teaming up to make security implementation part of the development workflow with the release of Embedded Trust and C-Trust.

Embedded Trust is a security development environment, which simplifies the configuration of security, from the root of trust and key storage for a connected device to the creation of security profiles and projects.

C-Trust is an extension to IAR Embedded Workbench that enables application developers to deliver secure, encrypted code as part of their standard workflow.

Make security a natural part of your day-to-day development

Learn more at iar.com/security



The Embedded Experts



emCompress-ToGo

Compress Data in Real-time on any Embedded System!



More Storage



More Bandwidth



Faster Updates

One Professional Compression Solution for All Applications



Data Loggers



Internet of Things



Space / Avionics



Networking



Medical Devices



Consumer Electronics

- Real-time compression
- Small footprint
- No static RAM required
- Compression of data stream
- High performance
- High compression ratio
- On-target compression & decompression

Worldwide: sales@segger.com

+49 2173 99312 0

U.S. East Coast: us-east@segger.com

+1 978 874 0299

U.S. West Coast: us-west@segger.com

+1 408 767 4068

segger.com

OUR NETWORK



SUPPORTING COMPANIES

Accutrace, Inc.	C3
All Electronics Corp.	77
CCS, Inc.	77
HuMANDATA	33
IAR Systems, Inc.	C2
Mentor, A Siemens Business	45
Pico Technology	13
SEGGER Microcontroller Systems	1
Slingshot Assembly	21
Sensors Expo & Conference	71
Technologic Systems, Inc.	C4, 77

NOT A SUPPORTING COMPANY YET?

Contact Hugh Heinsohn

(hugh@circuitcellar.com, Phone: 757-525-3677, Fax: 888-980-1303)
to reserve space in the next issue of *Circuit Cellar*.

THE TEAM

PRESIDENT

KC Prescott

EDITOR-IN-CHIEF

Jeff Child

ADVERTISING COORDINATOR

Nathaniel Black

CONTROLLER

Chuck Fellows

TECHNICAL EDITOR

Carol Bower

ADVERTISING SALES REP.

Hugh Heinsohn

FOUNDER

Steve Ciarcia

GRAPHICS

Grace Chen
Heather Rennae

PROJECT EDITORS

Chris Coulston
Ken Davidson
David Tweed

COLUMNISTS

Jeff Bachiochi (From the Bench), Bob Japenga (Embedded in Thin Slices), Robert Lacoste (The Darker Side), Brian Millier (Picking Up Mixed Signals), George Novacek (The Consummate Engineer), and Colin O'Flynn (Embedded Systems Essentials)

Issue 345 April 2019 | ISSN 1528-0608

CIRCUIT CELLAR® (ISSN 1528-0608) is published monthly by:

KCK Media Corp.
PO Box 417, Chase City, VA 23924

Periodical rates paid at Chase City, VA, and additional offices. One-year (12 issues) subscription rate US and possessions \$50, Canada \$65, Foreign/ ROW \$75. All subscription orders payable in US funds only via Visa, MasterCard, international postal money order, or check drawn on US bank.

SUBSCRIPTION MANAGEMENT

Online Account Management: circuitcellar.com/account
Renew | Change Address/E-mail | Check Status

CUSTOMER SERVICE

E-mail: customerservice@circuitcellar.com

Phone: 434.533.0246

Mail: Circuit Cellar, PO Box 417, Chase City, VA 23924

Postmaster: Send address changes to
Circuit Cellar, PO Box 417, Chase City, VA 23924

NEW SUBSCRIPTIONS

circuitcellar.com/subscription

ADVERTISING

Contact: Hugh Heinsohn

Phone: 757-525-3677

Fax: 888-980-1303

E-mail: hheinsohn@circuitcellar.com

Advertising rates and terms available on request.

NEW PRODUCTS

E-mail: editor@circuitcellar.com

HEAD OFFICE

KCK Media Corp.
PO Box 417
Chase City, VA 23924
Phone: 434-533-0246

COPYRIGHT NOTICE

Entire contents copyright © 2019 by KCK Media Corp. All rights reserved. Circuit Cellar is a registered trademark of KCK Media Corp. Reproduction of this publication in whole or in part without written consent from KCK Media Corp. is prohibited.

DISCLAIMER

KCK Media Corp. makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors printed in Circuit Cellar®. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, KCK Media Corp. disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published in Circuit Cellar®.

The information provided in Circuit Cellar® by KCK Media Corp. is for educational purposes. KCK Media Corp. makes no claims or warrants that readers have a right to build things based upon these ideas under patent or other relevant intellectual property law in their jurisdiction, or that readers have a right to construct or operate any of the devices described herein under the relevant patent or other intellectual property law of the reader's jurisdiction. The reader assumes any risk of infringement liability for constructing or operating such devices.

© KCK Media Corp. 2019 Printed in the United States

INPUT Voltage

A Drone By Any Another Name

For over a decade before I joined the *Circuit Cellar* team, I was Chief Editor of a magazine that covered embedded computing technologies used in military systems. At that publication, I naturally wrote and edited a lot of articles about UAVs (Unmanned Aerial Vehicles). Just as am now, I was very particular about terminologies and what they represent. And one word I quite adamantly wouldn't allow used in that magazine was "drone." Back then I didn't like the term for a number of reasons. First, drone is a word that implies mindlessness or lack of intelligence. To me that didn't feel right when covering military UAVs, because they typically embedded massive amounts of computing. Large military UAVs like the Global Hawk even had full backplanes of FPGA-based boards to do processing of imaging data and other functions. A second reason is that within the defense electronics industry, UAV was the term preferred over drone. Drone was what the unknowing, non-industry public called them—the word used for them in news stories. Most news stories using the word drone were—often justifiably—bad news.


So, for those reasons I banished any use of the word drone in that publication—at least I did before a change started happening in drone world. It's important to understand that there are very few areas where the defense industry is ahead of the commercial industry. One exception, however, is UAVs—for many years the defense industry was way out ahead of the commercial world in UAV technology and development activity.

But around 2014 or 2015 a shift happened where biggest growth area for drone technology became dominated by commercial/civil unmanned platforms. Within that the largest chunk is the huge number of small hobbyist kinds of air vehicles. But as commercial uses blossomed for drones—ranging from film making to agriculture to construction and more—the drone market morphed toward a multi-billion-dollar market.

With that trend happening, I softened my stance, and I did start using the term drone when referring to consumer and commercial drones. And I knew that the defense electronics industry in this day and age has to keep tabs on the consumer technology market, because that's where the rapid innovations happen. It's too soon to tell what impact the rapid growth of the commercial/

consumer drone industry will have on the defense side of drone technology. And since most (but not all) military drones are fixed-wing and commercial drones are mostly (but not all) rotary-wing, they may continue down separate paths. But it will be important for the defense industry to keep its eyes on where commercial drone technology is going.

Interestingly, this transition from defense to commercial also played out in the tradeshow realm. When I was at that military publication, the AUVSI Unmanned Systems show was a key event that I attended every year—this was even before they shifted to the new name Xponential and then to AUVSI Xponential. That show was dominated by companies marketing to the defense UAV market, along with all the defense primes (their customers). But in the 2014/2015 time frame, that show transitioned to where the number of consumer and commercial drone companies exhibiting began to be in the majority, and that's been its direction ever since. While that wasn't a positive trend for me when I was covering defense technologies, it's very much welcome for me here on *Circuit Cellar*.

I have to be honest, writing about consumer and commercial drones is way more fun than covering military drone technology. As I've said before in this column, drone technology fascinates me partly because it represents one of the clearest examples of an application that wouldn't exist without today's level of chip integration driven by Moore's law. That high level of integration has enabled 4k HD video capture, image stabilization, new levels of autonomy and even highly compact supercomputing to fly aboard today's commercial and consumer drones. I'm looking forward to attending this year's AUVSI Xponential event in Chicago, and next month I'll be sure to share with you my thoughts about what I saw there. And as far as my objections to the word drone? Clearly, I'm over it. 



Jeff Child

COLUMNS

46 **32-Bit Microcontrollers** Wireless Workhorse

PRODUCT FOCUS

By Jeff Child

50 **Embedded in Thin Slices** **Bluetooth Mesh (Part 2)** Provisioning Pondered

Embedded in Thin Slices

Provisioning Pondered

By Bob Japenga

54 **The Consummate Engineer** **Improving Software Quality** By the Book

The Consummate Engineer

Improving Software Quality

By the Book

By George Novacek

58 **The Darker Side** **The Art of Current Probing** Mindful Measuring

The Darker Side

The Art of Current Probing

Mindful Measuring

By Robert Lacoste

66 **From the Bench** **Building an All-in-One** **Serial Terminal** With a Tiny Touchscreen

From the Bench

Building an All-in-One
Serial Terminal

With a Tiny Touchscreen

By Jeff Bachiochi

79 **TECH THE FUTURE** **The Future of Robotics** **Dealing with Challenges and** **Tradeoffs in Motion Control**

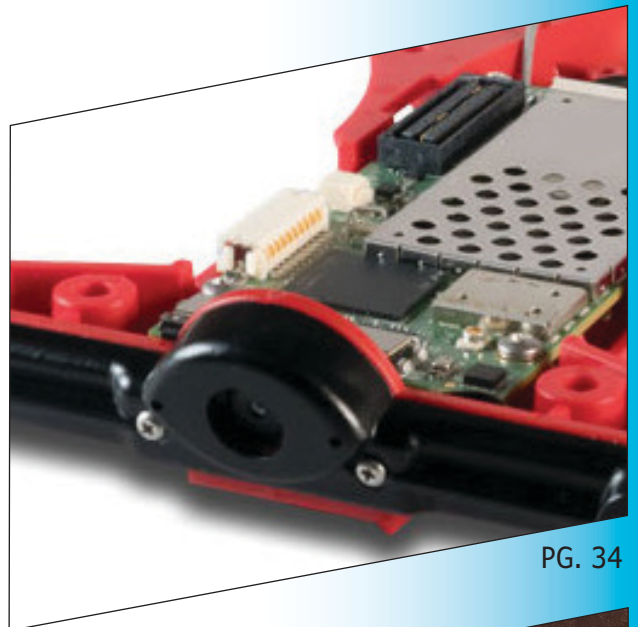
TECH THE FUTURE

The Future of Robotics
Dealing with Challenges and
Tradeoffs in Motion Control

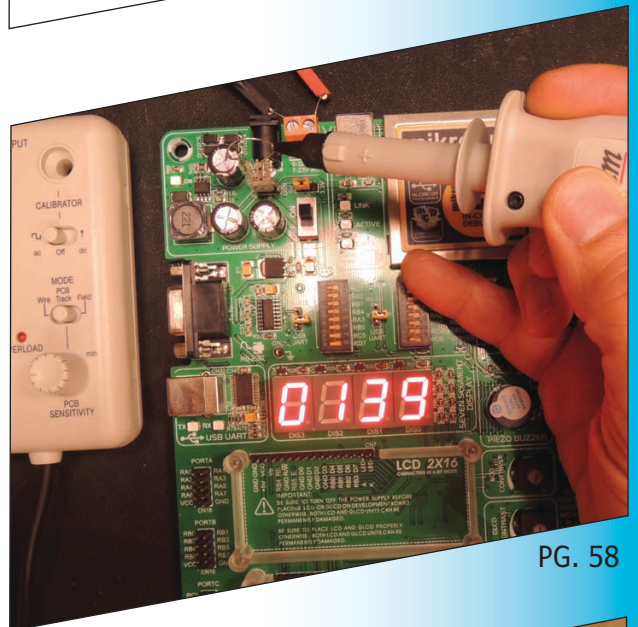
By James English

75 : PRODUCT NEWS

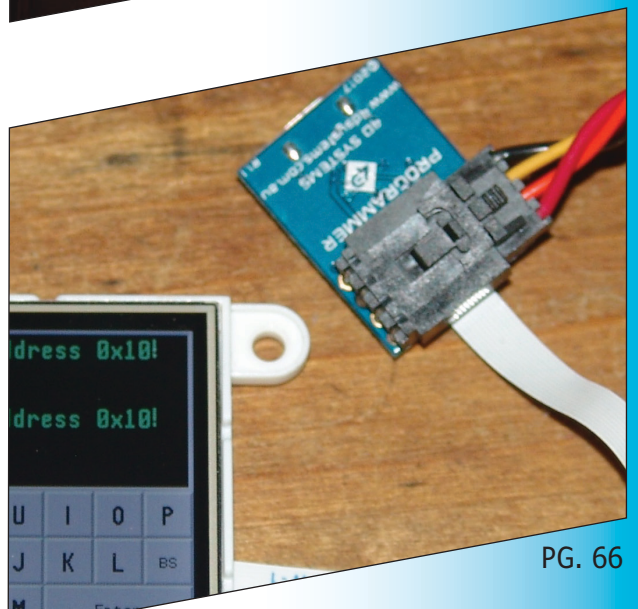
78 : TEST YOUR EQ



PG. 34

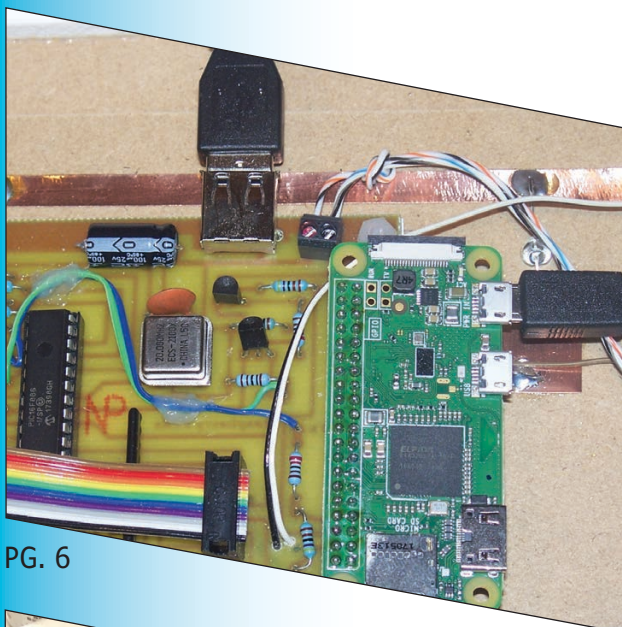


PG. 58



PG. 66

FEATURES



PG. 6

6 Build a Prescription Reminder

Using Raspberry Pi

By Devlin Gualtieri

14 IoT Monitoring System for Commercial Fridges

Using LoRa Technology

By Tyler Canton, Akio Yasu, Trevor Ford and Luke Vinden

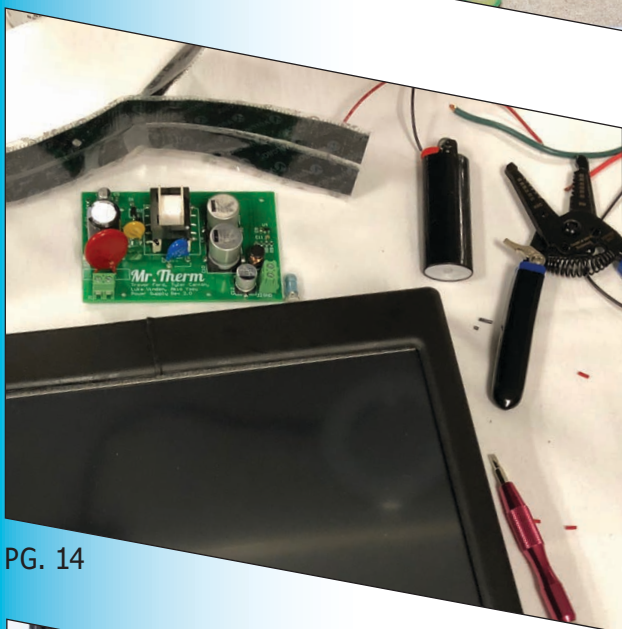
22 High-Voltage Differential Probe

Amplifiers in Action

By Andrew Levido

28 Build a PIC32-Based Recording Studio

Music from Micros

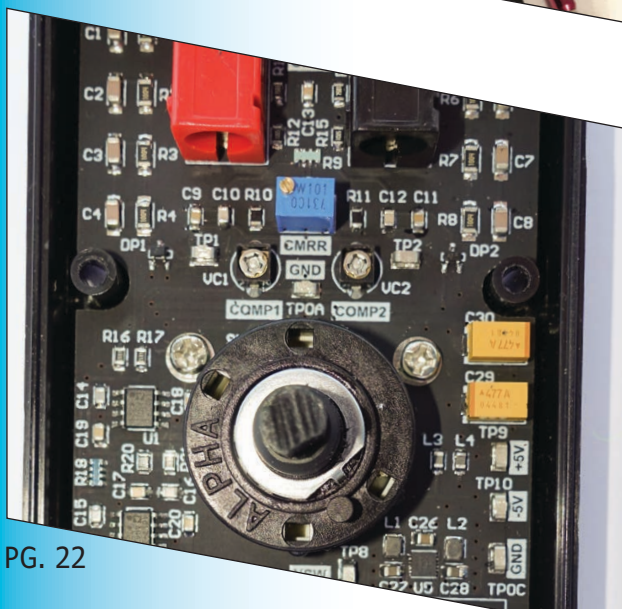
By Radhika Chinni, Brandon Quinlan and Raymond Xu

PG. 14

SPECIAL FEATURE

34 Drone Video Technologies are Flying High

Cameras, Boards and Kits

By Jeff Child

PG. 22

TECHNOLOGY SPOTLIGHT

40 Low-Power Wireless Solutions Feed IoT Edge Needs

Integration Meets Innovation

By Jeff Child

Build a Prescription Reminder Using Raspberry Pi

Pharmaceuticals prescribed by physicians are important, but they need to be taken according to a proper schedule to be effective. In this article, Devlin describes his Raspberry-Rx Prescription Reminder project, a network-accessible, Wi-Fi connected, Raspberry Pi-based device that alerts a person when a particular medication should be administered. It also keeps a log of the actual times when medications were administered.



FIGURE 1

The Raspberry-Rx. Each of the eight positions can accommodate a large or small pill bottle. There's a label for each position to ensure that the bottles are properly placed and properly entered into the dosage schedule.

By
Devlin Gualtieri

It's been said that television is a "vast wasteland," but the commercials are sometimes educational. Watching some of these will convince you that there's a pill to cure nearly every affliction, but the practical problem is that drugs need to be administered on a schedule. Some older people are prescribed so many drugs that it's often hard for them to remember which drugs should be taken when. My elderly mother had a small plastic tub filled with pill bottles and eyedropper bottles with medications that needed to be administered at different times during the day. Fortunately, a younger family member lived with her and helped with the dosage schedule.

Even younger people take many prescription drugs. I'm on a regimen of a daily baby aspirin and cholesterol medication that most men my age are advised to do. As

another preventative, I'm taking a once-a-day multivitamin/multi-mineral supplement. And I've also taken antibiotic and anti-inflammatory medications after dental surgery.

My mother's experience and my own lesser struggle with taking pills on time inspired me to devise the network-accessible appliance presented here. This device reminds me to take a drug at its intended time, and it also keeps a record of what was taken when. This is an automated system that should be as accurate as but more convenient than using a smartphone alarm, as many people do.

PRECAUTIONS

A medical appliance is quite a step up from the flashing light and beeper boxes normally built by hobbyists. While my device doesn't do something extreme—like zap my brain

with electricity—there's always a potential for harm. One adage of the early days of computers was that anything that can go wrong can go wrong faster with computers. Fortunately, anyone with the skill to build this device would realize when something isn't right—just as when I know that something is wrong when my smartphone alarm tells me to take a drug at the wrong time. The same is true for the system presented here. Due care is required in using any type of reminder system.

The US FDA has decided to encourage the development of low-risk medical apps, because these have the potential to enhance personal health. Apps that keep track of medications and provide user-configured reminders for improved medication adherence are one type encouraged by the FDA. There is also an exemption from regulation for non-commercial apps used for research. As a precaution, I contacted the FDA about this prescription drug reminder device. They replied that since it wasn't being commercialized, they weren't involved. They also didn't see any problem with my publishing this article.

RASPBERRY-Rx

I call my prescription drug reminder device the Raspberry-Rx, since a Raspberry Pi Zero W module is a principal part of its circuitry. The device, as shown in **Figure 1**, operates quite simply. Light-emitting diodes are located in front of a row of eight shallow wells designed to hold pill bottles in their standard large (1-3/4") and smaller (1-1/4") sizes. So that there will be no confusion, a label identifies the drug associated with its

well. When it's time to take a particular drug, the LED will light, and there will be an optional short reminder beep. Phototransistors are mounted below each bottle, so a signal is generated to extinguish the LED when a bottle is removed. The Raspberry Pi keeps track of the dosage schedule, and it also generates a log of when the bottles were removed.

The Raspberry Pi Zero W has built-in Wi-Fi connectivity, so it was easy to program the device to modify the dosage schedule and read the dosage log by a web browser interface. To do this, the Raspberry Pi needs to have a web server with a PHP interface enabled. There's also a supervisory program written in PHP that interfaces the Raspberry Pi to another circuit board having a Microchip Technology PIC microcontroller (MCU) that controls the LEDs and phototransistors. A screenshot of the web browser interface for managing the dosage schedule is shown in **Figure 2**.

I did the Raspberry Pi software development with a monitor, keyboard and mouse connected directly to the module. The Raspberry Pi has just one available micro USB port, so I used an inexpensive USB hub to connect the keyboard and mouse. I also needed an HDMI adapter to go from the mini HDMI on the module to a standard HDMI cable. The Raspberry-Rx operates without having these connected, so they were removed after development. There's a common complaint that the Raspberry Pi boards seem to be inexpensive, but it takes a lot of extras to make them work. The USB hub and HDMI adapter I needed for this project are useful devices, and they were removed after development. With all that in mind, I wasn't concerned about their added cost.

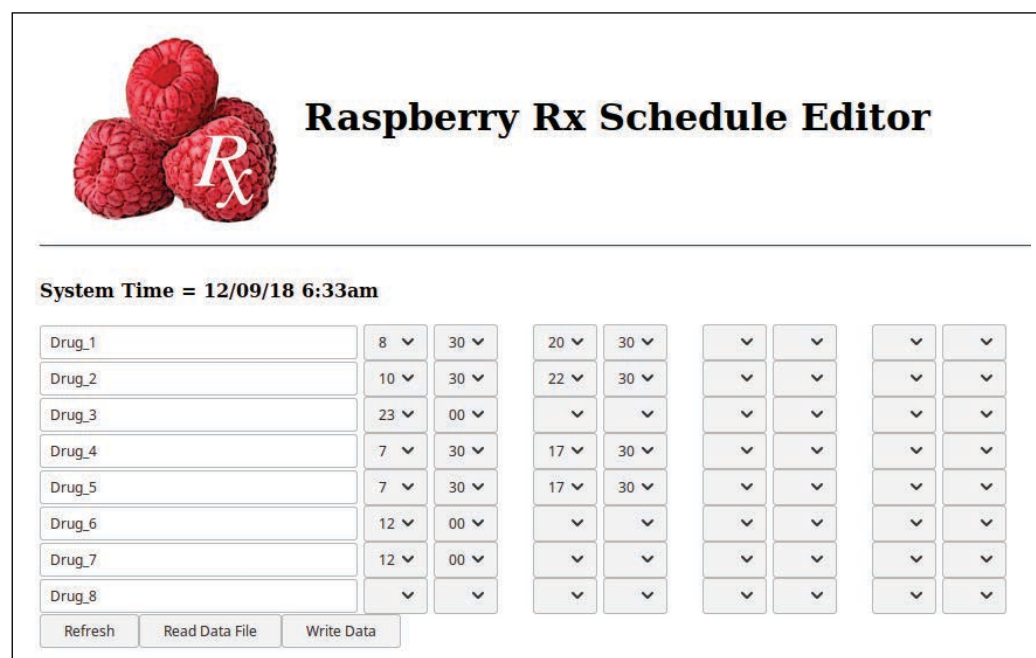


FIGURE 2

Screenshot of the Web browser interface for the Raspberry-Rx, showing the current system time. Each slot can be programmed for up to four times of the day. The 24-hour clock allows a simpler data entry than having an additional AM/PM setup. The actual drug names can be added to the first column.

The PIC module's schematic diagram (**Figure 3**) is extremely simple, because most of the heavy lifting is done in software. For easy development, I included a connection for an in-circuit serial programmer (ICSP), but the PIC firmware found on the *Circuit Cellar* article code download webpage shouldn't need to be modified, and the ICSP connector can be eliminated. The circuitry is powered by a 5 V supply, which is also the voltage requirement for the Raspberry Pi. The circuit board layout includes a USB type-A connector to couple the supply voltage to the Raspberry Pi through a short USB cable.

The PIC interfaces to the Raspberry Pi GPIO connector using a serial interface. As can be seen, this requires just a few of its 40 GPIO pins. Because the PIC is a 5 V device, level-shifting circuitry around Q1 and Q2 is needed to interface it to the 3.3 V Raspberry Pi. The Raspberry Pi is mounted on the PIC circuit board using this connector, which means that the header pins are soldered on the bottom of the Raspberry Pi board. I suggest that you buy the Pi without a connector, and solder your own onto the back side. Otherwise, a short cable can connect the two if you have a Pi with a pre-soldered connector on the front side.

Also connected to the 40-pin connector is a pushbutton switch at J4 that allows a gentle shutdown of the Raspberry Pi by shorting pins 5 and 6. Although Linux systems are usually robust against power failures—as my desktop computer has proven time and again—this shutdown should be done before removing power from the Raspberry-Rx. Some software modification is needed to enable this shutdown mode, as explained later in this article. Connection to the LEDs and phototransistors is done by two additional connectors (**Figure 4**). An external 20 MHz clock is used to keep good timing for the serial port while the PIC does its many other tasks. **Table 1** and **Table 2** list the connections, and **Figure 5** shows the PCB layout. A SIP resistor pack was used for the 10 k Ω pull-up resistors for the phototransistors. You can simulate a SIP with a bunch of vertical resistors connected at one end.

The PIC 16F886 MCU has 8,192 words of program memory, which leaves room to implement many functions. It also has 256 bytes of EEPROM for data storage, and this is used to store the dosage schedule. **Table 3** shows the serial commands accepted by the PIC module, and **Figure 6** is a portion of the underside of the Raspberry-Rx showing the GPIO connection of the Pi Zero to the PIC circuit board and the power connection through the USB-A connector and a 6" adapter cable.

PIC Pin	Function	IDC Pin	Connection
-	-	1	Ground
11	RC0	2	LED-1
24	RB3	3	LED-2
12	RC1	4	LED-3
22	RB1	5	LED-4
13	RC2	6	LED-5
21	RB0	7	Piezo Beeper
14	RC3	8	LED-6
16	RC5	9	LED-7
15	RC4	10	LED-8

TABLE 1 Connections for the LEDs and the piezo beeper

PIC Pin	Function	IDC Pin	Connection
4	AN2	1	Phototransistor-1
5	AN3	2	Phototransistor-2
3	AN1	3	Phototransistor-3
7	AN4	4	Phototransistor-4
2	AN0	5	Phototransistor-5
-	-	6	Ground
26	AN13	7	Phototransistor-6
-	-	8	5 Volts
25	AN11	9	Phototransistor-7
23	AN8	10	Phototransistor-8

TABLE 2 Connections for the phototransistors

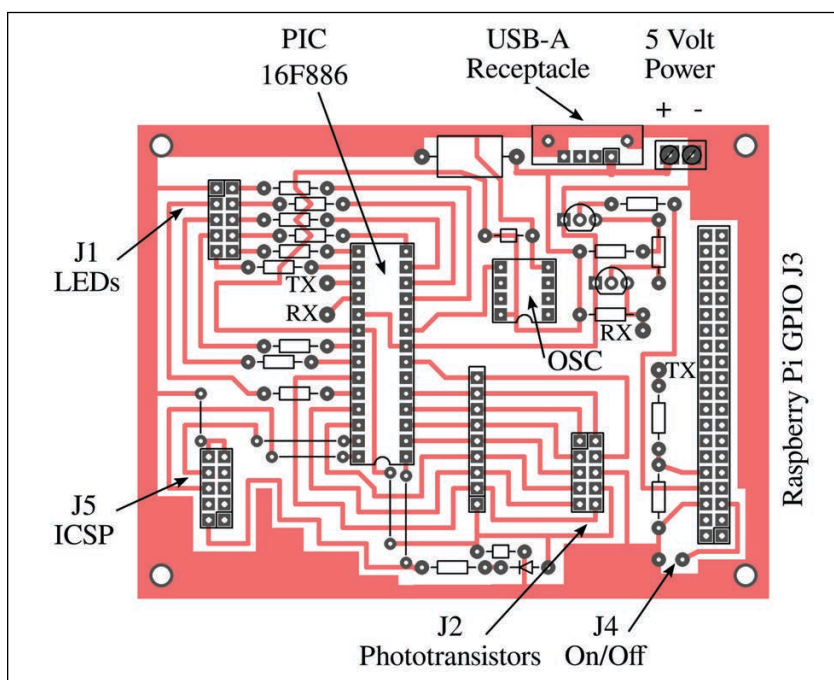


FIGURE 5

PCB layout of the Raspberry-Rx. If the 40-pin header on the Raspberry Pi Zero W is soldered to the bottom of the board, it can be plugged into this circuit board. Otherwise, the connection is made through a ribbon cable. A SIP resistor pack is used at J2 for efficient layout of the board. Connect RX to RX and TX to TX.

RASPBIAN SETUP

Raspbian is the premier operating system for the Raspberry Pi. The Raspberry-Rx uses Raspbian GNU/Linux 9.4 (stretch), which was the current version at the time of development. This must be loaded onto a

MicroSD card from another PC. Don't bother ordering an SD card pre-loaded with what's called NOOBS. When I tried this, it seemed to require network access to configure, and the network interface (password and such) was not configurable. Perhaps I was too much of a "noob" to figure it out.

While this article offers a few prescient tips relevant to this application, it can't include all the information you need to manage a Linux system. Many Internet resources are available to guide you through some necessary housekeeping, such as network connection, to get you to the point at which you can edit configuration files and download some necessary software. Among the programs you need to install are the Apache2 web server and its PHP interface:

```
sudo apt-get update
sudo apt-get install apache2
sudo apt-get install libapache2-
mod-php
```

There are instructions on many Internet websites for checking that this installation went well. While this allows PHP interaction with a web browser, you need an additional program to run PHP programs from a command line:

```
sudo apt-get install php5-cli
```

At this point, some optional utility programs, such as the terminal program CuteCom, would be useful. One editor program that I enjoy using is KWrite. When you're editing a file of known file type, such as a PHP file, this editor highlights the language syntax. To implement the off pushbutton at GPIO pins 5 and 6, you need to add the following line to the `/boot/config.txt` file:

```
dtoverlay=gpio-shutdown
```

While we're on the topic of file modification, it's useful to point out a common problem that novices have when using Linux: file permissions. Generally, if you can't create, read or write some file or folder, it's because you aren't permitted. You can view file permissions in the file manager, and you can change the permissions as the root user by prefixing commands with *sudo*. Permission is an involved topic that's better left to Internet sources.

I've found that the easiest way to edit files and change permissions is to open a file manager instance as the administrative user. This is easily done by executing the following on the command line:

```
sudo pcmanfm
```

Command	Response
?	Sends "Raspberry-Rx"
A<d>	Set alert mode (0=no beep; 1=one beep; 2= beeps every 5 minutes)
K	Sends a bare CRLF
L	Test LEDs
P	Test phototransistors
O<d>	Turn on specific LED
F<d>	Turn off specific LED
C	Read change register
X<d>	Clear bit in change register
T<bcd><bcd>	Get updated hours and minutes bytes (two packed BCD bytes)
U<bcd*64>	Update dosage times data in EEPROM (64 packed BCD bytes)
S	Send LED status
R	Read phototransistor state
V<d><d><d>	Set new phototransistor threshold. A 999 value enters a mode in which PT8 is used as a room light reference
B	Beep

TABLE 3

Listed here are the Serial Commands. In these commands <d> specifies a decimal digit, and <bcd> specified two binary-coded decimal numbers packed into a single byte. The companion PHP program on the Raspberry Pi interfaces to the PIC using these commands. Some commands are used for diagnostics and troubleshooting when the serial stream is controlled by a terminal program such as CuteCom on port `/dev/ttyS0`. The serial format is 9,600 baud, 8 data bits, one stop bit and no parity bit. A single carriage return character should follow each command.

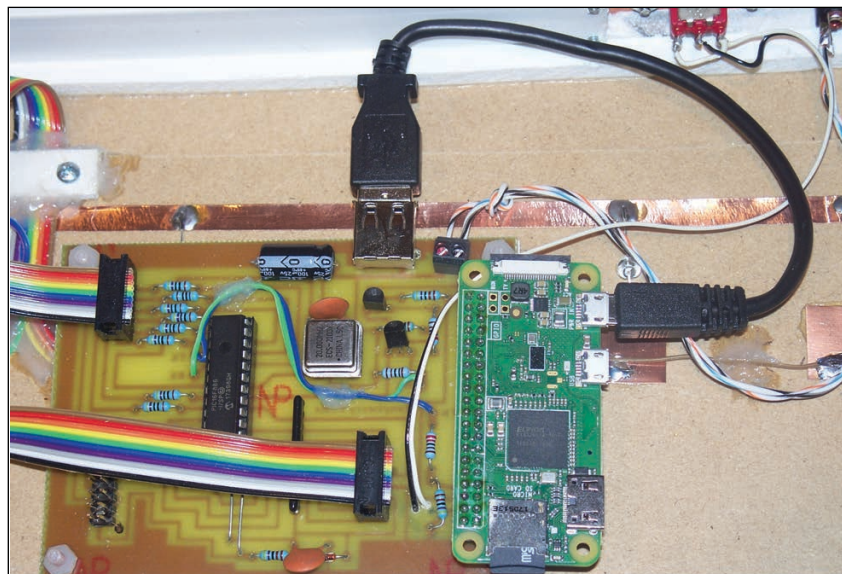


FIGURE 6

Underside of the Raspberry-Rx, showing the Pi Zero GPIO connection, and the power connection through the USB-A connector and a 6" adapter cable. Connections to the LEDs and phototransistors are facilitated by using copper tape as a bus.

Right clicking on a file or folder will give you a way to change its permissions. You can also open a file in the Leafpad text editor the same way. Since we need the serial port, and we might want to use an SSH network connection, enable SSH and the serial port in the Interfaces tab of the Raspberry Pi Configuration menu item.

SUPERVISORY PROGRAM

The firmware on the PIC acts in cooperation with a supervisory program in PHP that runs on the Raspberry Pi. I enjoy programming in PHP for several reasons. First, it has a syntax close to that of the C programming language. Second, it has myriad built-in functions that make many programming tasks easier. As an example of

this, I once needed to calculate what's called the Levenshtein distance between strings, and PHP had a built-in function to do that [1]. Finally, it allows a programmable interface in Web browser pages.

The only thing that PHP lacks for this application is an interface with the serial port of the Raspberry Pi. To accomplish this, I needed to write a short Python script that's called from the supervisory program using the PHP `exec()` command. This Python script—along with the PHP supervisory program—should be loaded into a newly created directory, Rx, at `/home/pi/Rx`. At this point you should use the CuteCom terminal program at port `/dev/ttyS0` (9,600 baud, 8 data bits, one stop bit, no parity bit and a single carriage return character following each command) to verify operation of the PIC subsystem. The `home/pi/Rx/` folder should also contain the Python script for the serial interface and symbolic links to the dosage schedule file and log file, as explained later. Then you should open a terminal in the Rx directory and run the PHP supervisory program from the command line by executing the following:

```
php raspberry_Rx.php
```

If there are any problems, you'll get error messages in the terminal to guide you.

Since normal device operation has the Raspberry Pi working without a keyboard or monitor, it's necessary to start the supervisory program automatically when Raspbian starts. This is accomplished by adding the following to the end of the `/etc/rc.local` file before its exit command:

```
php /home/pi/Rx/raspberry_Rx.php &
```

Because we're calling the supervisory program from a directory (`/etc`) that's different from its location (`/home/pi/Rx`), all file references in the supervisory program must have an absolute file path. After application of power, it takes a little less than a minute for the Raspbian operating system to load, at which time you'll see the LEDs on the Raspberry-Rx cycle on and off. At that point the device is operational.

NETWORK INTERFACE

You update the dosage schedule of the Raspberry-Rx and view the log of when medications were taken using a web browser. The wireless network interface is configured using Network Options in the graphical configuration obtained with the terminal command `sudo raspi-config`. After that, you can get the Raspberry Pi IP address by executing `ifconfig` from the command line, or using the `nmap` command on another Linux machine on your home network. These will output something like the following, from which I found the 192.168.1.253 IP address of my Raspberry-Rx:

```
wlan0: flags=4163UP,BROADCAST,RUNNING,MULTICAST>
    mtu 1500
        inet 192.168.1.253
netmask 255.255.255.0
broadcast 192.168.1.255
```

```
Nmap scan report for
raspberrypi.fios-router.home
(192.168.1.253)
Host is up (0.053s latency)
```

ABOUT THE AUTHOR

Dev Gualtieri received his PhD. in Solid State Science and Technology from Syracuse University in 1974. He had a 30-year career in research and technology at a major aerospace company and is now retired. Dr. Gualtieri writes a science and technology blog at www.tikaln.com/blog/blog.php. He is the author of three science fiction novels, and books about science and mathematics.

See www.tikalnpress.com for details.



You can set a static IP address if you want, although that takes some effort. However, I've found that the assigned IP address of my Raspberry Pi doesn't change, which is a likely consequence of the circuit being on all the time, and the router giving its connections a long lease life.

Webpages are served from the `/var/www/html/` folder, where the following files should be placed: `Rx.php`, `Rx_log.php`, `Rx_data.csv`, and `Rx_log.txt`. `Rx_data.csv` and `Rx_log.txt` are dummy files that are needed to prevent errors when the PHP supervisory program is looking for them initially. These files are in the `/var/www/html/` folder, and our program is run from `/home/pi/Rx/`, so we need to add symbolic links from one folder to the other. This is accomplished by the following commands:

```
sudo ln -s /home/pi/Rx/Rx_data.csv
/var/www/html/Rx_data.csv
```

```
sudo ln -s /home/pi/Rx/Rx_log.txt
/var/www/html/Rx_log.txt
```

WEB BROWSER INTERFACE

Although it's easiest to use the browser interface programs that I've written, you can access the Raspberry Pi using the secure shell (SSH) from a terminal. For my system with my IP address, this is accomplished in Linux with the following command:

```
ssh -X -l pi 192.168.1.253
```

Once you've connected with the Raspberry Pi, you can open a graphical instance of the file manager by typing `pcmanfm`. I've always found that graphical interfaces make life a lot easier.

You can also copy programs to and from the Raspberry pi using the secure copy protocol (SCP). On my system, I use the following commands for my Raspberry-Rx at IP address 192.168.1.253:

```
scp /local_folder/local_filename pi@192.168.1.253:/pi_folder/pi_filename
scp pi@192.168.1.253:/pi_folder/pi_filename /local_folder/local_filename
```

I access the dosage editor using the IP address of my Raspberry-Rx (192.168.1.253) by typing the following in the location box of my Firefox web browser:

```
http://192.168.1.253/Rx.php
```

To make things simple, the time granularity is 15 minutes, so valid minutes are 00, 15, 30 and 45. A screenshot of the editor page appeared earlier as Figure 2. Likewise, the log file is accessed at:

```
http://192.168.1.253/Rx_log.php
```

A screenshot of this is shown in **Figure 7**.

A FEW CAVEATS

Although most pill bottles now have child-proof caps, all medications—including the medications on this device—should be placed out of reach of children. Also, the web browser pages are not password protected and are not secure. That means that anyone on your home network can access these pages. Since the Raspberry-Rx isn't Internet-connected, it isn't really a part of the Internet of Things (IoT) that we're hearing a lot about—but hackers might be able to access it nonetheless. Some programmer more experienced than I am might be able to secure the application so that it might be safely Internet accessible.

Unlike a desktop computer, the Pi-Zero doesn't have an internal real-time clock. You need a network connection to get the time, or you can easily add a battery-powered real-time clock (RTC) to the Pi-Zero, using instructions available on the Internet. As long as the device has network access at boot-up, network time protocol (NTP) will set the clock initially and the

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials
 Reference [1] as marked in the article can be found there.


RESOURCES

Microchip Technology | www.microchip.com

Raspberry Pi Foundation | www.raspberrypi.org

Raspberry Pi will maintain the time accurately for many days using internal timekeeping. If a wireless network connection is not established at boot, the Raspberry-Rx will signal the fault by beeping at you. As an added measure, the dosage editor webpage shows the time from the Raspberry Pi clock.

Removal of a pill bottle is detected when the intensity of light at its phototransistor exceeds a threshold, so that won't happen in a dark room. Taking medications in the dark isn't likely, since we can't see what we're taking, so this shouldn't be a problem. However, many pill bottles are translucent, so some light will leak through even when the bottle is in place. This means that either the threshold needs to be adjusted to take this into account, or a small piece of opaque tape can be placed on the bottom of problematic bottles. I've found that such light-blocking tape isn't required for any pill bottle in my uniformly lighted bathroom. If you pull a bottle and the LED isn't extinguished, you either need more room light or a different threshold.

Some medications need to be refrigerated. In that case, the pill bottle in the Raspberry-Rx serves as a reminder to take the particular medication, and not the place where the actual medication is kept. Some eyedropper bottles are small, so they would need to be placed in a pill bottle for reliable operation of the Raspberry-Rx. 



Raspberry Rx Recent Log Entries

Newer		Older	
Date	Drug		
12/08/18 11:08pm	Drug 3		
12/08/18 10:31pm	Drug 2		
12/08/18 8:30pm	Drug 1		
12/08/18 5:41pm	Drug 5		
12/08/18 5:40pm	Drug 4		
12/07/18 12:02pm	Drug 7		
12/08/18 12:01pm	Drug 6		
12/08/18 10:35am	Drug 2		
12/08/18 8:45am	Drug 1		
12/08/18 7:34am	Drug 5		
12/08/18 7:33am	Drug 4		

FIGURE 7

Screenshot of a portion of the log file viewer

NEW! NEW! NEW! NEW! NEW! NEW! NEW! NEW!

pico[®]
Technology

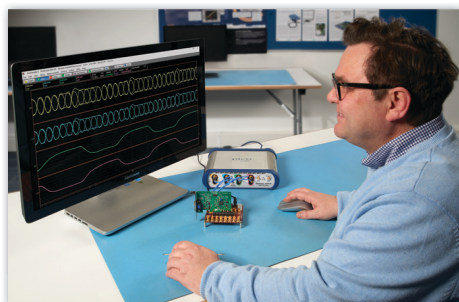
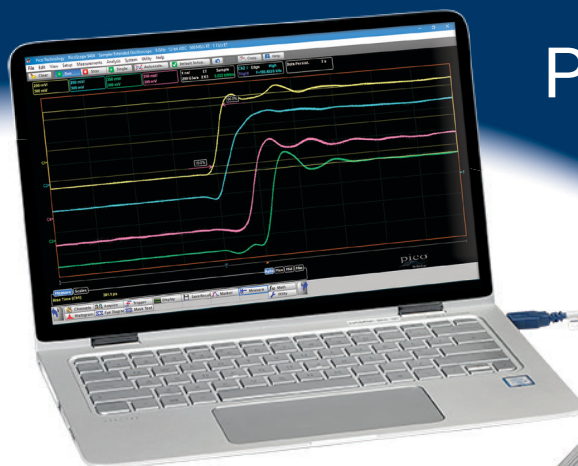
PicoScope[®] 9404-05 SXRT0

(Sampler eXtended Real Time Oscilloscope)

A new format of broadband oscilloscope

- 5 GHz bandwidth
- 1 TS/s effective sampling rate
- 4 channels
- 12-bit resolution
- Under \$15,000

The 9404-05 features four 5 GHz 12-bit channels, each supported by real-time sampling to 500 MS/s per channel and up to 1 TS/s (1 ps) equivalent-time sampling. Both the voltage and timing resolution specifications are characteristic of the highest performance broadband oscilloscopes.



For more information visit www.picotech.com/A267

Email: sales@picotech.com. Errors and omissions excepted. Please contact Pico Technology for the latest prices before ordering.

IoT Monitoring System for Commercial Fridges

Using LoRa Technology

FEATURES

IoT implementations can take many shapes and forms. Learn how these four Camosun College students developed a system to monitor all the refrigeration units in a commercial kitchen simultaneously. The system uses Microchip PIC MCU-based monitoring units and wireless communication leveraging the LoRa wireless protocol.

By
*Tyler Canton, Akio Yasu, Trevor Ford
and Luke Vinden*

In 2017, the commercial food service industry created an estimated 14.6 million wet tons of food in the United States [1]. The second leading cause of food waste in commercial food service, next to overproduction, is product loss due to defects in product quality and/or equipment failure [2].

While one of our team members was working as the chef of a hotel in Vancouver, more than once he'd arrive at work to find that the hotel's refrigeration equipment had failed overnight or over the weekend, and that thousands of dollars of food had become unusable due to being stored at unsafe temperatures. He always saw this as an unnecessary loss—especially because the establishment had multiple refrigeration units and ample space to move product around. In this IoT age, this is clearly a preventable problem.

For our Electronics & Computer Engineering Technologist Capstone project, we set forth to design a commercial refrigeration monitoring system that would concurrently monitor all the units in an establishment, and alert the chefs or managers when their product was not being stored safely. This system would also

allow the chef to check in on his/her product at any time for peace of mind (**Figure 1**).

We began with some simple range testing using RFM95W LoRa modules from RF Solutions, to see if we could reliably transmit data from inside a steel box (a refrigerator), up several flights of stairs, through concrete walls, with electrical noise and the most disruptive interference: hollering chefs. It is common for commercial kitchens to feel like a cellular blackout zone, so reliable communication would be essential to our system's success.

SYSTEM OVERVIEW

We designed our main unit to be powered and controlled by a Raspberry Pi 3B (RPI) board. The RPI communicates with an RFM95W LoRa transceiver using Serial Peripheral Interface (SPI). This unit receives temperature data from our satellite units, and displays the temperatures on a 10.1" LCD screen from Waveshare. A block diagram of the system is shown in **Figure 2**. We decided to go with Node-RED flow-based programming tool to design our GUI. This main unit is



FIGURE 1

This was the first picture we took of our finished project assembled. This SLA printed enclosure houses our 10.1" LCD screen, a Raspberry Pi Model 3B and custom designed PCB.

also responsible for logging the data online to a Google Form. We also used Node-RED's "email" nodes to alert the users when their product is stored at unsafe temperatures. In the future, we plan to design an app that can notify the user via push notifications. This is not the ideal system for the type of user that at any time has 1,000+ emails in their inbox, but for our target user who won't allow more than 3 or 4 to pile up it has worked fine.

We designed an individual prototype (**Figure 3**) for each satellite monitoring unit, to measure the equipment's temperature and periodically transmit the data to a centralized main unit through LoRa communication. The units were intended to operate at least a year on a single battery charge. These satellites, controlled by a Microchip Technology PIC24FJ64GA704 microcontroller (MCU), were designed with an internal Maxim Integrated DS18B20 digital sensor (TO-92 package) and an optional external Maxim Integrated DS18B20 (waterproof stainless steel tube package) to measure the temperature using the serial 1-Wire interface.

HARDWARE

All our boards were designed using Altium Designer 2017 and manufactured by JLCPCB. We highly recommend JLCPCB for PCB manufacturing. On a Tuesday we submitted our order to the website, and the finished PCB's were manufactured, shipped, and delivered within a week. We were amazed by the turnaround time and the quality of the boards we received for the price (\$2 USD / 10 PCB).

Main Unit Hardware: As shown in **Figure 4**, our main board's purpose is communicating with the RPi and the LCD. We first had to select an LCD display for the main unit. This was an important decision, as it was the primary human interface device (HID) between the system and its user. We wanted a display that was around 10"—a good compromise between physical size and readability. Shortly after beginning our search, we learned that displays between 7" and 19" are not only significantly more difficult to come by, but also significantly more expensive. Thankfully, we managed to source a 10.1" display that met our budget from robotshop.com. On the back of the display was a set of female header pins designed to interface with the first 26 pins of the RPi's GPIO pins. The only problem with the display was that we needed access to those same GPIO pins to interface with the rest of our peripherals.

We initially planned on fixing this problem by placing our circuit board between the RPi and the display, creating a three-board-stack. Upon delivery and initial inspection of the display, however, we noticed an undocumented

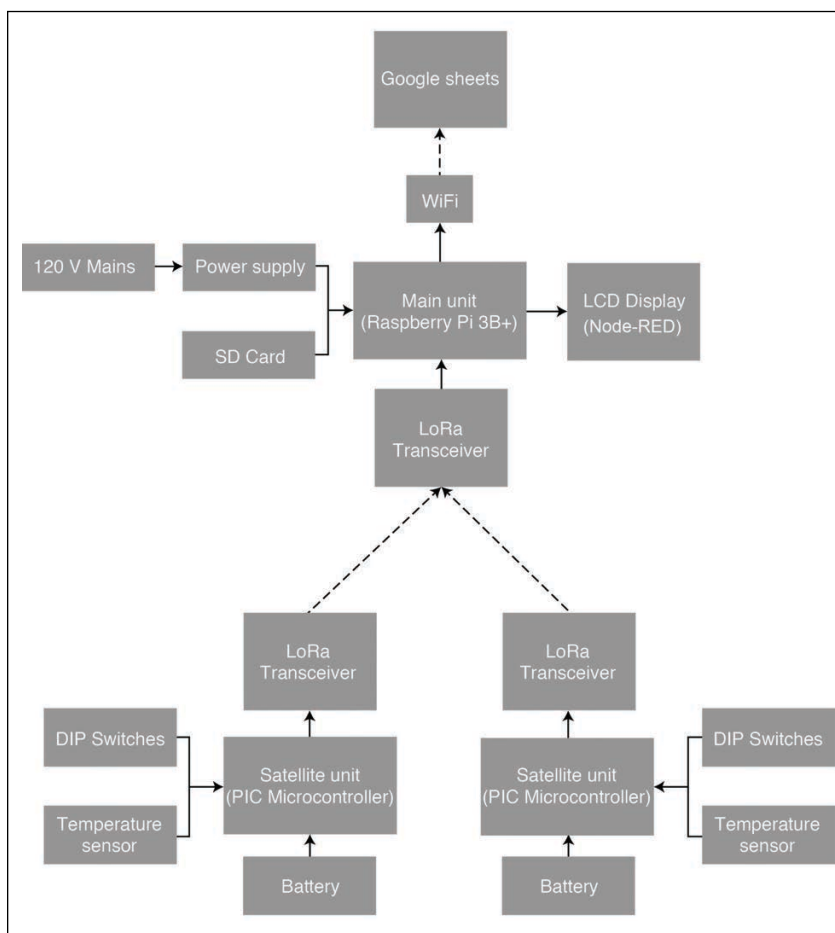


FIGURE 2

The main unit can receive temperature data from as many satellite units as required. Data are stored locally on the Raspberry Pi 3B, displayed using a GUI designed by Node-RED and logged online via Google Sheets.

footprint that was connected to all the same nets directly beneath the female headers. We quickly decided to abandon the idea of the three-board-stack and decided instead to connect our main board to that unused footprint in the same way the RPi connects to display (**Figure 5**). This enabled us to interface all three boards, while maintaining a relatively thin profile. The main board connects four separate components to the rest of the circuit. It connects the RFM95W transceiver to the RPi, front panel buttons, power supply and a small fan.

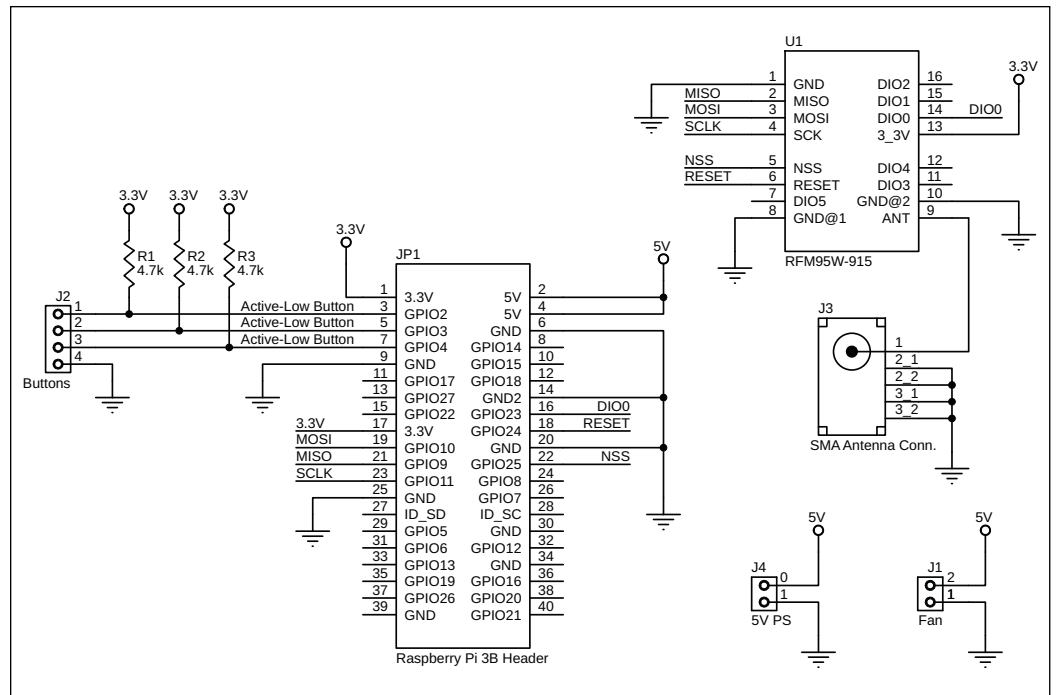


FIGURE 3

This enclosure houses the electronics responsible for monitoring the temperatures and transmitting to the main unit. These were 3D printed on Ultimaker 3 printers.

FIGURE 4

The main unit PCB's role is simply to allow the devices to communicate with each other. This includes the RFM95W LoRa transceivers, RPi, LCD screen and a small fan.



Power Supply Design: We designed our own power supply (**Figure 6**), to add more to the hardware side of our project. However, for safety's sake, we also designed a PCB to mount a CSA-approved power supply module, so we could test the system in commercial kitchens without worrying about burning them down. When we began the design process, we planned on connecting a transformer directly to the line voltage to step it down, followed by a buck converter for regulation. We did some searching and immediately realized that any transformer that fit our specification would be way too big and heavy to fit inside our enclosure, sending us back to the drawing board.

After doing some research, we came across a switching power supply topology that we had not covered in school--the flyback topology. Most power supplies used in modern electronics employ this topology, because it allows for a significantly smaller transformer while maintaining a relatively low part count. It accomplishes this by placing the switching element in series with the transformer. We began by finding a suitable PMIC (power management integrated circuit), the NCP1027 from ON Semiconductor. The next step was finding a transformer. Luckily, we found one specifically made for the NCP1027, the DA2077 from Coilcraft. From there, we were able to design the rest of the circuit shown in Figure 4, using the NCP1027's datasheet and the many reference designs Coilcraft provides.

Satellite Unit Hardware: We designed the satellite units (**Figure 7** and **Figure 8**) to be placed either inside a refrigerator or magnetically mounted on the refrigerator's side, with a temperature probe running in. Our primary concerns when designing the satellite units were power consumption and temperature accuracy. The goal was to keep the unit small, so that we could reliably solder by hand, and it would not take up too much room in smaller refrigerators. We decided to use four AA batteries to maximize both capacity and user serviceability. To reach the 1+ year battery life that we were aiming for, we had to design with efficiency in mind, starting with power regulation. After some research, we selected the Analog Devices LT3971 3.3 V buck converter for its ultra-low quiescent current ($< 2.5 \mu\text{A}$).

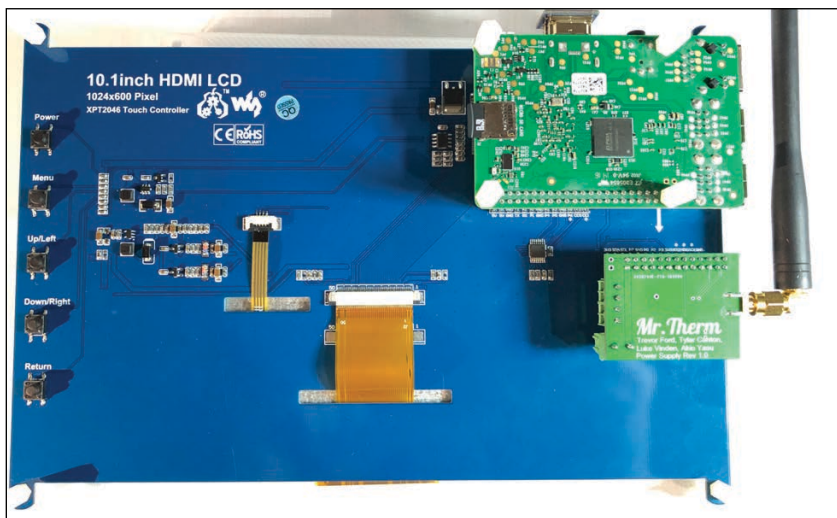


FIGURE 5

Our main board, labeled Mr. Therm, was designed to attach directly to the LCD screen headers. RPi pins 1-26 share the same connectivity as the main board and the LCD.

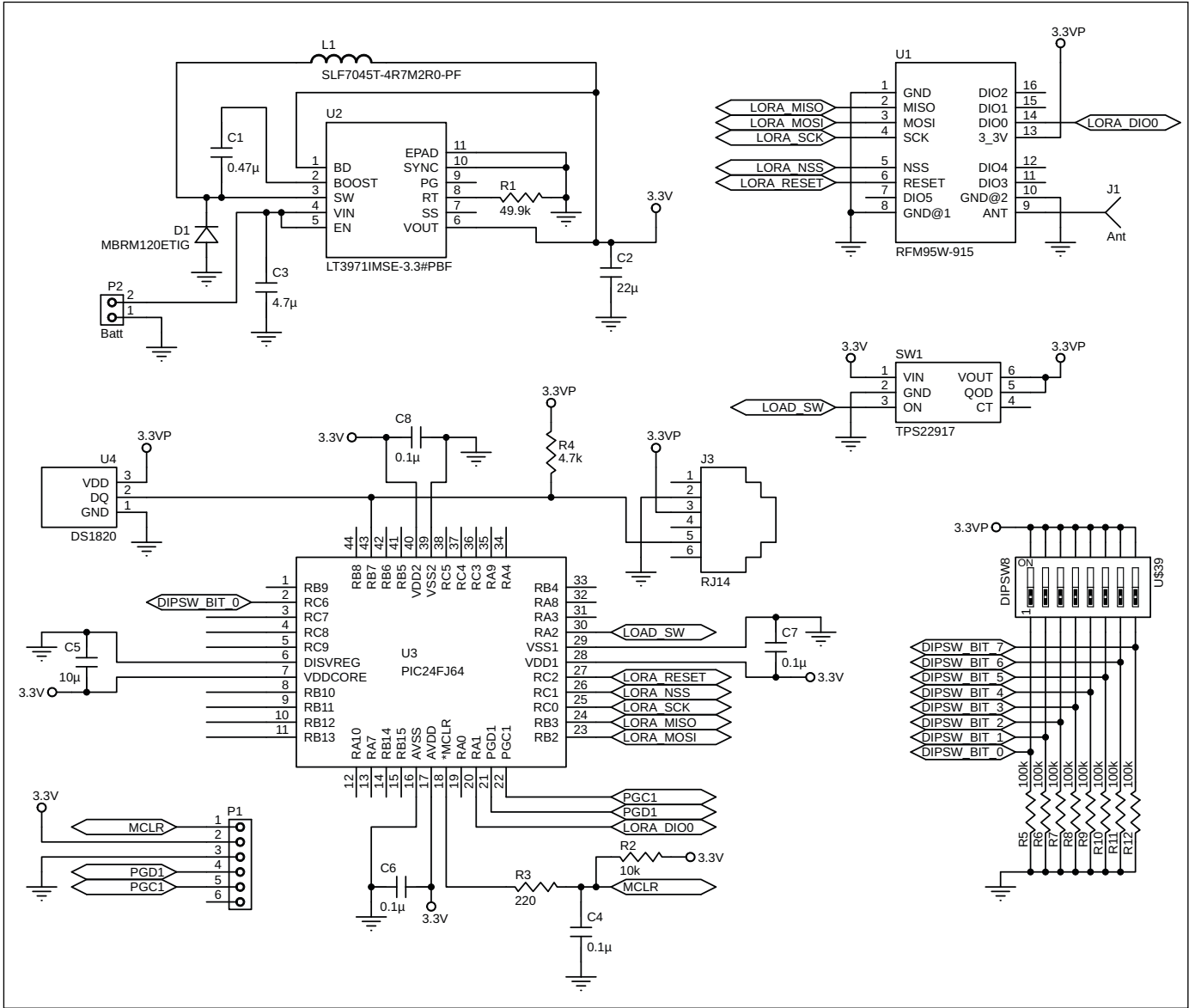


FIGURE 7

This device is controlled by the PIC. It monitors the refrigerator’s temperature with a DS18B20 on the 1-Wire bus, and transmits to the main unit via the RFM95W transceiver.

transmission to the RPi. The PIC communicates with the RFM95W module through SPI.

LoRa Initialization: For each transmission, the PIC initializes the RFM95W with an active low-logic reset on Pin 6 to reset the register values. Since this was developed in British

Columbia, Canada, RF is set to 915 MHz, the ISM (Industrial, Scientific and Medical) radio band up here in the north. Low-noise Amplifier (LNA) boost and Auto-gain control (AGC) are also set. Adding CRC (cyclic redundancy check) information is enabled to allow the receiver to discard packets with invalid data [3]. Transmission power is then set to +17 dBm (the datasheet maximum power is limited to +20 dBm). The PIC will then write the data packet (payload) onto the RFM95W FIFO register.

LoRa Transmission: The transceiver is put into TX mode, and transmission is initiated (**Figure 10**). The PIC is polling the RegIrqFlags register, waiting for the TxDone bit to go high. This indicates a transmission is complete. At this point, the PIC puts the RFM95W into RX mode to await a “packet received” message (ACK) from the main unit. There is a 100-ms timeout to await the response from the main unit. Then,

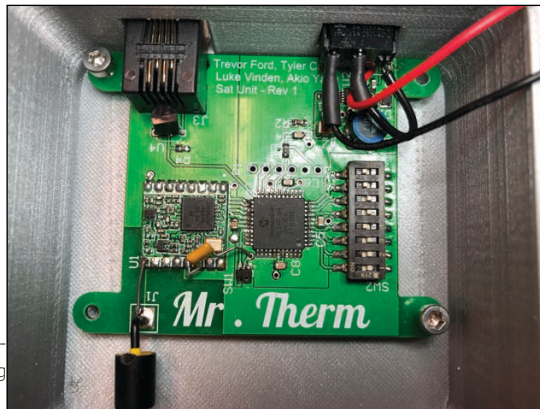


FIGURE 8

The finished satellite PCB with a long-legged post-production decoupling capacitor added for good measure.

if no ACK is received, there is a random interval delay (100 ms to 1 second) before the RFM95W will reattempt to transmit the payload. This randomized interval decreases the chance of interference with transmission from the other satellite units. The satellite unit makes three transmission attempts before going into sleep mode.

Watch Dog Timer (WDT): As soon as the PIC program starts, the WDT starts ticking. WDT times out at 16.384 seconds. The timeout causes a system reset, and the program restarts. The timeout is expected to happen when the program is stuck in a while loop during serial communication. WDT also works to wake-up the system from Sleep Mode. The program has to continue the process of sleep and wake-up several times to reach the desired time-out (about 15 minutes).

MAIN UNIT

LoRa Task: The LoRa Task puts the transceiver into Continuous Reception Operating Mode. In this mode, the transceiver continuously listens for an incoming packet. When the fidelity of the incoming packet is checked, it is stored in the transceiver's FIFO register. The transceiver then sends an interrupt signal via DIO0 to the RPi. This is the trigger for the ISR running on the RPi to initiate SPI protocol and access the RFM95W's FIFO register to read and save the data into a file.

Node-RED Task: The Node-RED task program was designed to read from a data file being written by the LoRa task. This file is updated with the received data connecting the temperature value to its ROM ID. Accordingly, this file overwritten each time data are received—no data logging here.

The Node-RED task's next job is to combine the data in both the XML file created during setup and the data being updated by the LoRa task. Basically, a new data packet is created that now associates the name of the unit being monitored with its temperature data. The Node-RED task also monitors the state of the data—have the data been updated recently, or are we sending old data and there may be

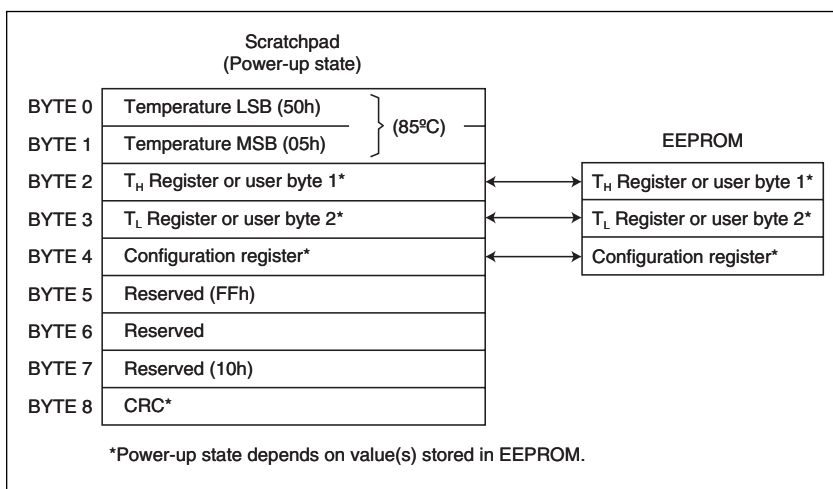


FIGURE 9 From the DS18B20 datasheet: The Memory Map (Scratchpad Table) of the temperature sensor [3].

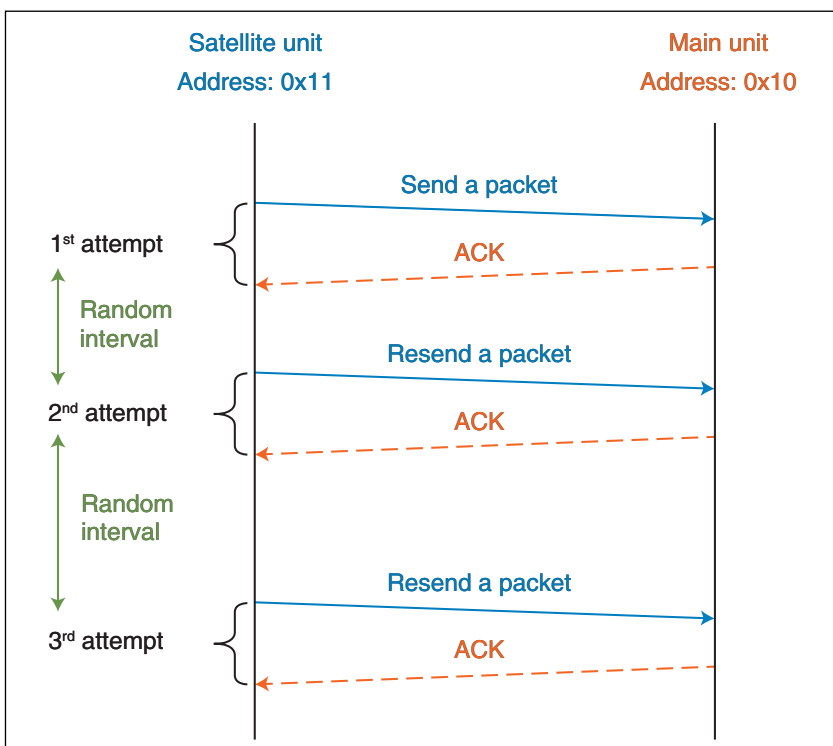


FIGURE 10 This is the format of the data packet being sent from the satellite unit to the main unit.

ABOUT THE AUTHORS

Tyler Canton (tylercanton1@gmail.com) worked as a Chef for 17 years before becoming an electronic and computer engineering technologist. He hopes to pursue a career in naval electronics on Vancouver Island, Canada.

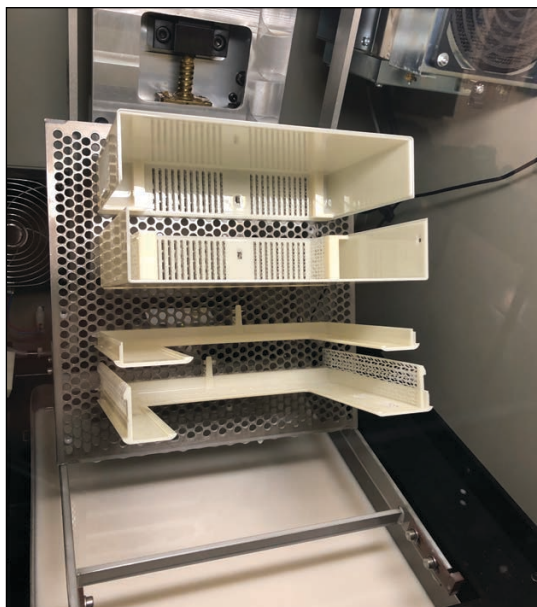
Akio Yasu (akioyasu@hotmail.com) worked as a programmer (C/C++) for 15 years. He chose to obtain a diploma in electronics and computer engineering technology to learn more about the hardware side of electronics.

Trevor Ford (trevorford@hotmail.ca) is an electronics and computer engineering technologist who, after completing the bridge program at Camosun College, plans to obtain his bachelor's in engineering from the University of Victoria. Trevor is interested in analog circuit design and printed circuit board design.

Luke Vinden (lukevinden@gmail.com) is an electronics and computer engineering technologist from Camosun College. He has a background in construction and hospital renovations. He has an interest in object oriented programming.

FIGURE 11

This is the enclosure after being raised out of the resin.



a problem with the satellite unit? As we aim to update the LCD every 15 minutes, if the timestamp of the data is more than 30 minutes old, we include this in the payload to Node-RED to alert the user of the possible issue. The Node-RED task now posts this data packet to Node-RED for the GUI display, local data logging and email alerts.

Node-RED is responsible for bringing the whole system together. It monitors the data packet containing temperature data, utilizes "email" nodes to alert the users when their temperatures are not in a safe food-storage range, displays the relevant data on the LCD screen and logs 24 hours of data. It also actively monitors the GPIO pins of the RPi, which are responsible for GUI navigation and reset.

Anyone who has used Node RED before will agree that it is an enjoyable way to program a GUI. We did find that the Node-RED community support was a bit lacking. That made it tougher for us to customize the layout and appearance of our GUI.

Note: To access some of the nodes we used, you must first install the Node-RED dashboard via the "Manage Palette" setting. This gives you access to the "template" node, where you can build a GUI using static HTML and JavaScript. This gives you the freedom to completely customize your GUI.

ENCLOSURES & GOOGLE DOCS


Satellite Units: We designed the enclosure for our satellite monitoring units using Autodesk Fusion 360. We had access to several Ultimaker 3 and Ultimaker 3 Extended 3D printers. Printing our enclosure prototypes with polylactic acid (PLA) would not give us the long-term water resistance that our sensors needed, but served as acceptable prototype models. PLA is biodegradable and derived from renewable resources, and was available to us in a variety of bright and vibrant colors. PLA also holds up well in cold environments, considering we were popping these into refrigerators and freezers.

Main Unit: We recruited a mechanical engineering technologist to design and print the enclosure for the main unit and LCD. He provided us access to a Viper SLA printer, and designed some sleek enclosures for our LCD housing (**Figure 11**).

Google Docs: Because our target demographic is a chef or food service manager (not usually technically oriented), we chose to go with a basic and user-friendly format. Storing the data on Google Forms gives accessible data on any mobile platform, and with multiple PC operating systems via Google Sheets. Compared to using ThingSpeak or Firebase, we didn't think that for our purpose, accessing the data in .JSON or .CSV format was necessary. We required something that could be opened up and checked by the least technologically savvy person.

CONCLUSION

This project was a great learning experience for our team of rookie technologists. We had huge success with the RFM95W transceivers. When testing our signal strength, we were able to transmit reliable data until about -120 RSSI. We had no problem transmitting from inside a refrigerator, up several flights of stairs, and to the main unit. The line-of-sight transmission data were reliable up to about 1 km.

In hindsight, we likely would have used an alternative to Node-RED. Although it was fast and easy to get a basic GUI up and running, we found it quite difficult to customize. However, that may be because of our own inexperience. In addition, we should have spent more time designing for effective EMC (electromagnetic compatibility) and making use of proper decoupling capacitors in the PCB design stage. This as a topic was introduced after we had designed our PCBs, and would have influenced the way we designed our circuit boards. 

For detailed article references and additional resources go to: www.circuitcellar.com/article-materials
References [1] through [3] as marked in the article can be found there.

RESOURCES

Analog Devices | www.analog.com

Altium | www.altium.com

Coilcraft | www.coilcraft.com

JLCPCB | jlcpcb.com

Linx Technologies | linxtechnologies.com/wp

Maxim Integrated | www.maximintegrated.com

Microchip Technology | www.microchip.com

ON Semiconductor | www.onsemi.com

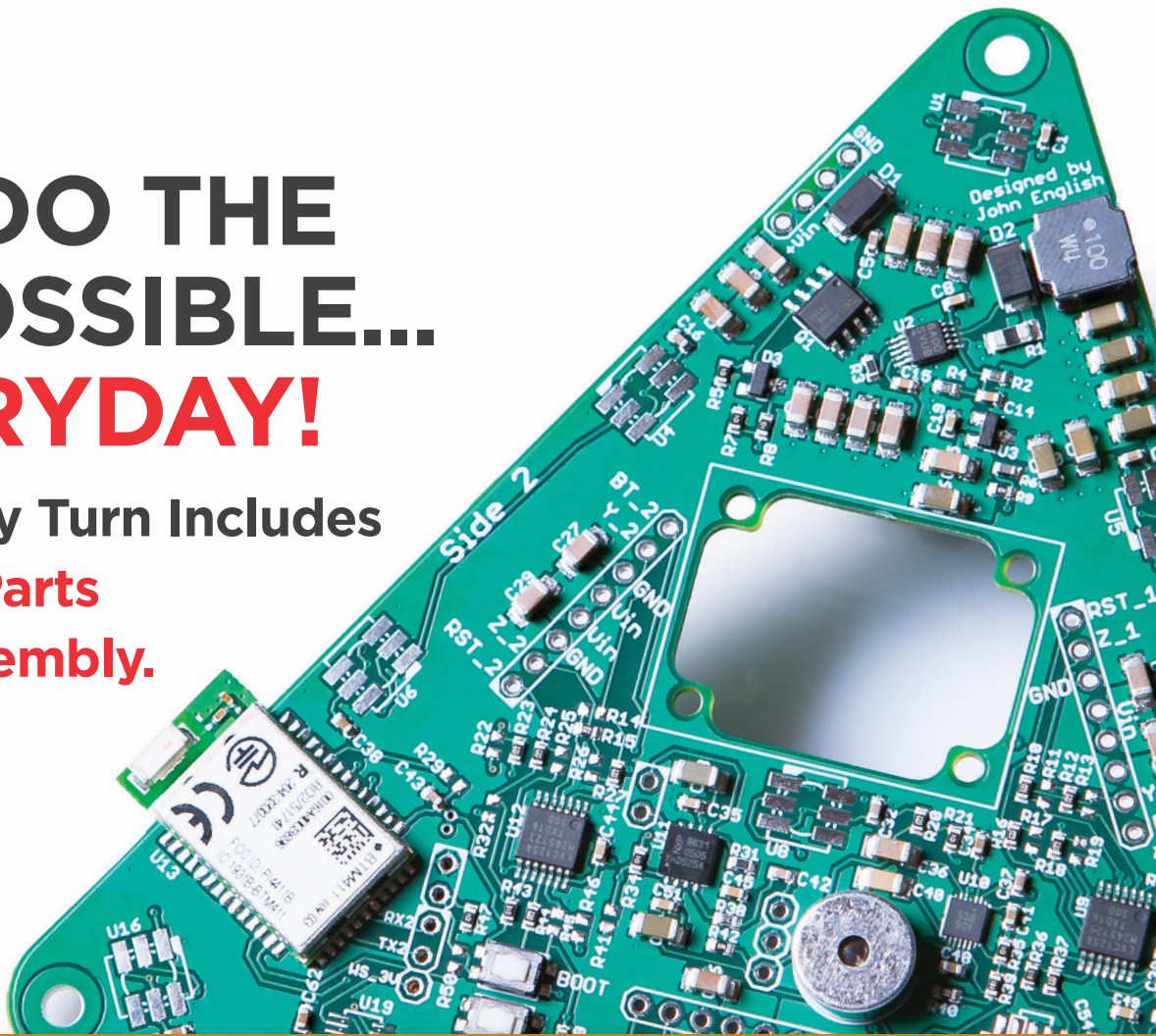
Raspberry Pi Trading | www.raspberrypi.org

RF Solutions | www.rfsolutions.co.uk

Waveshare | www.waveshare.com

WE DO THE IMPOSSIBLE... EVERYDAY!

Our 5-Day Turn Includes
**Boards, Parts
AND Assembly.**



FREE LABOR

TRY SOMETHING DIFFERENT!

1st time customers receive FREE LABOR, up to \$1,000 on your first turn-key order.

OUR ASSEMBLIES START AT \$250

DOWNLOAD YOUR OFFER CODE HERE: Circuitcellar.com/SlingShot

We want to see **NEW DESIGNS** and **NEW CUSTOMERS!**

No more sacrificing quality for speed or price.

We are your **PCB ASSEMBLY SPECIALISTS!**

SlingShot
ASSEMBLY



Find out why we're different at SlingShotAssembly.com/Different

Call for details: **720.778.2400** or Email: sales@sassembly.com

*Free labor, up to \$1000, for first-time customers on full turn-key assembly orders only.

High-Voltage Differential Probe

Amplifiers in Action

By
Andrew Levido



A high-voltage differential probe is a critical piece of test equipment for looking at high-voltage signals. In his article, Andrew describes his design of a high-voltage differential probe with features similar to commercial devices, but at a considerably lower cost. It uses just three op amps in a classic instrumentation amplifier configuration and provides a great exercise in precision analog design.

A high-voltage differential probe is an indispensable piece of test equipment for anyone who wants to examine high voltage signals on a standard oscilloscope and do so safely. For safety reasons, the ground side of your oscilloscope probe is connected directly to mains earth. This means you can only measure earth-referenced signals, or truly floating signals, such as those encountered in battery-powered circuits, where it is possible to connect one part of the circuit to mains earth via the scope.

But what if we want to measure some signals in a mains-referenced circuit such as an off-line switch-mode power supply? The control circuit is typically referenced to the negative side of the rectified mains. This point is certainly not at earth potential. If you were to connect the ground clip of a standard oscilloscope probe to this point, you would create a short directly to earth via your oscilloscope. This would almost certainly damage your oscilloscope and probably destroy the circuit under test.

Incidentally, when I started my career many years ago in the power electronics field, you would routinely see oscilloscopes with the earth wire cut at the mains plug to avoid just this problem. This was—and remains—an extremely dangerous practice, because it meant that the entire oscilloscope would be floating at mains potential. Merely touching the case of the scope could deliver a fatal shock. I even owned a 1970s era oscilloscope with a “ground lift” switch on the back panel to make it more convenient to make the instrument lethal.

Fortunately, there is a safe way to measure high-voltage circuits such as in this example: a differential probe. It has two high-impedance inputs and a single, ground-referenced output. The output is proportional to the difference in voltage between the positive and negative inputs. Any common-mode signal is rejected. In the example of our off-line switcher, we would connect the negative input to the “reference” rail, and the positive input to the point we want to investigate. The output would be proportional to the difference between the two.

DESIGN APPROACH

While I have owned a commercial differential probe for some years, I find it a little bit inconvenient to use. That’s because it requires a wall-wart power supply that clutters up my benchtop and it is quite chunky. I thought it would be nice to build a differential probe that was reasonably small sized with a rechargeable battery. It has proved to be an interesting exercise in precision analog design, as you will see later.

I wanted specifications similar to commercial units costing \$300 or \$400, so I started by writing down a few target specifications:

- Basic gain accuracy of better than 1%.
- Input impedance above 1 M Ω with parallel capacitance under 5 pF.
- Input voltage range suitable for measuring 230 V +15% mains, which we have in Australia (let’s say ± 400 V). It should be safe to connect to considerably higher voltages

- A bandwidth of at least 25 MHz
- DC and 50/60 Hz common-mode rejection ratio (CMRR) of 60 dB or better (1/1,000)
- More than 3 hours of battery life

In principle, the concept of a differential probe is pretty straightforward: a matched pair of input attenuators followed by a classic three op-amp differential instrumentation amplifier, as shown in **Figure 1**. You can think of this circuit as having three sections: an attenuator, a buffer stage and a difference amplifier stage. The overall gain of the circuit is given by multiplying the gains of each of these stages as shown in this equation:

$$\frac{V_{out}}{V^+ - V^-} = \left(\frac{R_b}{R_a + R_b} \right) \left(1 + \frac{2R_c}{R_d} \right) \left(\frac{R_f}{R_e} \right)$$

I started by choosing the input attenuation ratio to be 1/200, so that the maximum operating voltage of ± 400 V will be attenuated to ± 2 V—low enough to be within the input common mode range of typical FET input op amps powered from ± 5 V rails. Of course, this means smaller signals—such as the gate drive in our notional power supply—will be just a few tens of millivolts after the attenuator. Clearly, we need the following stage to optionally provide some gain when we are looking at small differential signals.

The good news is that the gain of the buffer stage of a classic instrumentation amp such as this can be programmed a single resistor, R_d in Figure 1. If R_d is open circuit, the gain of this stage will be unity, suitable for input signals in the multi-hundred-volt range. By switching in a resistor R_d we can add an additional range with a gain of perhaps 10, allowing us to sensibly measure differential signals in the tens of volts.

The final stage is the difference amplifier, which converts the differential signal into a single-ended ground-referenced one. I decided to set the gain of this stage to 2, so that the overall differential gain of the instrument would be 1/100 or 1/10, and the maximum output swing is nominally ± 4 V which we should be able to achieve with ± 5 V rails. Therefore, our full-scale input ranges are ± 400 V or ± 40 V.

So far, so good. I now had three analog stages to design, plus a power supply. To achieve an overall DC gain accuracy less than 1%, each stage had to have a gain accuracy of about 0.25%. Achieving greater than 60 dB CMMR (common mode rejection ratio) means matching the attenuators and the resistors in the difference amplifier to at least 1 part in 1,000. Achieving a large-signal bandwidth greater than 25 MHz would require some pretty special op amps and careful attention

to parasitic capacitances and board layout. I also had to contend with all the usual non-ideal characteristics of op amps such as input offset voltages, input common mode ranges, input bias currents and the like. Sounds like fun, so here I will walk through the design process pretty much as I tackled it.

INPUT ATTENUATOR

For this stage, we need a pair of input attenuators capable of safely withstanding mains voltages, with a high input impedance, each with a gain of 1/200 matched to better than 0.1% and with a bandwidth of at least 50 MHz.

I started by selecting the input resistance R_a to be 4 M Ω , made up of a string of four, 1 M Ω , 0.1%, 0.4 W, 1206 SMT resistors in series. Series resistors are required here, since the maximum working voltage of a single resistor of the type I chose is limited to 200 V. A quick check shows power dissipation in this resistor string will not be an issue up to 2.5 kV, so the maximum voltage is limited to 800 V by the resistors' maximum working voltages. This comfortably exceeds our design requirement of 400 V.

The value of R_b can now be calculated to be 20.1005 k Ω . The schematic in **Figure 2** shows this pretty odd value is made up of a string of resistors and a trim-pot. In practice, we can achieve a DC CMRR of better than 120 dB with this arrangement.

Half the trimpot accounts for 50 Ω of the value of R_b . The rest is made up of two 10 k Ω , 0.1% resistors and a 91 Ω , 1% fixed resistor. The 10 k Ω resistors are part of a 4-resistor matched array. This turns out to be a much cheaper way of buying precision resistors than as individual components, though with a limited range of values. As an added bonus, these resistors are matched to each other within 0.05% and track with temperature, because they are in on the same substrate. This helps keep our attenuators matched.

The astute reader will have already observed that this combination of resistors adds up to 20.141 k Ω , not the 20.1005 k Ω

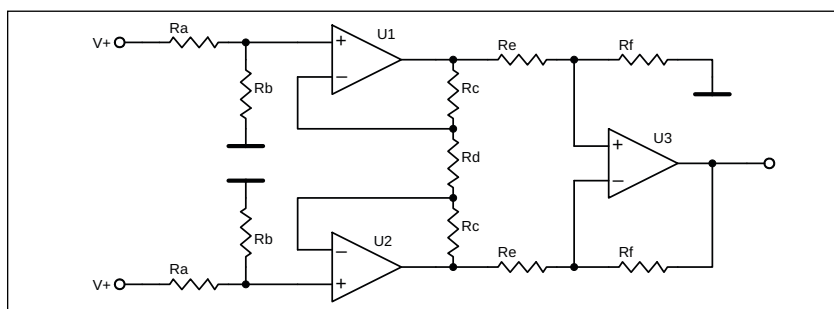


FIGURE 1

The basic circuit for the differential probe is deceptively simple. A matched pair of attenuators is followed by a classic three op amp instrumentation amplifier. Of course, making this work in reality is quite a challenge.

mentioned above. This is because we have to take into account the effect of the parallel 10 M Ω resistors (R12 and R15). 10 M Ω in parallel with 20.141 k Ω is bang on 20.1005 k Ω . R12 is associated with op amp offset nulling, discussed later, while R15 is there purely to keep the circuit symmetrical.

We can safely ignore the input impedance of the op amp as we will use an FET input type with an impedance north of 100 G Ω . Similarly, a quick calculation shows that with an approximately 20 k Ω source impedance, we can ignore the op amp input bias current as long as it is less than 50 pA. The diode pairs DP1 and DP2 are there to protect the op amp inputs from any overvoltage making its way through the attenuator. They effectively clip the signal to one diode drop above or below the supply rails.

This is all well and good for DC signals. We have a safe circuit with precise attenuation and great CMRR. But what about the AC behavior? The buffer op amps will have a small but finite input capacitance, as will the protection diodes. This capacitance, with the 4 M Ω input resistance, forms a low-pass filter that will severely attenuate high-frequency signals. According to the datasheets, this combined capacitance of the op amp and

protection diodes will be on the order of 4.3 pF. Not much, you might say. But the corner frequency of this low-pass filter will be under 10 kHz.

The answer, of course, is to add some frequency compensation capacitance across the 4 M Ω resistors. The impedance of this and the parasitic capacitance should be in the same ratio as the resistances in the voltage divider. In our case, the compensation capacitors should be 199 times smaller than 4.3 pF. Obviously, this is not practical, so we have to tackle this problem from the other direction. First, let's choose a reasonable compensation capacitance and calculate the other capacitance.

I chose to use a string of four 10 pF, 500 V, 5%, 1206 SMT capacitors in series. This gives us a compensation capacitance of 2.5 pF and will handle input voltages to 2 kV, comfortably exceeding our ± 800 V input limit defined by the resistors. For proper frequency compensation we now need a total capacitance at the input to the buffer op amps of 497.5 pF. In practice, this is made up by the parallel combination of the op amp and diode input capacitance, two fixed capacitors (C9/C10 and C11/C12) and a trimmer capacitor. The trimmer provides a range of adjustment of 461 pF to 524 pF to

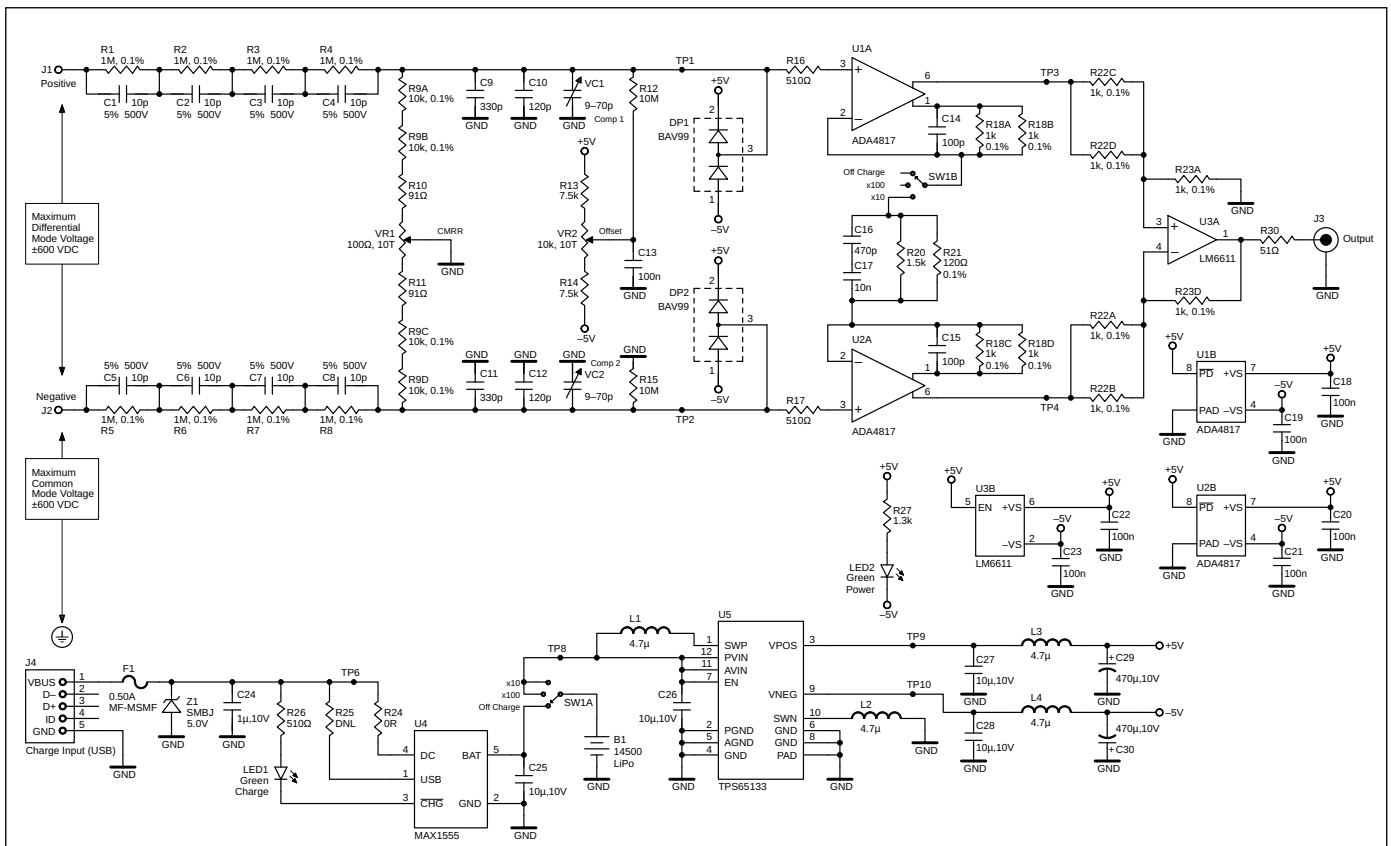


FIGURE 2

This is the full schematic of the project. Comparing this to the simple diagram of Figure 1 gives you an idea of what is involved in implementing a relatively simple circuit if you want a high-precision result.

allow for component tolerances and layout differences. The total capacitance seen at the input terminals is nominally 2.5 pF, since the compensation capacitors will dominate.

INSTRUMENTATION AMPLIFIER

Next is the three op amp instrumentation amplifier. This classic circuit has a few very nice features that come in handy for this application. It has very high input impedance, common mode gain of 1 independent of resistor matching, and differential mode gain programmable via just one resistor as already discussed.

The technical requirements for the input op-amps for U1 and U2 are pretty tough. I needed FET inputs for high impedance and low bias current, a very wide large signal bandwidth, a flat response below about 50 MHz, low input offset voltage, ± 5 V supply rail supplies and an input common mode range to ± 2 V. I chose Analog Devices' ADA4817, which is expensive at around \$10, but fits the bill nicely. It has an input impedance of 500 G Ω in parallel with 1.3 pF, typical input bias current of 2 pA, a large signal bandwidth to 200 MHz, 0.1 dB gain flatness to 60 MHz, a typical offset voltage of 0.4 mV (very low for a FET input op amp) and an input common mode range of -4.2 to 2.2 V with ± 5 V rails.

If I only required a gain of 1 for this stage, I could have simply wired these op-amps as non-inverting buffers. Since I wanted to have the option of a gain of 10, I had to close the feedback loop around each op amp with a resistor (R18 in the schematic and R_c in Figure 1). It is a good idea to choose a fairly low value for this resistor, because it will form an RC low-pass filter with the op amp input capacitance—the effect of which will be to increase the gain of the buffer as the frequency rises.

I selected a resistance of 500 Ω , made up from a pair of paralleled 1 k Ω resistors from another precision matched resistor array. This minimizes the effect of gain peaking in the frequency range of interest, but there is still a potential issue at high frequencies that should be addressed. The 510 Ω resistor in series with each op amp's non-inverting input is the result. It forms another low-pass filter that attenuates the input signal at about the same rate as the low-pass filter in the feedback loop amplifies it. Neat.

With R_c (Figure 1) fixed at 500 Ω , we can easily calculate the value we will need for R_d to achieve a gain of 10 on the high-gain range. This turned out to be the infinitely repeating value of 111.11 Ω . This odd value was easy to create with a parallel combination of 120 Ω and 1.5 k Ω (R20 and R21 in Figure 2). You will note from the schematic that I have specified a 0.1%

tolerance for the 120 Ω resistor, but only 1% for the 1.5 k Ω resistor. I could do this because the error in the 120 Ω resistor dominates, and any tighter tolerance on the other resistor is just wasted money. The same applies for the 91 Ω resistors and the trimpot in the input attenuator. This is a trick well worth keeping in mind for your precision designs.

Just as for the input attenuator, I had to provide frequency compensation for the voltage divider formed by R_c and R_d (Figure 1). As before, I selected a nice round value (100 pF) for the capacitors across R_c (R14 and R15 in Figure 2) and calculated that we needed 450 pF across R20 and R21. This is a series connection of 470 pF and 10 nF capacitors.

The final difference amplifier is, by comparison, fairly simple to design. The rejection of any remaining common-mode signal relies on well-matched components, so again I took advantage of low-cost matched resistor arrays for R22 and R23. For this op amp, I could relax the requirements a little compared with U1 and U2. Any input impedance greater than about 5 M Ω and an input capacitance lower than about 5 pF should be fine, because the source impedance is relatively low. Similarly, an input bias current less than 10 μ A should be no problem. I needed an input common mode range of ± 2 V and an output swing of ± 4 V. A lower cost CMOS op amp should suffice here, so one would expect fairly low input offset voltages compared to the FET input op amps used in the previous stage. I did, however, need a pretty good large signal bandwidth.

I chose the LM6611 from Texas Instruments (TI), which has an input impedance of 6 M Ω in parallel with 2.5 pF. Typical input bias current is -6.5 μ A. Input common mode range is -5.2 V to +3.8 V, and input offset voltage is typically 74 μ V. The output can swing to within 200 mV of the supply rails into a 1 k Ω load. The output is intended to be connected to a standard oscilloscope input, which is typically a 1 M Ω resistance in parallel with a few picofarads of capacitance. The 51 Ω resistor R30 is there to protect the output op amp from inadvertent short circuits.

ABOUT THE AUTHOR

Andrew Levido (andrew.levido@gmail.com) earned a bachelor's degree in Electrical Engineering in Sydney, Australia, in 1986. He worked for several years in R&D for power electronics and telecommunication companies before moving into management roles. Andrew has maintained a hands-on interest in electronics, particularly embedded systems, power electronics, and control theory in his free time. Over the years he has written a number of articles for various electronics publications and occasionally provides consulting services as time allows.

FIGURE 3

The finished differential probe. The 4 mm banana-type sockets are on the top end and the BNC connector is at the bottom. Two LEDs show the power and charging status.



POWER SUPPLY

Calculations show we need ± 5 V power rails with a maximum current drain in the order of 80 mA. I chose to use a single cell LiPo battery as the power source, since this could be charged from the standard USB power supplies that have become ubiquitous. An AA-sized (14,500) cell would fit nicely in the case I had in mind. I needed two switch-mode converters—one to step the battery voltage up to +5 V and another to create the -5 V rail. There are a few ways to do this, but I figured I could not be the first person with the need for dual 5 V supplies from a single LiPo cell.

After a few hours on various manufacturer websites, I came across the TPS65133 from TI, which fit the bill perfectly. This nice little chip accepts a 2.9 to 5.0 V input and produces ± 5 V rails at up to 250 mA. It is 92% efficient at 100 mA. It requires only a couple

of inductors and three low ESR ceramic capacitors. The chip also has a low-voltage shut-down to protect the LiPo cell from over-discharge. I was concerned that there might be a bit too much switching noise on the power rails, so added an LC filter (L3/C29 and L4/C30) between the switcher and the analog circuitry. A green LED (LED2) across the power rails provides user indication that the power is on.

The battery is charged from a Micro B USB connector via a MAX1555 charger from Maxim Integrated. This is a very simple linear charger with two inputs, intended to be used in applications where there is both a USB power source and a DC wall wart. If power is present at the DC input (U4 pin 4), the LiPo cell will be charged at 280 mA. If only the USB source is available (U4 pin 1), charging will be limited to 100 mA, respecting the USB specification. I have configured the circuit so I can connect either of these inputs to the USB connector via R24 and R25. By omitting R24 and loading a 0 Ω resistor in R25, I can choose to use a high-current USB charger for faster charge or, by omitting R25 and loading a 0 Ω resistor in R25, configure the differential probe to be USB compliant and put up with a longer charge cycle. A green LED (LED 1) indicates that the battery is charging. The USB power input is protected by a TVS diode (Z1) and a resettable thermal fuse (F1).

Note that the power switched via SW1A. In the Off/Charge position, the battery is connected to the charger and isolated from the rest of the circuit. In either the x10 or x100 gain position, the battery is connected to the switcher and isolated from the charger. This means it is not possible to both use and charge the probe. This was a deliberate choice on my part to keep the modes completely separate and encourage me to keep my bench as clear as possible.

Testing shows the circuit can achieve 3 hours and 45 minutes of operation on a full charge with the 800 mA-hours LiPo I used. Charging takes a couple of hours on the high-current setting, and about 6 hours on the low-current setting. You could substitute an 18650 cell if you reconfigure the mechanical design, which should more than double both the running and charging times.

CONSTRUCTION

I built my probe on a double-sided printed PCB, sized to fit a Hammond Industries 1593X handheld case, measuring 66 W mm x 140 L mm x 28 D mm. **Figure 3** shows a view of the case. Two 4 mm banana input sockets protrude through one end of the case, and the BNC output connector and USB Mini-B

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials

RESOURCES

Analog Devices | www.analog.com

Maxim Integrated | www.maximintegrated.com

Texas Instruments | www.ti.com

connectors protrude through the other end. The power/range switch is on the top of the case, which also houses two off-the-shelf light-pipes for the LED indicators. The label was made using a laser printer label and transparent sticky plastic film. A pdf of this artwork is available on the *Circuit Cellar* code & files download webpage.

The board layout is critical. If you intend to do this yourself, you must keep the input attenuator symmetrical and observe the manufacturer's layout guidelines for the ADA4817s, which will happily oscillate at 250 MHz given half a chance. Trust me, I learned this the hard way. Use short traces, use the special "feedback" output pin and do not use a ground plane in this area. You should also take care with the layout of the DC-DC converter, keeping loops small and the whole lot far from the sensitive analog circuitry. **Figure 4** shows the completed circuit board in the case.

Assembly was straightforward, with the DC-DC converter chip being the only leadless part that takes a bit of care to solder by hand. I stuck with 0805 and larger components where possible, to make things a bit easier for myself.

TESTING AND CALIBRATION

Once you have built the differential probe, you need to go through a simple calibration and testing procedure.

1. Check that the ± 5 V supplies are present and at the right value when the power switch is in the x10 or x100 position. The power LED should be lit. Make all the tests with the power on.
2. Temporarily short C13, to eliminate the effect of the offset adjustment on the input attenuator.
3. Adjust the CMRR. Connect a high but safe DC voltage between both inputs and ground. I used 60 V from my ± 30 V bench supply. Connect a digital multimeter between TP1 and TP2. Adjust VR1 (CMRR) to null the voltage between these inputs as close to zero as possible. You should be able to get down below 60 μ V (-120 dB) with a bit of care.
4. Adjust the DC offset. Connect the voltmeter between TP5 (the output) and ground. Remove the short from C13, connect the two inputs together and adjust VR2 (Offset) for the lowest possible output voltage. You will need to switch between ranges frequently to dial in the best compromise. It will be unlikely you can get to zero on both ranges, but you should be able to get below ± 2 mV on each without too much trouble.




FIGURE 4

The internals of the completed probe. Note the symmetric layout of the input attenuators. The buffer amplifiers are to the left of the switch and the output amp on the left near the BNC connector. The power supply is below the switch to the right, well away from the analog circuitry.

5. Adjust the frequency compensation. Feed a 1 kHz square wave into the positive input with respect to ground. Crank up the amplitude on your signal generator as far as it will go. Set the probe to the x10 setting. Connect your oscilloscope to the probe via a BNC-to-BNC cable. Ensure the scope is on the high-impedance setting and any bandwidth limiting is off. You should now be able to trigger on the square wave and adjust the compensation trimmer VC1 for optimum compensation, just as you would compensate a standard x10 probe. The correct compensation is achieved when the rising edge of the square wave shows no overshoot or undershoot. Use a non-metallic tool to make this adjustment. Repeat the process for the negative input and VC2.

CONCLUSION

I found designing and building this project very satisfying. The opportunity to design a project that does not include a microcontroller and required me to dust off my analog design skills was a welcome change. The end result is a useful instrument that exactly meets my needs and is in regular use on my bench. 

Build a PIC32-Based Recording Studio

Music from Micros

By
*Radhika Chinni, Brandon Quinlan
 and Raymond Xu*

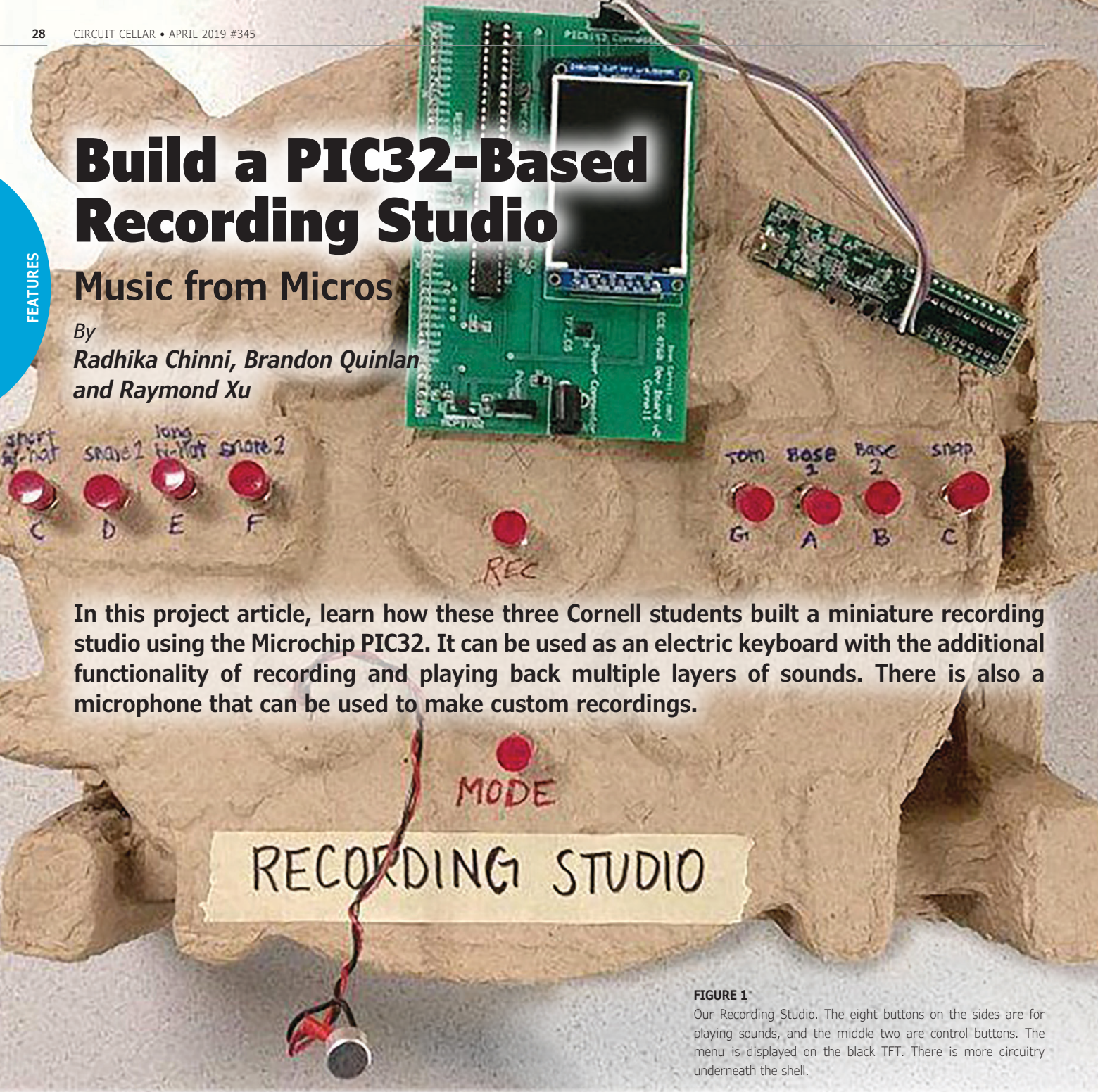
In this project article, learn how these three Cornell students built a miniature recording studio using the Microchip PIC32. It can be used as an electric keyboard with the additional functionality of recording and playing back multiple layers of sounds. There is also a microphone that can be used to make custom recordings.

FIGURE 1

Our Recording Studio. The eight buttons on the sides are for playing sounds, and the middle two are control buttons. The menu is displayed on the black TFT. There is more circuitry underneath the shell.

Our project was motivated by combining our passion for music and the knowledge we gained from an introductory microcontrollers (MCUs) course at Cornell University [1]. Inspired by electronic keyboards and the idea of making a song by looping multiple tracks together, we designed and built the PIC32 Recording Studio (**Figure 1**). A user can record a few seconds of music on various instruments and layer them over one another.

For example, a user can first record a basic drum beat, then play back the drumbeat and add a bass track to the “song” they are currently building. Continuing in this fashion, a user can build a complete song, and using the provided microphone, can also add personalized sounds—such as their own voice—to the track. With visual feedback through an LCD screen on the Recording Studio, the user can easily switch modes and playback/recording options.



USER INTERFACE

The Studio can be controlled through a TFT LCD screen and two menu buttons. The TFT screen serves as a menu interface and provides some visual feedback for the user. The screen displays a menu that allows the user to toggle between playable instruments, to record/stop recording and to manage the recording through playback and delete options. As the toggle button is pressed, an arrow is cycled through all the menu options, indicating the current operating mode of the Studio. This is demonstrated in **Figure 2** where the menu arrow is at “piano,” meaning the Studio is currently in piano mode. The various modes displayed on the menu are Piano, Guitar, Bass, Drums, User, Playback and Delete.

If the menu arrow is at one of the four “instrument” modes, the user can play one octave of that instrument on the buttons similar to those on an electric keyboard. To start/stop playback or delete the current recording, the user has to move the arrow to that menu option and then press the Select button. To record the instruments the user is currently playing, the user can press a designated Record button. If the user is recording or playing back a recording, a progress bar appears on the screen, so that current progress in the length of the recording can be tracked. We found this feature extremely useful for synchronizing the beats of our recording. User mode activates a microphone through which the user can sing/make sounds to add to the current track they are building, allowing the user to add personalized sounds to a recording.

In addition to the two buttons for the user interface, there are eight buttons for playing specific notes. For each of the three tonal instruments, these eight buttons allow the user to play one octave, without flats or sharps. Each instrument plays in a different octave from the others, to give the greatest possible range. While in drum mode, each of the eight buttons plays a unique drum hit. We arranged the buttons in the style of a keyboard to give the Studio a natural feel for experienced musicians.

To format the sounds to work with the PIC32, we searched YouTube for people playing C major scales with guitar, piano and bass. Links to these are provided on the *Circuit Cellar* article materials webpage. For the drums, eight different drum sounds were selected—about 0.5 s each. Using Audacity, we clipped the notes (not the drum sounds) to a relatively short length, about 0.05 s, and then put them on repeat for playback on the device. We then used MATLAB to sample the sound files at 8 kHz, and scaled them to be

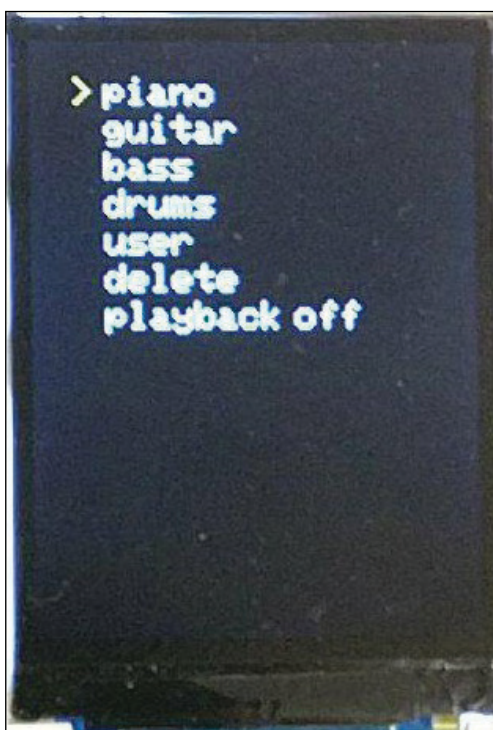


FIGURE 2

An image of our menu, showing that the Studio currently is in “piano” mode

integers in the range “0 to 256.” The final step was to copy these values into a header file as an array of unsigned chars. To save space for recording, we saved the header file in flash memory, separate from the main memory. Having the sound array in a header file also made it easier to access and modify the array, which we found ourselves doing many times throughout the debugging process.

RECORD AND PLAYBACK

After basic sounds could be played in real time, the next step in the project was to incorporate the recording and playback features. The Studio allows for the user not only to record and layer the instruments they are playing, but also to record a personalized sound through a microphone. The microphone is set up to receive input sound from the user and feed it to the PIC32 through ADC. The output of the microphone was amplified by implementing a level shifter and non-inverting amplifier on the output of the microphone before wiring it to ADC (**Figure 3**). Since the microphone was intended for a person’s voice, we chose appropriate RC time constants to allow those frequencies to pass. For our amplifier, we chose to increase our gain to 100, so the waveforms would reach their full amplitude from -1.5 V to 1.5 V. This caused some clipping to occur when sounds were very loud into the microphone, but as long as the mic was used from a few inches away this was not a problem. To use the microphone to record sounds, a user must press Record

while in user mode, then the input from the microphone is added to the recorded sound.

The Record button has several different functions. First and foremost, it can be used to begin recording. A user can record music by pressing Record while in one of the four instrument modes. If playback is on—which is useful to help time multiple tracks—the user should see “Get ready!” in the bottom right of the screen. On the next cycle of playback, the recording will begin. If playback is not on, a countdown appears on screen, so the user can see when the recording will begin. When the recording begins, a progress bar appears on the screen to indicate the current point in the recording. If a recording is already in progress, the current recording will stop. Another use of the Record button is to delete the current track. If Delete is selected in the menu, pressing the Record button sets the entire record wave to 0s, effectively deleting the recording. Alternatively, pressing the Record button while Playback is selected

in the menu toggles playback between on and off. The tasks of actually generating the sounds through a speaker were divided into two parts in our code: a thread for reading input from the user and an interrupt service routine (ISR) for creating the sounds.

SOFTWARE IMPLEMENTATION

The software for this project was divided into a few main parts. An ISR was responsible for writing the bits of our sound waves to the output. We also had two threads [2]. The first—called the button thread—was for reading input from the buttons. The second was the draw thread, which updated the content on the TFT display, moving the arrow to the current mode selected and changing the progress bar if recording or playback was active. In the button thread, we read and debounced the buttons to determine what sounds needed to be generated, while the ISR took that information about what buttons were pressed and wrote the data from our sound array to the DAC through SPI [4].

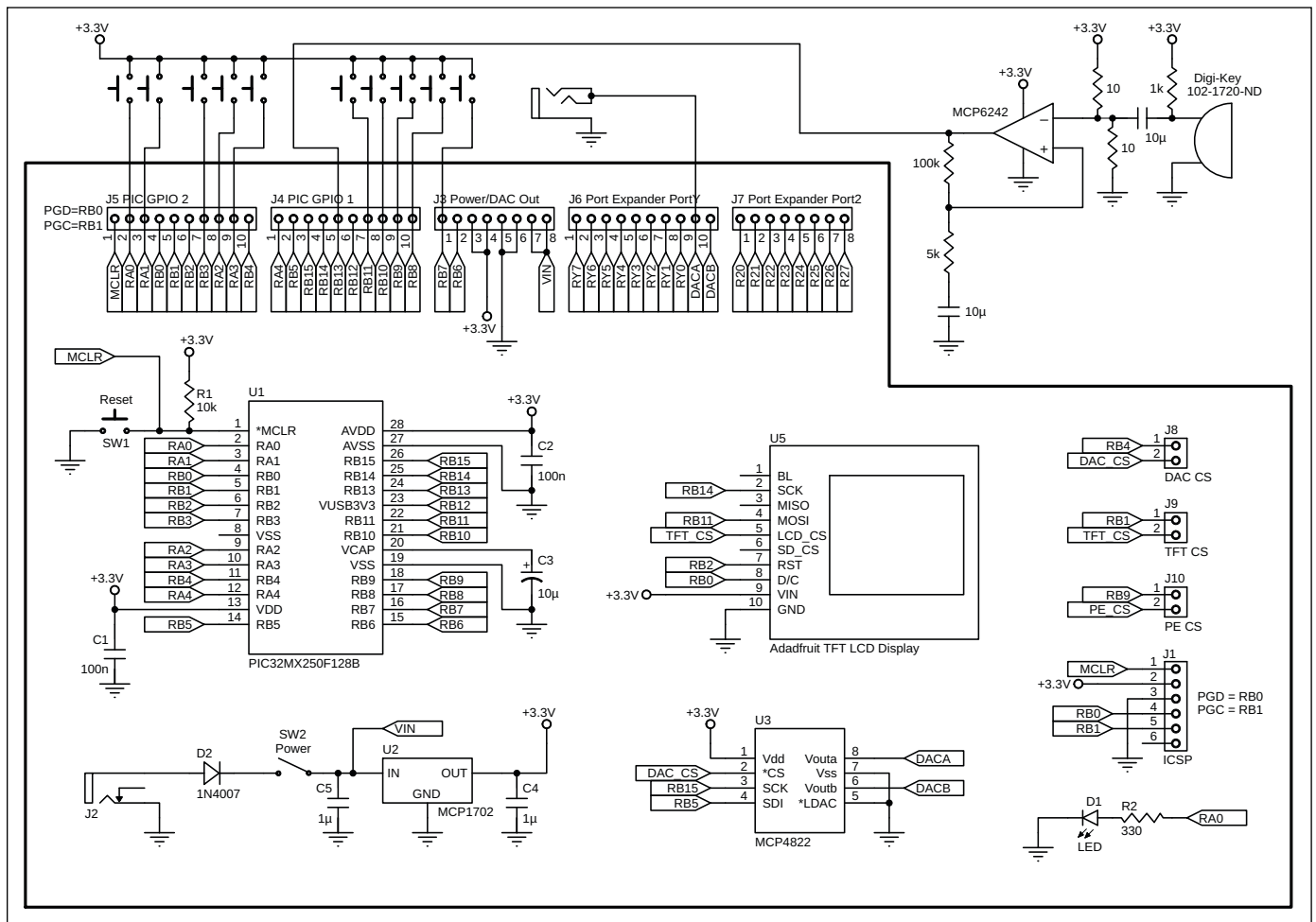


FIGURE 3

A complete diagram of our circuitry [2]. The microphone circuit is connected to the PIC32 through the ADC pin, and the audio output is through the DAC output. Each button used for the menu and playing sounds was connected to a different GPIO pin

The first challenge when writing the button thread was figuring out how to prevent mechanical bouncing from interfering with our readings. For this project, we designed a simple state machine that, after testing, we found worked reliably. In the thread, we read input from each button and stored the reading, then 30 ms later (enough time for the mechanical connections to settle), we read it again. If the new reading was the same as the original, we updated a variable to indicate the new state of the button, either pressed or unpressed. If the two readings were different, we considered the current state of the button unknown and left the variable unchanged. **Figure 4** shows a state diagram describing these transitions.

The second challenge in writing this thread was that some of the buttons needed to be treated differently than others. For example, when holding down a C key in piano mode, we wanted to play a sustained C. However, if we were in drum mode, we only wanted the drum sound to be played once—not on a loop. This also applied to the menu buttons. If they were held down, we only wanted to indicate that they had been pressed once. To implement this, we only changed the value of our pressed variable if the reading went from low to high, not high to high. This setup ensured that before a button could be pressed again, it first had to be released by the user, because we were focusing only on low to high transitions. This also required us to add a flag to indicate when a drum sound finished playing, which indicated that we could change the state to unpressed, however this was handled in the other component of playing sounds: the ISR. **Figure 5** shows a finite state machine (FSM) describing the buttons that we only wanted to read as being pressed once.

We configured our ISR to trigger on a timer at a rate of 8 kHz, the same sampling rate we used to sample our sounds. Each time through the ISR, we had to check which buttons were currently pressed, and then combine the values stored at the corresponding locations in the sound array to be sent to the DAC. Every time the ISR was triggered, we wanted to continue from the same place we left off on sounds that had to be played in the last pass. To do this, we kept a two-dimensional array of our current location in each of the sounds. When we played a sound, we then also had to increment the corresponding entry in the locations matrix. Again, we had to handle the drum sounds slightly different than the tonal sounds, because they should only be played once. This meant that after reaching the end of the drum sound, in addition to

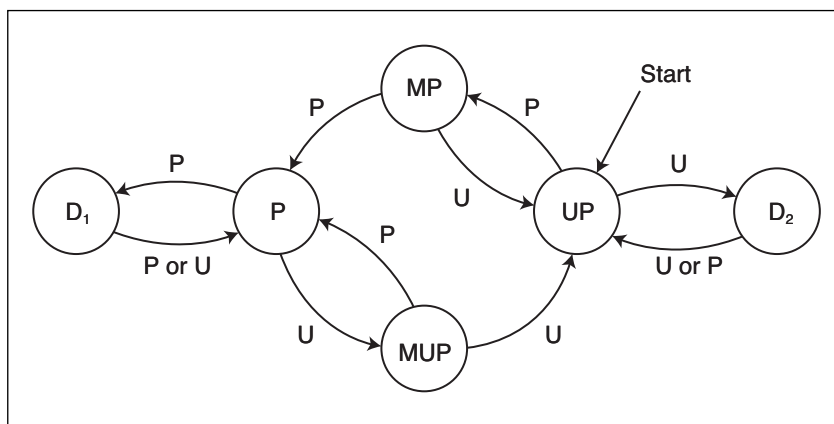


FIGURE 4

This state diagram describes the code we used for our buttons. They are states P (pressed), UP (unpressed), MP (maybe pressed), and MUP (maybe unpressed). The transitions P (pressed) and U (unpressed) occur when we read from one of the GPIO pins. The value of our pressed variable is 1 while the state is P, and is 0 when the state is UP.

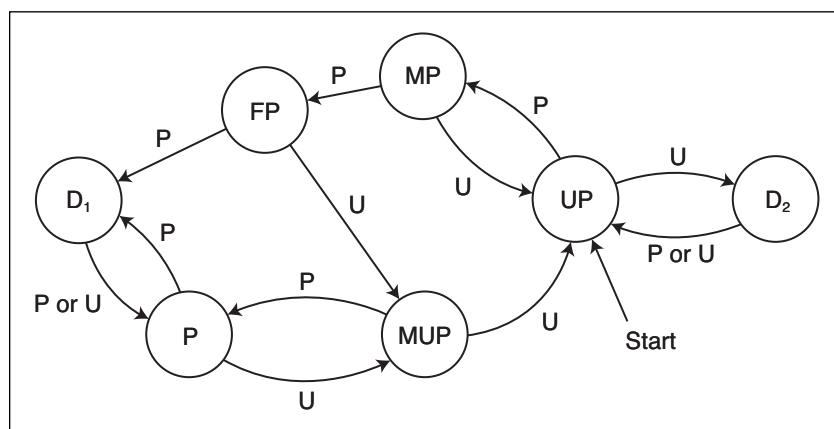


FIGURE 5

A state diagram describing buttons for which we only want to register a single press. This FSM includes an extra state, FP (first press), where the value of pressed is 1. While in either P or UP, pressed is 0. Transitions occur in the same way as for the other FSM.

ABOUT THE AUTHORS

Radhika Chinni (rpc222@cornell.edu) will be pursuing a Masters in Electrical and computer Engineering beginning in Fall 2019. She graduated from Cornell University in December 2018 with a B.S. in Electrical and Computer Engineering and a minor in Computer Science. She ultimately wants to pursue a working with embedded systems in robotics.

Brandon Quinlan (bmq4@cornell.edu) is senior at Cornell University, double majoring in Computer Science and Electrical and Computer Engineering. He hopes to pursue a career in computer architecture after completing a Masters of Engineering in 2019.

Raymond Xu (ryx2@cornell.edu) graduated with a Master of Engineering from Cornell University in Electrical and Computer Engineering in May 2018. He previously graduated with a Bachelor of Science (also in ECE at Cornell) in December 2017.

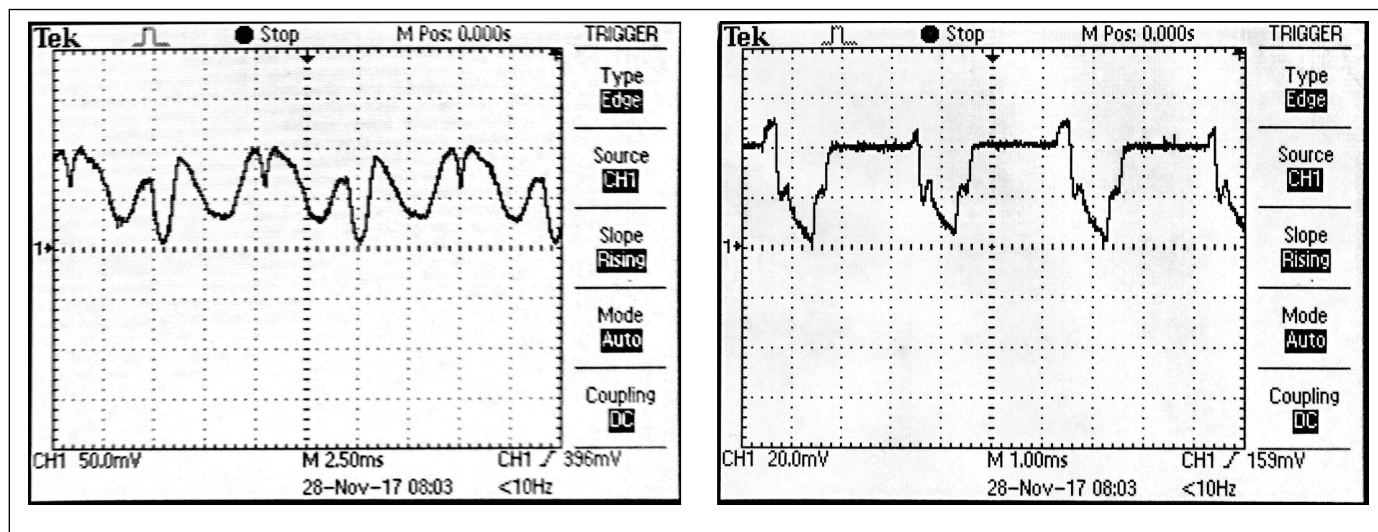


FIGURE 6

Shown here are oscilloscope readings from before (left) and after (right) the indexing issue was corrected.

resetting the location back to the beginning of that portion of the array, we set a flag indicating that the sound had been played to completion.

One other nuance that we had to deal with was that some of the sounds we took from YouTube had different volumes. We adjusted for this by shifting the bits of some sounds more than others to get a more uniform volume among instruments. After we were confident that our software was correct for playing notes in real time, we had to modify the code to implement recording and playback.

Recordings were stored in a recording array of predetermined size. In the ISR, we modified our array of recorded sounds and the output to the DAC, depending on the values of Recording and Playback buttons. The size of recordings were limited to approximately 2 s, due to space limitations. If we were recording, we added the resulting sound from pressing buttons to our recording array, and if we were in user mode, we added the results of reading from the microphone to our recording array. If we were currently playing back, we just needed to add the recorded wave to the sound output to the

DAC. When recording, if we reached the end of our recorded wave array, we stopped recording. If we were playing back and we reached the end of the recorded wave array, we looped back to the beginning of the array and repeated the playback.

TESTING AND RESULTS

Throughout the testing and debugging of our project, several changes were made to the original code to arrive at the final product previously described. One bug was that the recorded sounds would get louder as additional recordings were layered—even when the additional recordings were empty. We discovered the problem was that the recording wave was accidentally doubled when we were simultaneously recording and playing back. The most extensive testing we performed was with the sound files, which were loaded on the board and assessed by playing them out loud. One major error was that we had incorrectly indexed our individual instrument sounds. This was discovered through some oscilloscope testing (Figure 6).

Once we had the sounds indexed correctly, we found that the sound quality was poor, due to discontinuities in the waveforms. To correct this, we looked at each of the 24 tonal waveforms individually and smoothed them out by hand, so that there were no discontinuities from the end of the wave to the beginning. This meant that we also had to adjust other values near the endpoints. After this fine tuning, the sound quality was actually good for most notes, despite there being only a single period of the waveform. We also had an issue with too much audio played at the same time, and reaching the

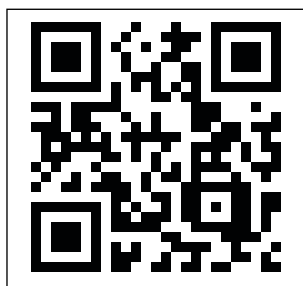


FIGURE 7

This QR code links to a YouTube video of the project.

For detailed article references and additional resources go to:

www.circuitcellar.com/article-materials

References [1] through [4] as marked in the article can be found there

RESOURCES

Digi-Key | www.digikey.com

Mathworks | www.mathworks.com

Microchip Technology | www.microchip.com

upper limit of our SPI, so we limited the device to playing eight tones at any given point.

As a whole, we were very pleased with the resulting project. It worked! An especially interesting outcome we observed was that 2 s is enough time to create interesting sounds. A YouTube video of us demonstrating the PIC32 Recording Studio is linked via the QR code in **Figure 7**. Most iPhone cameras can do this automatically but otherwise you may need an app. The video shows that 2 s is enough time to create a beat and baseline, while playing a live melody and harmony above it.


While dealing with the large amount of sound files and playback ability, we ran into a roadblock with the limited memory space on the MCU. We found that sampling at 44 kHz would make our keyboard sounds nice enough that the different instruments and notes would be recognizable. This sample rate limited our recording capability to approximately 0.25 s, and completely erased the possibility of incorporating a user mode where the user could add sounds as a microphone input. We decided this was unacceptable, because it did not give the user enough recording time to make any interesting sounds. Therefore, we decided that a trade-off was required. We lowered our sampling rate to 8 kHz, which allowed us to provide about 2 s of recording capabilities. At 120 beats per minute, this was just about 1 measure of music, which we found was just enough to allow the user to experience the overlay capabilities and build up a nice music track. This trade-off also had the benefit of allowing us to incorporate the microphone.

The only feature we did not have time to implement for this project was an advanced user mode, with which the user could store newly created sounds and play them back with the press of a button, so that such sounds could be continuously incorporated in different music tracks. Given that we already maxed out our memory with our recording storage, we couldn't add the advanced user mode without detracting from the recording.

THOUGHTS FOR THE FUTURE

In the future, if we were to repeat this project or improve upon it, we would love to incorporate this advanced user mode to expand the capabilities of the Recording Studio. We could do this by expanding the memory of the MCU—perhaps by using an external SRAM along with the existing flash memory. In addition to potentially longer recording times, this could allow us to include more sounds such as flats and sharps.

Because we found it difficult to keep tempo between different overlays when using our project, we also would find it useful to include a metronome option to keep tempo, so the user can layer tracks more confidently. Additionally, in another iteration of this project, we would hope to improve our user interface even more, perhaps by using a touch-screen display to avoid the hassle of the toggle buttons, and possibly by making our menu and interface appear more appealing.

As people who enjoy and appreciate music, from a usability perspective, we thoroughly enjoyed making music—albeit short lengths of music—with our Recording Studio once it was complete. We are very proud of our results and satisfied with what we were able to achieve. Any further improvements we make would only optimize our current setup and expand its potential. 

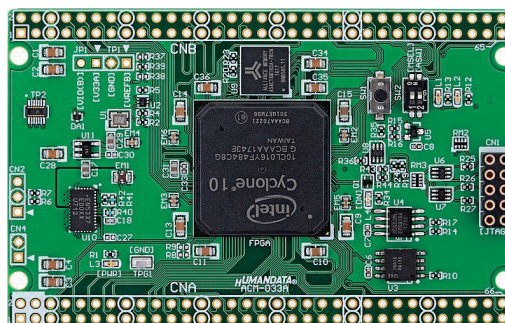
FPGA Boards from HUMAN DATA JAPAN

SAVING COST=TIME with readily available FPGA boards

- Basic and simple features, single power supply operation
- Free download technical documents before purchasing

INTEL ACM-033

Intel Cyclone 10 LP F484 FPGA board



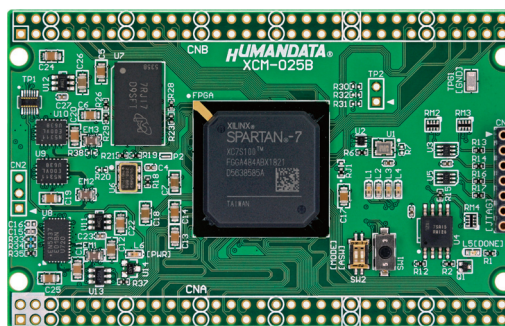
Cyclone 10 LP SDRAM SPI-Flash RoHS

SIZE : 3.386" x 2.126" (86 x 54 mm)

ACM-033 is an FPGA board with Intel high-performance FPGA Cyclone 10 LP. It's compact and very simple. 3.3V single power supply operation.

XILINX XCM-025

Xilinx Spartan-7 FGGA484 FPGA board



Spartan-7 DDR3 RoHS

SIZE : 3.386" x 2.126" (86 x 54 mm)

XCM-025 is an FPGA board with Xilinx high-performance FPGA Spartan-7. It's compact and very simple. 3.3V single power supply operation.

See all our products, A/D D/A conversion board, boards with USB chip from FTDI and accessories at:

www2.hdl.co.jp/CC19B



Drone Video Technologies are Flying High

Cameras, Boards and Kits



SPECIAL FEATURE

The video technologies available for today's drones continue to advance. New products and solutions are adding new intelligence, features and performance levels to enhance how video is captured and processed aboard both consumer and commercial drones.

By *Jeff Child*,
Editor-in-Chief

While drones can have a variety of sensor types, clearly video ranks the most common capability of today's consumer and commercial drones. Long gone are the days when placing an ordinary camera on a quadcopter style drone is a big deal. Today, drone cameras are highly sophisticated with designs evolved for drone use. In fact, some cameras embed so much processing, the term camera-computer is gaining steam.

Drone cameras are linked with board-level solutions that support multiple camera video streams and even perform AI-based intelligence functions aboard drones. Add those to the emergence of complete drone design platforms—that include camera and all—and it's clear that we're in a golden age for designing and developing powerful drones.

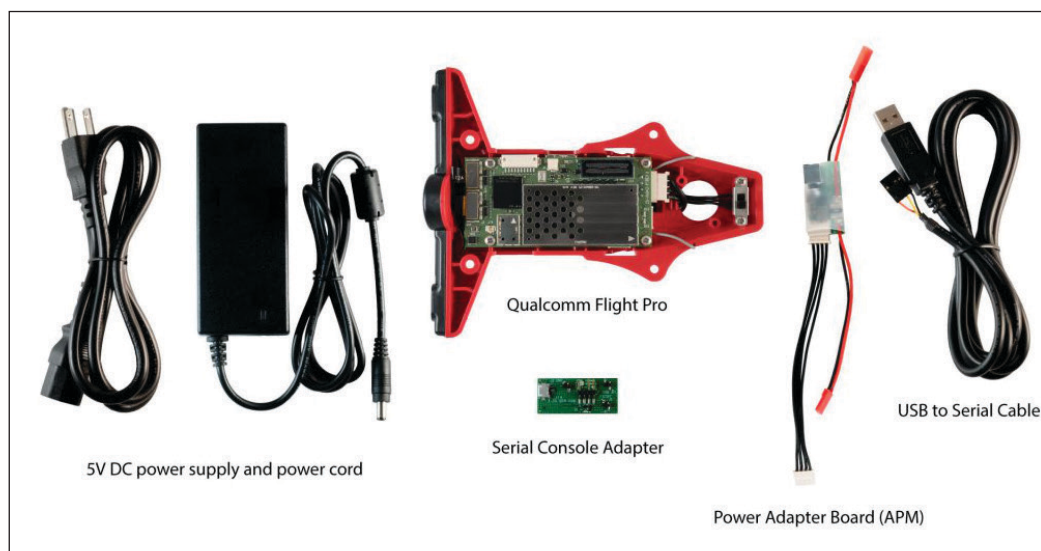
Video technology for drones spans a wide area of subjects including chip-level video processing, 4K HD video capture, image

stabilization, complex board-level video processing, drone-mounted cameras, hybrid IR/video cameras and drone development platforms. Over the past 12 months, vendors at the camera-, board- and system-level have been evolving their existing drone video technologies while also creating new innovative solutions.

COMPLETE REFERENCE PLATFORM

Starting at the platform level, drone development has definitely become easier these days, with companies both large and small providing complete drone development kits. One of these on the large company side is Qualcomm. In December, Qualcomm's partner Intrinsic Technologies announced it will distribute the Qualcomm Flight Pro reference platform (**Figure 1**). The platform is Qualcomm's latest optimized board and development kit targeted specifically for consumer drones.

The Qualcomm Flight Pro reference platform for consumer drones and robotics

**FIGURE 1**

The Qualcomm Flight Pro reference platform is Qualcomm's latest optimized board and development kit targeted specifically for consumer drones. The image on the opposite page shows the Qualcomm Flight Pro board with Snapdragon 820. The image here shows the complete kit.

applications is a follow-on to the Qualcomm Flight platform, which was previously launched under the name Snapdragon Flight. The Qualcomm Flight Pro steps up from a 2.2 GHz Qualcomm Snapdragon 801 with 4x Cortex-A53 like Krait cores to a Snapdragon 820 (APQ8096SG) with 4x higher-end Kryo cores—2x at 2.15 GHz and 2x at 1.6 GHz. The Snapdragon 820 also integrates an Adreno 530 GPU and Hexagon 680 DSP.

The system runs on a Linux 3.18 and Yocto/OpenEmbedded based stack with SDK, a Docker container and support for the Robot Operating System (ROS). An optional Qualcomm Navigator SDK supports autonomous, vision-supported Wi-Fi-based flight controls with advanced flight modes, built-in sensor calibration and automatic flight logging.

The Qualcomm Flight Pro is slightly larger than the original at a still very compact 75 mm x 36 mm, making room for 4x cameras driven by MIPI-CSI interfaces. The kit includes a pair of forward-facing stereo-vision cameras using Omnivision OV7251 black and white VGA sensors by way of a Sunny GP161C module, as well as a forward-facing, 13-megapixel, 4K-at-30-fps camera with a Sony IMX214 color sensor in a KLT Module. There's also a downward-facing camera with a black and white VGA OV7251 sensor via a Sunny MD102A module.

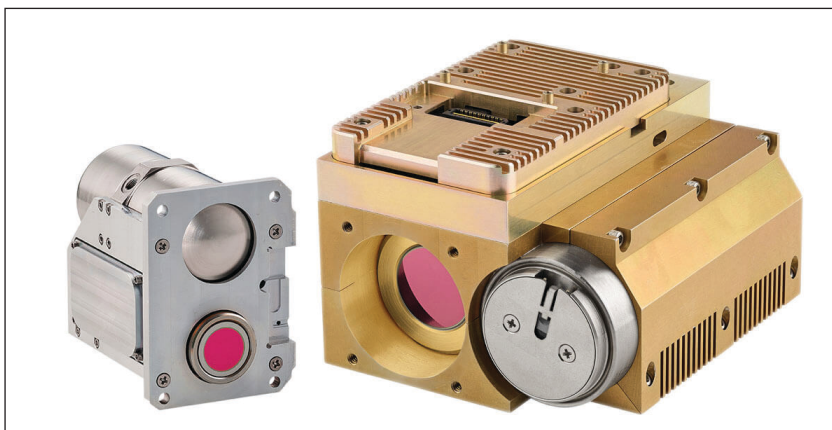
The Pro board includes 4 GB LPDDR4, a microSD slot and 32 GB UFS 2.0 (HS-G3 1-Lane) storage. Other features include a Qualcomm QCA6174 wireless module with 802.11ac 2x2 MIMO and Bluetooth 4.2 (with antenna mounts), as well as a Qualcomm WGR7640 GNSS location chip that supports an optional U-blox GPS module. The SBC is further equipped with an IMU with gyroscope, accelerometer, compass

(Dual InvenSense MPU9250) and a barometer/pressure sensor (Bosch BMP280).

More recently, in late February, Qualcomm and Thundercomm launched their Robotics RB3 Platform" that includes an octa-core Snapdragon 845 via a new "DragonBoard 845c" 96Boards SBC and tracking cameras. While that platform appears to be marketed toward terrestrial robots, Qualcomm did tell us that it can also be used for developing drones.

CAMERAS, CAMERAS, CAMERAS

Switching to the camera side of drone video, the latest crop of drone-based camera systems includes a wide range of solutions, some focusing on photo and video quality, others on new features and capabilities. For its part, in December, FLIR Systems announced three Neutrino midwave infrared

**FIGURE 2**

The Neutrino LC (left) is FLIR's first High Operating Temperature (HOT) MWIR camera core and the first model in the SWaP+C series. The Neutrino QX, with more than 3.1 megapixels, is FLIR's highest resolution MWIR camera core.



FIGURE 3

The Pensar is a dual spectrum digital vision platform that does real-time data analysis using a miniaturized Nvidia embedded processor with 1.5 teraflops of power. The dual spectrum is provided by a built-in Sony 30x full HD optical zoom camera with 1920 x 1080 resolution and a 30 fps Boson FLIR integrated thermal camera.

(MWIR) camera cores. These include the small, lightweight FLIR Neutrino LC and two FLIR Neutrino Performance series cores, the SX12 and QX (**Figure 2**). The latest models expand the FLIR Neutrino cooled camera core family for commercial, industrial and defense OEMs and system integrators.

CAMERA MEETS SWaP+C NEEDS

The Neutrino LC is FLIR's first High Operating Temperature (HOT) MWIR camera core and the first model in the SWaP+C (Size, Weight, Power and Cost) series. As the smallest, lightest weight and lowest power consuming Neutrino model available, the LC can be integrated with smaller drones and allow drone operators to fly longer. With HOT technology, Neutrino starts imaging two times faster than previous models, allowing optical gas imaging professionals to detect gases faster. Additionally, the Neutrino's longer operational lifetime allows installation in security applications where maintenance access is restricted, difficult or costly.

The two new Neutrino Performance series products, the Neutrino SX12 and the Neutrino QX, offer the highest-resolution MWIR performance from FLIR. The Neutrino SX12 produces high-definition (HD) thermal imaging video, while Neutrino QX, with more than 3.1 megapixels, is FLIR's highest resolution MWIR core. Both Neutrino Performance models provide crisp imagery at long distances while maintaining a wide field of view and are well suited for ground-based or airborne intelligence, surveillance, reconnaissance (ISR) and counter-drone solutions. The Neutrino SX12, QX and LC are dual-use camera cores for commercial, industrial and defense products and are classified under the U.S. Department of Commerce Export Administration Regulations

as Export Control Classification Number 6A003.b.4.a.

INTELLIGENT CAMERA

One area of innovation in drone cameras is adding more intelligence to them. Exemplifying this trend, in August last year Aerialtronics launched a new version of its Pensar camera-computer driven by artificial intelligence. According to the company, Pensar is one of the world's first platforms with dual spectrum digital vision that allows real-time analysis of images or data (**Figure 3**). Infinitely customizable, it can be mounted on a professional drone, mobile robot or used as an independent camera. The dual spectrum is provided by a built-in Sony 30x full HD optical zoom camera with 1920 x 1080 resolution and a 30 fps Boson FLIR integrated thermal camera. The Pensar is 112.5 mm x 98.5 mm x 67.5 mm in size and weighs 672 g.

Pensar does real-time data analysis using a miniaturized Nvidia embedded processor with 1.5 teraflops of power. Its computing power, accelerated by the Nvidia Jetson TX1 GPU processor in the Nvidia Jetson module, enables it to detect, recognize, analyze and classify objects or people in real time. Simultaneous data acquisition and processing allows for immediate decision making.

Pensar's integrated camera with a 30x optical zoom makes it possible to spot very small details. Also embedded in Pensar is a FLIR thermal camera used to identify heat sources and determine their temperature. The streams from these two cameras, recorded simultaneously, help optimize image analysis in day and night time and bad weather conditions.

This camera-computer can be customized and adapted for multiple applications: surveillance, inspection, public security and anti-terrorist operations, search and rescue and so on. It's equipped with a system for facial recognition, object recognition such as license plates, animal recognition and similar tasks. A digital "privacy mask" can be integrated into the images to guarantee confidentiality and anonymity. The intelligent platform comes with an Ubuntu Linux Open Source operating system that allows system integrators to customize it to suit their needs. Pensar is compatible with open source libraries such as Google's Tensor Flow.

FANCY PHOTOGRAPHY

As today's drone cameras have evolved, they're now offering many very sophisticated

features for high-end photography. Along those lines, Lucint Systems' Lucint12 camera basically provides a complete aerial image collection system in a small, rugged, low-power box (**Figure 4**). Lucint12 is a 12-megapixel high-quality color or monochrome image sensor with all the controls, metadata, processing and storage needed for a complete system. The unit also features a powerful built-in GPU processor to handle real time georeferencing, image preprocessing, and custom user algorithms.

Designed for photogrammetry the camera features large pixels that result in excellent dynamic range. It has lightweight, high-quality Micro Four Thirds lenses and captures metadata and precise GPS timestamp with each frame. Lucint12 provides a number of automated functions including auto-exposure designed for aerial capture ensures consistent exposures, auto-focus optimized for aerial, automotive or ground installations, and auto-trigger at fixed rate, percent image overlap, or external trigger.

The Lucint12 is complete system with rugged and reliable design features. Its global electronic shutter has no moving parts and no rolling shutter distortion. Industrial components extend operating temperature range. The unit has a fully sealed housing for harsh operating environments. The Lucint12 integrates an internal GPS to record image capture location, on-board mSATA-based image storage up to 1TB. Users can configure settings over Wi-Fi by phone or tablet.

CAMERA WITH V-SLAM TECH

Qualcomm isn't the only big chip vendor with a hand in drone technology. Intel's latest drone video offering is its RealSense Tracking Camera T265. Announced in January, the T265 uses proprietary visual inertial odometry simultaneous localization and mapping (V-SLAM) technology and is suited for applications that require a highly accurate and low-latency tracking solution, including robotics, drones, augmented reality (AR) and virtual reality. V-SLAM uses a combination of cameras and Inertial Measurement Units (IMU) to navigate in a similar way, using visual features in the environment to track its way around even unknown spaces with accuracy.

At the heart of the T265 is the Intel Movidius Myriad 2 vision processing unit (VPU), which directly handles all the data processing necessary for tracking on the machine (**Figure 5**). According to Intel,



FIGURE 4

Lucint12 is a 12-megapixel high-quality color or monochrome image sensor with all the controls, metadata, processing and storage needed for a complete system.

the T265 is good for applications where tracking the location of a device is important, especially in locations without GPS service, such as warehouses or remote outdoor areas where the camera uses a combination of known and unknown data to accurately navigate to its destination. The T265 is also designed for flexible implementation and can be easily added to small-footprint mobile devices like lightweight robots and drones, as well as for connectivity with mobile phones or AR headsets.

The T265 uses inside-out tracking, which means the device does not rely on any external sensors to understand the environment. Unlike other inside-out tracking solutions, the T265 delivers 6-degrees-of-freedom (6DoF) inside-out tracking by gathering inputs from



FIGURE 5

Intel's RealSense Tracking Camera T265 uses proprietary visual inertial odometry simultaneous localization and V-SLAM technology. V-SLAM uses a combination of cameras and Inertial Measurement Units (IMU) to navigate in a similar way, using visual features in the environment to track its way around unknown spaces with accuracy.

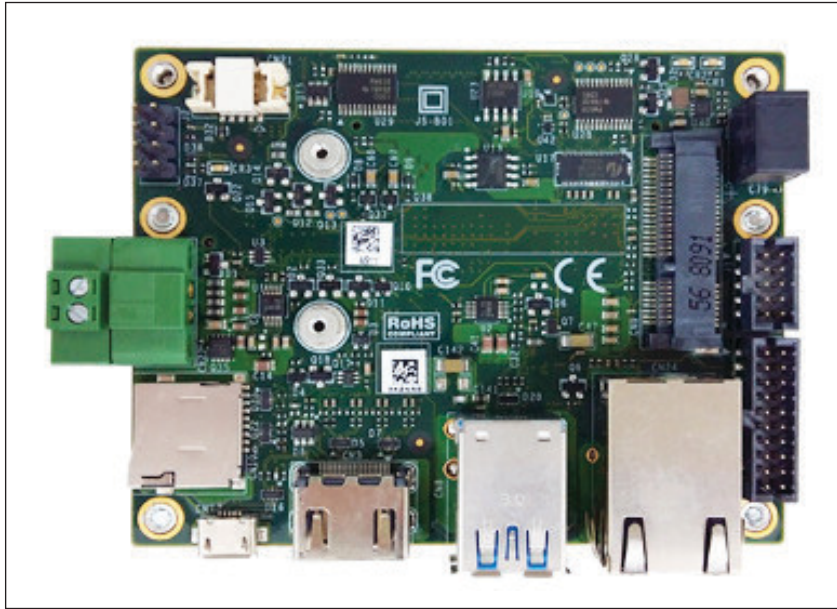


FIGURE 6

The ACE-N310 enables 360-degree surrounded view applications in vehicles, drones and robots. With the help of the Jetson modules' AI-enabled Pascal GPU, the ACE-N310 lets you build multi-visual intelligent systems with advanced on-premises analytics.

two onboard fish-eye cameras, each with an approximate 170-degree range of view. The V-SLAM systems construct and continually update maps of unknown environments and the location of a device within that environment.

Because all position calculations are performed directly on the device, tracking with the T265 is platform independent and allows the T265 to run on very low-compute devices. The only hardware requirements are sufficient non-volatile memory to boot the device and a USB 2.0 or 3.0 connection that provides 1.5 W of power. The camera measures 108 mm x 25 mm x 13 mm in size and weighs only 55 g.

MULTI-CAMERA SUPPORT

In August last year, Aetina launched its first carrier board for Nvidia's Jetson TX1 and

TX2 modules that supports up to 6x cameras and offers -40°C to 85°C support. The ACE-N310 enables "360-degree surrounded view application in vehicles, drones, robots, surveillance and automation and intelligent systems at the edge," says Aetina. With the help of the Jetson modules' AI-enabled Pascal GPU, the ACE-N310 lets you build multi-visual intelligent systems with advanced on-premises analytics and inference, according to the company (**Figure 6**).

The module integrates its iNAVI Linux distribution, which adds customizable security, system recovery and backup features. iNAVI is also available with the ACE-N310 and other Aetina Jetson carrier boards, which similarly support the TX1, TX2 and TX2i. The 87 mm x 70 mm board is most closely comparable to the ACE-N510 carrier, which has an 87 mm x 50 mm footprint that matches that of the TX2 and TX2i modules themselves. Aetina also offers the Nano-ITX (120 mm x 120 mm) form factor ACE-N261 and ACE-N622 boards.

The ACE-N310 can be configured with up to 12x lanes of MIPI-CSI connectors through CSI-II or FPD-LINK III extension modules. This enables the connection of 6x 2-lane, 2-megapixel cameras with 1080p/30fps resolution or 3x 4-lane 4K cameras. Aetina offers a variety of Sony IMX based, HD resolution MIPI-CSI camera modules to choose from, as well as an optional, FPC-connected ACE-CAM6C camera board with 6x CSI-2 cameras. There are also "certified" mini-PCIe based I/O modules including dual isolated GbE and PoE add-ons and a 4x USB 3.0 option, all with 0 to 70°C and -40°C to 85°C support.

Other ACE-N310 features include HDMI, GbE, micro-USB 2.0 and 2x USB 3.0 host ports. Onboard interfaces include RS-232, I²C and 5x GPIO, as well as 2x CAN Bus connections that work only with the Jetson TX2 and TX2i. A mini-PCIe slot supports PCIe and mSATA, and there's a 9-19 VDC input. Other options include fan and heatsink add-ons, cable kits, and a 100-240 V, 60 W 12V/5A adapter.

BOARD FOR SMALL DRONES

Last summer Gumstix released a version of its Aerocore 2 drone control board that runs Linux on Nvidia's Jetson TX2. The Aerocore 2 drone control board arrived in 2014 and was followed in 2016 by a more advanced version that swapped out the original's Gumstix Overo module for a DragonBoard 410C SBC. This most recent 2018 board—dubbed Aerocore 2 for Nvidia Jetson—works with Nvidia's

RESOURCES

Aerialtronics | www.aerialtronics.com

Aetina | www.aetina.com

FLIR Systems | www.flir.com

Gumstix | www.gumstix.com

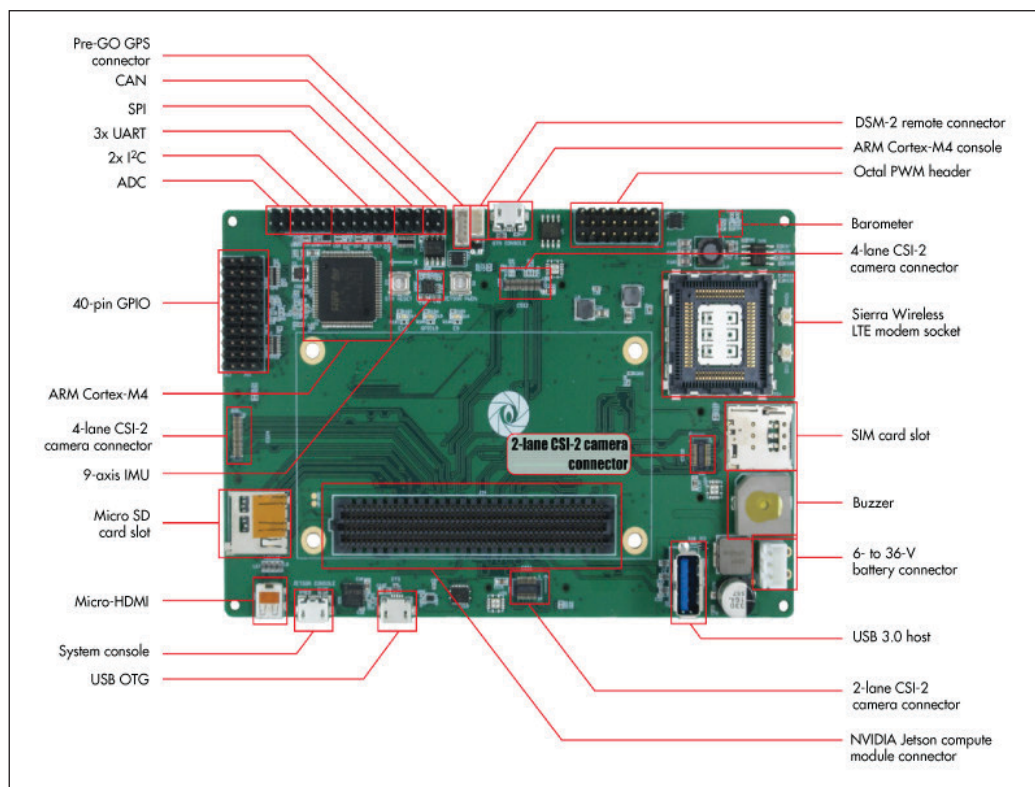
Intel | www.intel.com

Intrinsyc Technologies | www.intrinsyc.com

Lucint Systems | www.lucintsystems.com

Nvidia | www.nvidia.com

Qualcomm | www.qualcomm.com

**FIGURE 7**

Aerocore 2 for Nvidia Jetson works with Nvidia's Jetson TX1 and Jetson TX2 modules and can be customized in Gumstix's Geppetto online design service.

Jetson TX1 and Jetson TX2 modules and can be customized in Gumstix's Geppetto online design service (**Figure 7**).


The Jetson TX2 is equipped with dual high-end "Denver 2" Arm cores and 4x Arm Cortex-A57 cores. The 256-core Pascal GPU with CUDA libraries for running AI and machine learning algorithms offer the potential for improved image recognition applications in drones and robotics. The Aerocore 2 is best suited for small drones called micro-aerial vehicles (MAVs), but it can also be used for larger drones, robots and other image processing applications.

The Jetson TX2 module provides the Aerocore 2 for Nvidia Jetson with 8 GB of LPDDR4 RAM, 32GB of eMMC 5.1 and 802.11ac Wi-Fi and Bluetooth. The Aerocore 2 carrier board adds an STMicroelectronics STM32F427 Cortex-M4 chip clocked at 168 MHz. This MCU is pre-loaded with the open source NuttX RTOS and APM-based PX4 firmware for real-time drone autopilot operation. It should also work with PX4-compatible projects such as QGroundControl and MAVLink. Because the Jetson boards are modules rather than an SBC, the Aerocore 2 carrier board has added more ports and other features to compensate. The board ships with a microSD slot, as well as micro-HDMI, USB 3.0 host, micro-USB OTG and micro-USB device ports.

There are two separate 4-lane MIPI-

CSI-2 interfaces that support Gumstix's \$30 Caspa 4K cameras, which are built on Sony's 13-megapixel IMX214 AF Camera sensor and support 4208 x 3120-pixel stills and 4K video at 30 fps. In addition, you get a pair of 2-lane CSI-2 connectors for 5-megapixel cameras with 2592 x 1944 resolution. The Aerocore 2 board is capable of driving 4x cameras with HD or higher resolution simultaneously.

Like other Aerocore boards and most other Gumstix boards, the Aerocore 2 for Nvidia Jetson can be customized with the Gumstix Geppetto D20 online development platform. The Geppetto drag-and-drop GUI interface lets developers add network connections or I/O, as well as create multiple projects and compare alternative designs for features and costs. Geppetto supplies free automated documentation on demand with all saved designs. The service lets you develop custom BSPs and go straight from a design to an order in one session, with 15-day manufacturing turnaround.

We're obviously far past the days when commercial drone video was just a straightforward proposition of mounting a camera on a drone. Today there are many technology options for building drones for a variety of applications and mission types. Camera, board and system vendors are keeping pace with these trends, feeding the demands of this dynamic and growing market. 

Low-Power Wireless Solutions Feed IoT Edge Needs

Integration Meets Innovation

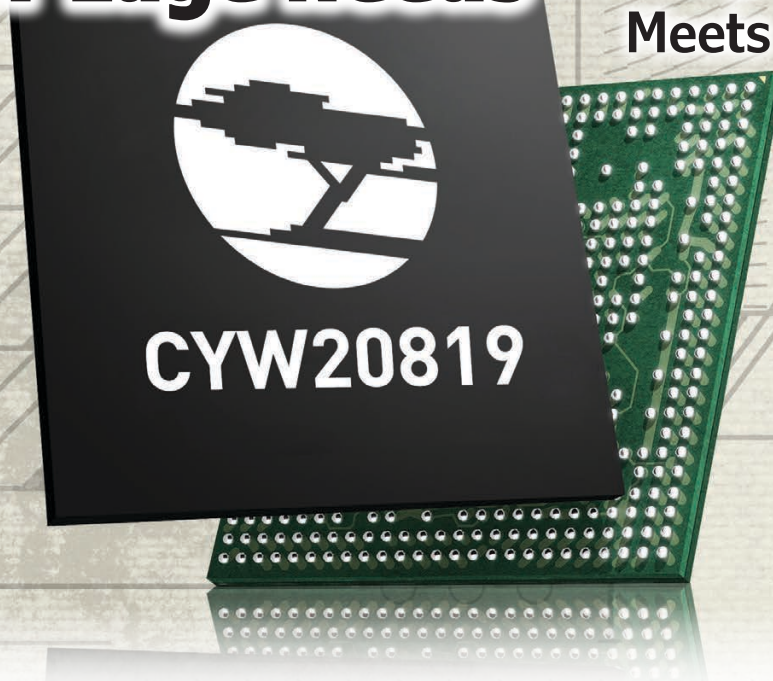


FIGURE 1

The CYW20819 Bluetooth/ BLE MCU has the ability to maintain Serial Port Profile (SPP) protocol connections and Bluetooth mesh connections simultaneously. The device provides 123 days of battery life from a CR2032 coin cell battery.

By **Jeff Child**,
Editor-in-Chief

TECH SPOTLIGHT

Escalating demand for IoT solutions is fueling the need for products and technology that perform wireless communication with low-power edge devices. Using technologies including BLE, LoRaWAN and others, embedded system developers are searching for ways to get efficient IoT connectivity while drawing as little power as possible.

Enabling various nodes of an IoT implementation to communicate can involve a number of wired and wireless network technologies. Because most IoT systems can't be completely hardwired end to end they must rely on a variety of wireless technologies including everything from device-level technologies to Wi-Fi to cellular networking.

Fortunately, today's IoT system developers have a rich set of wireless solutions from which to choose from. And these can implement communication from the gateway and the device side using a variety of wireless IoT solutions in both module and chip form. This article focuses the low power subset of these technologies such as LoRaWAN and Bluetooth Low Energy (BLE). Over the last 6 months, technology vendors have announced several design wins and innovative products based on those technologies.

LOW POWER BLUETOOTH

Though not specifically created for the IoT edge market, Bluetooth LE (low energy), also called BLE, is a wireless personal area network technology designed and marketed by the Bluetooth SIG. It consumes only a fraction of

the power of classic Bluetooth radios. While it was created for applications in healthcare, fitness, security and home entertainment, Bluetooth LE offers connectivity for any low power device especially systems that need to operate these devices for more than a year without recharging. This makes it ideally suited for the IoT.

Same as the classic Bluetooth, the range of the BLE radio can be optimized according to application. While most Bluetooth devices on the market today offer the basic 10 m range, there is no limit imposed by the specification. Manufacturers may choose to optimize the range to 200' and beyond, particularly for sensor applications where longer range is a necessity.

The latest BLE microcontroller (MCU) offerings from Cypress Semiconductor are its two low-power, dual-mode Bluetooth 5.0 and BLE MCUs that include support for Bluetooth mesh networking for the IoT. Announced as sampling in February, the new CYW20819 and CYW20820 MCUs each provide simultaneous Bluetooth 5.0 audio and BLE connections.

The CYW20819 Bluetooth/BLE MCU has the ability to maintain Serial Port Profile (SPP) protocol connections and Bluetooth mesh connections simultaneously (**Figure 1**).

The CYW20820 offers the same features and integrates a power amplifier (PA) with up to 10 dBm output power for long-range applications up to 400 m and whole-home coverage. This provides classic Bluetooth tablet and smartphone connections while enabling a low-power, standards-compliant mesh network for sensor-based smart home or enterprise applications.

Both MCUs embed the Arm Cortex-M4 core. It enables operation at 60% lower active power for connected 200-ms beacons compared to current solutions—delivering up to 123 days of battery life from a CR2032 coin cell battery. Previously, users needed to be in the immediate vicinity of a Bluetooth device to control it without an added hub. Using Bluetooth mesh networking technology, combined with the high-performance integrated PA in the CYW20820, the devices within a network can communicate with each other.

LOW POWER AND REAL TIME

The latest Bluetooth offering from STMicroelectronics (ST) are its STM32WBx5 dual-core wireless MCUs. Announced in February, the devices come with Bluetooth 5, OpenThread and ZigBee 3.0 connectivity combined with ultra-low-power performance. Fusing features of ST's STM32L4 Arm Cortex-M4 MCUs and in-house radio managed by a dedicated Cortex-M0+, the STM32WBx5 is power-conscious yet capable of concurrent wireless-protocol and real-time application execution (**Figure 2**). It is well suited to remote sensors, wearable trackers, building-automation controllers, computer peripherals, drones and other IoT devices.

Security features of the STM32WBx5 MCUs include Customer Key Storage (CKS), Public Key Authorization (PKA), and encryption engines for the radio MAC and upper layers. The MCUs have up to 1 MB of on-chip flash and a Quad-SPI port for efficient connection to external memory, if needed. Additional features include crystal-less Full-Speed USB, 32 MHz RF oscillator with trimming capacitors, a touch-sense controller, LCD controller, analog peripherals and multiple timers and watchdogs. The balun for antenna connection is also integrated.

Leveraging ultra-low-power technologies of the STM32L4 line, STM32WBx5 MCUs feature multiple power-saving modes including 13 nA shutdown mode, adaptive voltage scaling, and the adaptive real-time (ART) accelerator to maximize energy efficiency and ensure long-lasting performance in self-powered applications. The integrated radio transmitter is optimized for high RF performance and low power consumption to maximize battery runtime. The RF output power is programmable up to +6 dBm in 1 dB

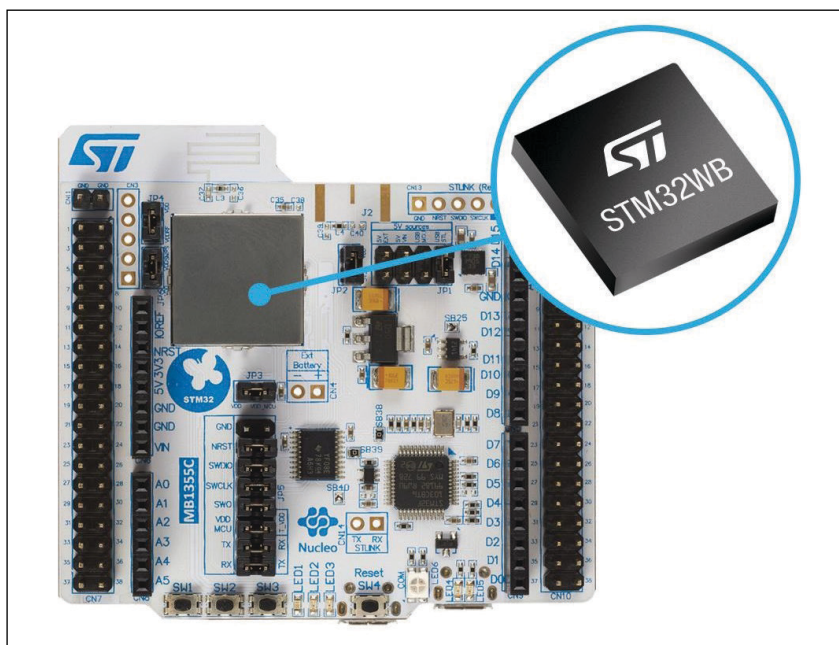


FIGURE 2

Fusing features of ST's STM32L4 Arm Cortex-M4 MCUs and in-house radio managed by a dedicated Cortex-M0+, the STM32WBx5 is power-conscious yet capable of concurrent wireless-protocol and real-time application execution.

increments, and the MCU draws only 5.2 mA when transmitting at 0 dB. Receive sensitivity is -96 dBm for BLE communication at 1mbps. Designed for a link budget of 102 dB, the radio ensures robust communication over long connection distances and includes support for an external Power Amplifier (PA).

IoT MODULE EXAMPLE

In a recent BLE design in example, in January Nordic Semiconductor announced that Nanopower selected Nordic Semiconductor's nRF52832 BLE System-on-Chip (SoC) to provide the wireless connectivity for its nP-BLE52 module, designed for developers of IoT applications with highly restricted power budgets (**Figure 3**).

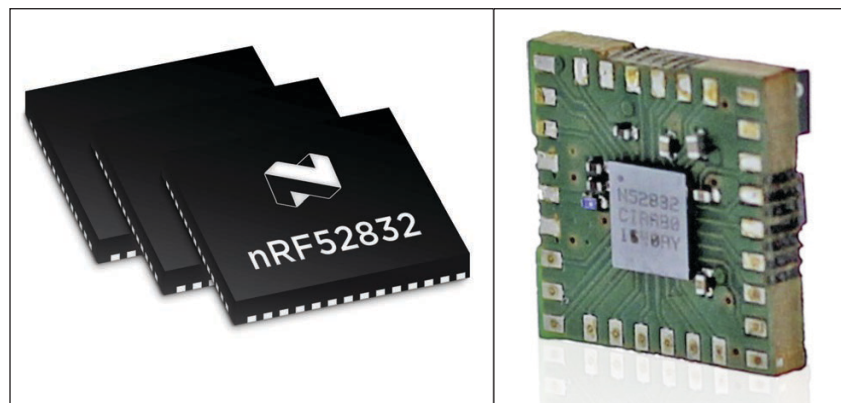


FIGURE 3

Nanopower selected Nordic Semiconductor's nRF52832 BLE SoC (left) to provide the wireless connectivity for its nP-BLE52 module (right), designed for developers of IoT applications with highly restricted power budgets.

The nP-BLE52 module employs a proprietary power management IC—integrated alongside Nordic’s nRF52832 Wafer-Level Chip Scale Package (WL-CSP) SoC in a System-in-Package (SiP). It enables the SoC’s power consumption in sleep mode to be reduced to 10 nA, making it well suited for IoT applications where battery life is critical by potentially increasing cell lifespan 10x.

In active mode, the nRF52832 SoC runs normally. The SoC has been engineered to minimize power consumption with features such as the 2.4 GHz radio’s 5.5 mA peak RX/TX currents and a fully-automatic power management system. Once the Nordic SoC has completed its tasks, it instructs the nP-BLE52 to put it to sleep and wake it up again at the preset time. The nP-BLE52 then stores the Nordics SoC’s state variables and waits until the nRF52832 SoC needs to be powered up again. On wake-up, the device uploads the previous state variables, allowing the Nordic SoC to be restored to the same operational state as before the power was cut. The SoC’s start-up is much more rapid than if it was activated from a non-powered mode.

The nP-BLE52 module also features a low power MCU that can be set to handle external

sensors and actuators when the Nordic chip is switched off. In this state, the module still monitors sensors and buffer readings and can trigger wake-ups if these readings are above predetermined thresholds, while consuming less than 1 μ A. The nP-BLE52 also integrates an embedded inertial measurement unit (IMU). The module’s power management is controlled through a simple API, whereby the user can predetermine the duration of the Nordic SoC’s sleep mode, set the wake-up time and date parameters and select pins for other on/off triggers.

LoRaWAN GAINING MOMENTUM

The LoRaWAN specification is a Low Power, Wide Area (LPWA) networking protocol designed to wirelessly connect battery operated “things” to the Internet in regional, national or global networks. Managed by the LoRa Alliance, LoRaWAN targets key IoT requirements such as bi-directional communication, end-to-end security, mobility and localization services.

The networking architecture of LoRaWAN is deployed in a star-of-stars topology in which gateways relay messages between end-devices and a central network server. Gateways are connected to the network server via standard IP connections and act as a transparent bridge, simply converting RF packets to IP packets and vice versa.

The wireless communication takes advantage of the long-range characteristics of the LoRa physical layer, allowing a single-hop link between the end-device and one or many gateways. All modes are capable of bi-directional communication, and support is included for multicast addressing groups to make efficient use of spectrum during tasks such as Firmware Over-The-Air (FOTA) upgrades or other mass distribution messages.

In November last year, Microchip Technology introduced a highly integrated LoRa SiP family with an ultra-low-power 32-bit MCU, sub-GHz RF LoRa transceiver and software stack (**Figure 4**). The SAM R34/35 SiPs come with certified reference designs and interoperability with major LoRaWAN gateway and network providers. The devices also provide ultra-low power consumption in sleep modes, enabling extended battery life in remote IoT nodes.

Most LoRa end devices remain in sleep mode for extended periods of time, only waking up occasionally to transmit small data packets. Powered by the ultra-low-power SAM L21 Arm Cortex-M0+ based MCU, the SAM R34 devices provide sleep modes as low as 790 nA to significantly reduce power consumption and extend battery life in end applications.

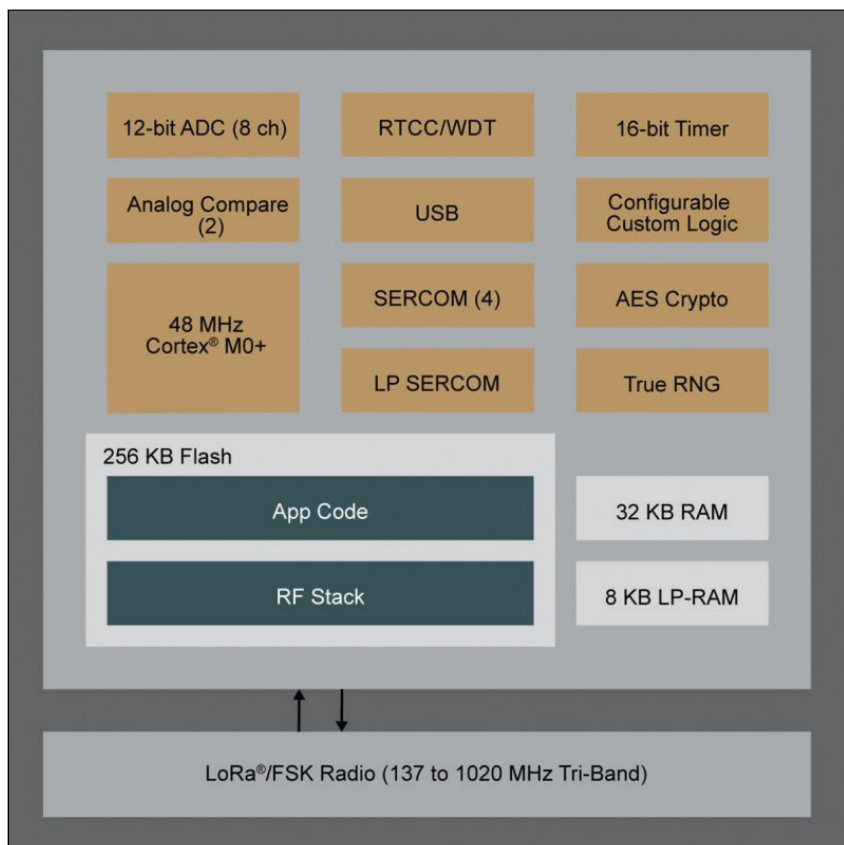


FIGURE 4

The SAM R34/35 is a highly integrated LoRa System-in-Package (SiP) family with an ultra-low-power 32-bit MCU, sub-GHz RF LoRa transceiver and software stack. The chips come with certified reference designs and interoperability with major LoRaWAN gateway and network providers.

Highly integrated in a compact 6 mm x 6 mm package, the SAM R34/35 family is well suited for a broad array of long-range, low-power IoT applications that require small form factor designs and multiple years of battery life.

The SAM R34/35 family is supported by Microchip's LoRaWAN stack, as well as a certified and proven chip-down package that enables embedded systems developers to accelerate the design of RF applications with reduced risk. With support for worldwide LoRaWAN operation from 862 MHz to 1,020 MHz, developers can use a single part variant across geographies, simplifying the design process and reducing inventory burden. The SAM R34/35 family supports Class A and Class C end devices as well as proprietary point-to-point connections.

LoRaWAN DESIGN WINS

LoRaWAN continues to gain momentum, with several design wins based on the technology announced over the past several months. In an example along those lines, in November last year Semtech announced that EasyReach Solutions, an Indian startup specializing in smart IoT solutions for industrial applications, has incorporated Semtech's LoRa devices and wireless radio frequency technology (LoRa Technology) into its industrial and smart vehicle monitoring products. EasyReach's LoRa-enabled sensors have been developed to include electrical current testing, temperature reading and GPS capabilities. All sensors are compatible with the LoRaWAN protocol and have been verified for GPS tracking ability over 8 km line of sight.

According to EasyReach, the LoRa Technology allows the company to remotely monitor its equipment and vehicles in new ways and to more intelligently manage its industrial resources. Meanwhile, the flexible capabilities of the sensors allow the solution to scale to its needs. EasyReach's LoRa-based applications for smart industry include sensors for steam traps, concrete mixers, forklifts, diesel tankers, back hoes, water meters and trucks.

Semtech revealed another design win for its LoRa in December, announcing that Lemonbeat, an IoT solution provider, has integrated Semtech's LoRa devices and technology into its smart metering solutions for easier reading and collection of utility usage. Lemonbeat's LoRa-connected smart meters work by using embedded LoRa-based IoT technology to connect the meter to their own purpose-built receiver units.

Using this connectivity, meters send data through multiple floors in bigger buildings or all way into the street, where network operators conveniently collect the data



FIGURE 5

The SAM R30 is a small IEEE 802.15.4-compliant module that combines an ultra-low-power MCU with a sub-GHz radio, providing long-lasting battery life in wireless-networked sensors.

without having to enter the building. Using the meters' other radio frequency, Lemonbeat Radio, meters provide customers accurate data on their energy consumption. With a third-party application, individuals can view and analyze this data, and change their habits accordingly.

PROTOCOL MEETS INDUSTRIAL NEEDS

At the device-level, the ISM 802.15.4 is a popular standard for lower power kinds of IoT devices gear. The IEEE standard 802.15.4 is designed for low-cost, low-speed communication between devices. 802.15.4 is also well suited for data sharing of devices in close proximity that have very limited infrastructure. 802.15.4 is the basis for established industrial network schemes like Zigbee and can be used with protocols like 6LoWPAN to add higher layer functions using IP technology.

For its part, in February Microchip Technology announced what it claims is the industry's smallest IEEE 802.15.4-compliant module that combines an ultra-low-power MCU with a sub-GHz radio, providing long-lasting battery life in wireless-networked sensors. At half the size of the next smallest module on the market, the SAM R30 module meets the needs of space-conscious designs such as home automation sensors and controls (**Figure 5**).

Based on IEEE 802.15.4, the SAM R30 module supports proprietary networks that can be easily customized and configured. This is ideal for applications where interoperability is not desired due to their inherent vulnerability to remote attacks, such as alarm systems, building automation, smart cities and industrial sensor networks.

A key advantage of an IEEE 802.15.4-based network is that member devices can sleep for extended periods of time and remain part of the network. The SAM R30 module features ultra-low-power sleep modes, with wake from General-Purpose Input/Output (GPIO) or its built-in Real-Time Clock (RTC) while consuming approximately 800 nA. Devices can sleep for years, only waking as needed to transmit data.

The SAM R30 module measures just 12.7 mm x 11 mm and includes the necessary features to drive any remote connected sensor, thereby eliminating the need for a separate MCU in the design. The device also


offers up to 256 KB of flash and 40 KB of RAM, as well as serial data interfaces, USB, and digital and analog I/O for advanced sensor development.

Operating in the sub-GHz RF spectrum, the SAM R30 module delivers two times the connectivity range and better propagation through walls and floors than similarly powered devices using the 2.4 GHz frequency band. This robustness is critical in applications such as leak detection, where the sensor may be buried deep in a remote cabinet, or for pool and spa controllers, which require reliable sub-gigahertz solutions that can communicate through exterior walls.

BAW RESONATOR INNOVATION

For Texas Instruments, (TI), its latest solution aimed at low power wireless communications has to do with a new way to do clock synchronization. Its relevance to this article is that the first implementations of this takes the form of a wireless MCU solution. In February TI announced new bulk acoustic wave (BAW)-based embedded processing and analog chips. The first two devices developed with TI BAW technology, the SimpleLink CC2652RB MCU and the LMK05318 network synchronizer clock are expected to enable new levels of reliability in wireless communication systems.

Communications and industrial automation systems with discrete clocking and quartz-crystal devices can be costly, time-consuming and complicated to develop and are often susceptible to environmental stress. The new devices with TI BAW resonator technology integrate reference clocking resonators to provide the highest frequency in a small form factor. This higher level of integration improves performance and increases resistance to mechanical stresses, such as vibration and shock. As a result of stable data transmission enabled by TI BAW technology, data synchronization of wired and wireless signals is more precise and allows for continuous transmission, which means data can be processed quickly and seamlessly to maximize efficiency (**Figure 6**).

Using TI's new BAW technology, the SimpleLink multi-standard CC2652RB wireless MCU is the industry's first crystal-less wireless MCU on the market. The device integrates a BAW resonator within the quad flat no-lead (QFN) package, eliminating the need for an external high-speed 48-MHz crystal. TI says the CC2652RB is the lowest power multi-standard device supporting Zigbee, Thread, BLE and proprietary 2.4-GHz connectivity solutions on a single chip. Unlike many crystal-based solutions currently on the market, the CC2652RB works in the full -40°C to 85°C temperature range. 

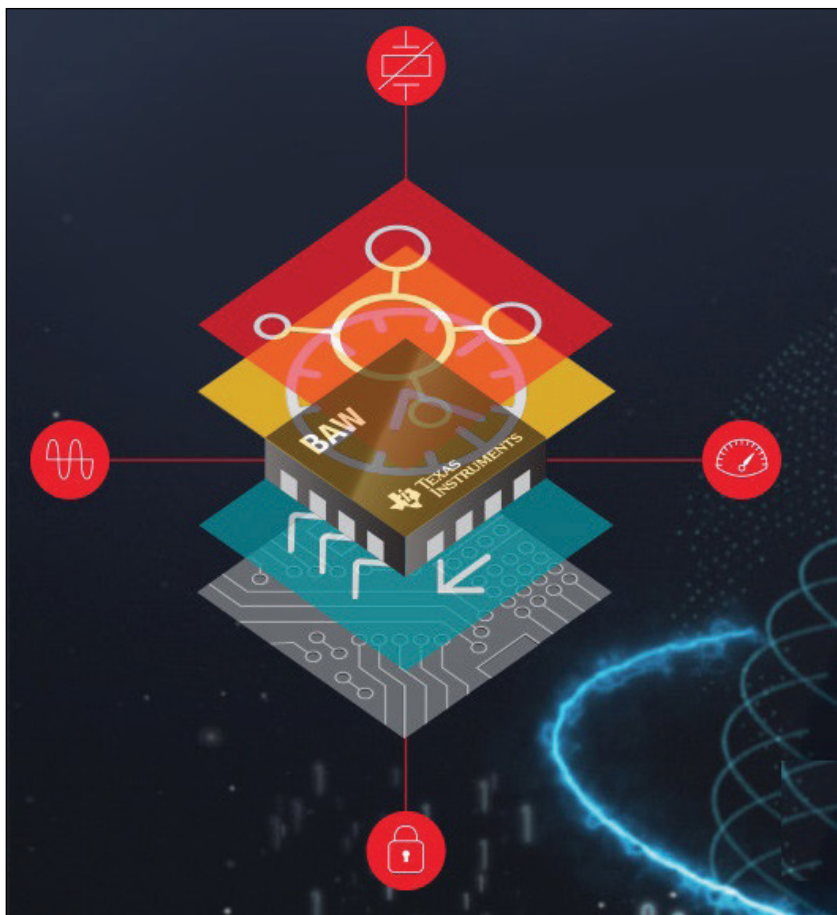


FIGURE 6

Thanks to stable data transmission enabled by TI BAW technology, data synchronization of wired and wireless signals is more precise and allows for continuous transmission, which means data can be processed quickly and seamlessly to maximize efficiency.

RESOURCES

Cypress Semiconductor | www.cypress.com

Microchip Technology | www.microchip.com

Nordic Semiconductor | www.nordicsemi.com

Semtech | www.semtech.com

STMicroelectronics | www.st.com

Texas Instruments | www.ti.com

Mentor[®]

A Siemens Business

Speed up your
PCB design verification

by up to **90%**



7 Habits of Highly Efficient PCB Designers

Design habits that expedite design completion, improve design quality, and enhance productivity are instrumental to highly efficient PCB design.

Learn what you can do to succeed!

Learn More in this FREE White Paper

www.circuitcellar.com/mentor

Product Focus: 32-Bit Microcontrollers

Wireless Workhorse

Suited for wide variety of applications, 32-bit microcontrollers continue to be the workhorse device for today's embedded application. To keep pace, MCU vendors are evolving their product lines with robust security, rich I/O functionality and support for all the popular wireless protocols.

By **Jeff Child**,
Editor-in-Chief

Today's crop of 32-bit MCUs are the fruit of decades of evolution and innovation, taking advantage of Moore's Law to keep increasing chip capabilities. Among the most dynamic trend in MCUs over the past couple years has been the integration of wireless connectivity. Driven mostly by the fast-growing Internet of Things (IoT) phenomenon, today's new generation of MCUs includes many product offerings that include support for Bluetooth Low Energy (BLE), IEEE 802.15.4g, 6LoWPAN and other technologies. Because MCUs are embedded in systems across a wide diversity applications and industries, you can't really generalize about how they are used. Automotive, industrial systems, smart city, smart home, wearable devices and medical gear are among the leading MCU application areas—but there are a whole host of other segments.

An example of a 32-bit MCU application is livestock position and health monitoring (**Figure 1**). According to STMicroelectronics, the meat industry hopes to boost its efficiency and production by maintaining location and

activity data for every animal in the field. Positioning and tracking applications rely on a core MCU operating with MEMS motion sensors to detect animal movement or changes in their position, a dedicated GNSS-based positioning chipset, and a connectivity block for transmitting the acquired data. For its part, ST says it can provide a complete solution for all kinds of animal positioning and tracking devices based on connectivity, geolocation and motion sensors. The company's STM32 MCU based on low-power Cortex-M0 cores, or its ultra-low-power version (M0+), are well-suited to these challenging performance and power consumption requirements. MCU-based BLE provides seamless connectivity with smart phones for parameter analysis, data collection and parameter setting.

All of the leading vendors each offer extensive 32-bit products lines, each with many versions. With that in mind, the MCUs shown in the product gallery on the next couple pages are just a representative sampling today's offerings. [E](#)

FIGURE 1

An example of a 32-bit MCU application is livestock position and health monitoring. The meat industry hopes to boost its efficiency and production by maintaining location and activity data for every animal in the field.



MCU Has Integrated Power Management

The ADuCM4050 from Analog Devices is an ultra-low power integrated MCU system with integrated power management for processing, control and connectivity. The MCU system is based on the Arm Cortex-M4F processor. The MCU also has a collection of digital peripherals, embedded SRAM and flash memory, and an analog subsystem which provides clocking, reset and power management capability in addition to an ADC subsystem.

- Up to 52 MHz Arm Cortex-M4F processor
- 512 KB of embedded flash memory with error correction code (ECC)
- 128 KB system SRAM with parity
- Power management unit (PMU)
- Multilayer bus matrix, central DMA controller
- Crypto hardware supporting AES-128, AES-256 and others
- Serial port, SPI, I²C and UART
- Real-time clock (RTC), general-purpose and watchdog timers
- 12-bit SAR ADC

Analog Devices
www.analog.com



Low-Power MCU Supports Bluetooth Mesh Networking

The CYW20819 from Cypress Semiconductor is a Bluetooth 5.0-compliant, standalone baseband processor with an integrated 2.4 GHz transceiver with support for BLE, BLE 2 Mbps, EDR 2 Mbps and 3 Mbps, synchronous connection-oriented (SCO) and extended SCO (eSCO). The device is intended for use in audio, IoT, sensors and HID markets.

- 32-bit 98-MHz Arm Cortex-M4 CPU
- Complies with Bluetooth 5.0
- Support for BR, EDR 2 Mbps and 3 Mbps, eSCO,
- BLE and LE 2 Mbps
- 256-KB on-chip secure flash; 176-KB on-chip RAM
- AES-128 and TRNG
- Up to 40 GPIOs; I²C, I²S, UART and PCM interfaces
- Two Quad-SPI interfaces
- Auxiliary ADC with up to 28 analog channels

Cypress Semiconductor
www.cypress.com



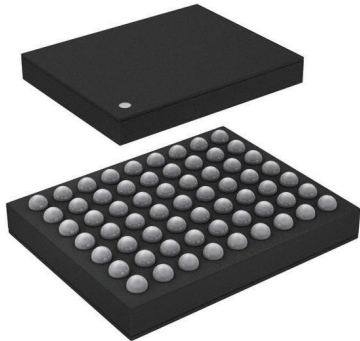
Flash MCU Provides EtherCAT Slave Interface

Infineon Technologies' XMC4800 is a flash-based Arm Cortex-M MCU series that provides an EtherCAT slave interface with a 100 Mb/s data-transfer rate. It has two MII ports for EtherCAT, eight fieldbus memory management units, eight sync manager units and 64-bit distributed clocks. The MCU featured in the Amazon Web Services FreeRTOS kit is an XMC4800-F100x2048.

- 144 MHz Arm Cortex-M4 core
- 2 MB of flash; 353 KB of RAM
- 100-pin package
- On-board debugger
- LED and touch-sense controller
- SD and Multi-Media Card interface
- Four 12-bit ADCs
- Delta-sigma demodulator
- Two-channel, 12-bit DAC

Infineon Technologies
www.infineon.com

32-Bit Microcontrollers

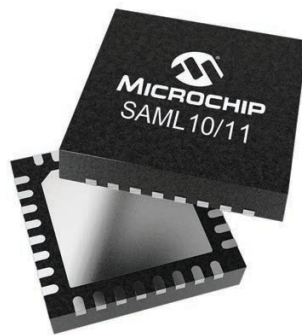


MCU Has Always-On Features for Wearable Designs

The MAX32625/MAX32626 from Maxim Integrated is MCU based on Arm Cortex-M4 with FPU. It's well suited for the emerging category of wearable medical and fitness applications. An internal 96 MHz oscillator provides high-performance capability, and the internal 4 MHz oscillator supports minimal power consumption for applications requiring always-on monitoring.

- Internal oscillator operates up to 96 MHz
- Low power 4 MHz oscillator system clock option for Always-On monitoring applications
- 512 KB flash memory, 160 KB SRAM
- 1.2 V core supply voltage; 1.8 V to 3.3 V I/O
- Wide operating temperature: -30°C to +85°C
- 106 $\mu\text{A}/\text{MHz}$ executing from cache; 49 $\mu\text{A}/\text{MHz}$ executing from flash
- Wake-up to 96 MHz clock or 4 MHz clock
- Peripheral mix provides SPI, I²C, USB and more
- 10-bit delta-sigma ADC

Maxim Integrated
www.maximintegrated.com



MCU Provides Enhanced Security for IoT Systems

The SAM L10 and SAM L11 MCU families from Microchip Technology are based on the Arm Cortex-M23 core, with the SAM L11 featuring Arm TrustZone for Armv8-M, a programmable environment that provides hardware isolation between certified libraries, IP and application code. Security features on the MCUs include tamper resistance, secure boot and secure key storage.

- 32 MHz Arm Cortex-M23 core
- Up to 64 KB flash; 16 KB SRAM
- Enhanced Peripheral Touch Controller
- ARM TrustZone and TRNG (L11 only)
- Crypto accelerators: AES, SHA and GC (L11 only)
- 24- and 32-pin package options
- Under 25 $\mu\text{A}/\text{MHz}$ active; under 100 nA sleep
- Fast wake up times: 1.5 μs
- "SleepWalking" peripherals

Microchip Technology
www.microchip.com

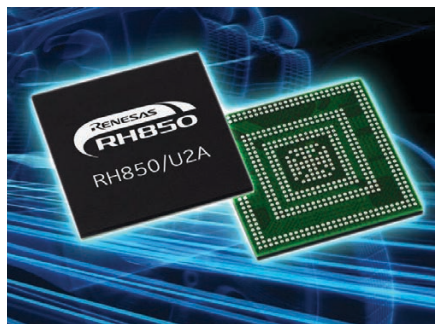


Low-Cost Flash MCU Targets IoT Edge Applications

NXP Semiconductors' LPC5500 MCUs combine single- or dual-core Arm Cortex-M33 cores and Arm TrustZone technology. Built on a low-power 40 nm embedded flash process, the LPC5500 MCU achieves 32 $\mu\text{A}/\text{MHz}$ efficiency at up to 100 MHz core clock frequency. Devices within the LPC55S6x family are starting at a per unit price of \$1.99 for 256K B flash and \$2.49 for 640 KB flash, in 10,000-unit quantities.

- Arm Cortex-M33 core
- Up to 640 KB flash: 320 KB on-chip SRAM
- for advanced edge applications.
- 16-bit SAR ADC) with differential pair mode
- SPI, USB, 8x FlexComm, 2x SDIO
- Dedicated co-processor interface
- Secure boot, SRAM PUF based unique key storage
- Certificate based secure debug authentication
- AES-256 and SHA2-256 acceleration

NXP Semiconductors
www.nxp.com



Automotive Flash MCU Features Virtualization

Renesas Electronics' RH850/U2A16 MCU is a member of Renesas' cross-domain MCUs, a new generation of automotive-control devices, designed to address the growing need to integrate multiple applications into a single chip to realize a unified electronic control units (ECUs) for the evolving electrical-electronic architecture (E/E architecture).

- Four 400 MHz lockstep cores (RH850 G4MH core)
- Hardware-based virtualization
- Self-diagnostic SR-BIST
- Secure OTA updates
- 16 MB of flash ROM; 3.6 MB of SRAM
- Evita Light support up to Evita Full
- SGMIi-standard 1 Gbps Ethernet
- 16 channel CAN-FD interface

Renesas Electronics
www.renatas.com



Dual-Core Wireless MCUs Provide Real-Time Capability

The STM32WBx5 dual-core wireless MCUs from STMicroelectronics come with Bluetooth 5, OpenThread, and ZigBee 3.0 connectivity combined with ultra-low-power performance. They combine the features of ST's STM32L4 Arm Cortex-M4 MCUs and in-house radio managed by a dedicated Cortex-M0+. It is well suited for remote sensors, wearable trackers, building-automation controllers, computer peripherals, drones and other IoT devices.

- 64 MHz Arm Cortex-M4 core
- 32 MHz Arm Cortex-M0+ core (network processor)
- Bluetooth 5 and IEEE 802.15.4 support
- Bluetooth Mesh 1.0 network support
- USB 2.0 FS interface, audio support, LCD driver
- Up to 72 GPIOs, integrated SMPS
- 256-bit AES hardware encryption
- PCROP read/write protection
- JTAG fuse and public-key cryptography

ST Microelectronics
www.st.com



Crystal-less BAW Multiprotocol 2.4-GHz Wireless MCU

Texas Instruments' CC2652RB BAW (bulk acoustic wave) device is a multiprotocol wireless 2.4-GHz MCU targeting Thread, Zigbee, Bluetooth 5 Low Energy, IEEE 802.15.4, IPv6-enabled smart objects (6LoWPAN), Wi-SUN. Very low active RF and MCU current, in addition to sub- μ A sleep current with up to 80 KB of RAM retention, provide excellent battery lifetime and allow operation on small coin-cell batteries in energy-harvesting applications.

- 48-MHz Arm Cortex-M4F CPU
- Dedicated Radio Controller (Arm Cortex-M0)
- 352 KB of flash, 256 KB of ROM
- 80 KB of SRAM with parity
- Supports Over-the-Air Upgrade (OTA)
- Complete RF system and an on-chip DC/DC converter
- Integrated BAW eliminates need for external 48-MHz crystal.
- Sensor Controller CPU with 4 KB of SRAM
- Fast wake-up and ultra-low-power 2-MHz modes

Texas Instruments
www.ti.com

Embedded in Thin Slices

Bluetooth Mesh (Part 2)

Provisioning Pondered

Continuing his article series on Bluetooth mesh, this month Bob looks at the provisioning process required to get a device onto a Bluetooth mesh network. Then he examines two application examples and evaluates the various options for each example.

By
Bob Japenga

COLUMNS

A number of years ago we were working with a company on an OS/2 system. Does anyone remember that operating system by IBM? For a number of years, our company standardized on OS/2 for all of our desktops (**Figure 1**). Eventually Microsoft crushed the competition for the desktop market and made it impossible for us to continue using OS/2. We were working with one of our client's designers on a process control system using OS/2 [1]. After several weeks of interaction with this designer, I was still not sure if his basic design and the architecture of his software was pure spaghetti code or if it was a magnificent and complex architecture that I just couldn't understand. After several months, I think I finally settled on the fact that it was a bit of both. But the elegance of the design escaped me for quite a while.

This reminds me of reading the Bluetooth mesh specification. I am not sure if I am looking at a mishmash of ideas thrown together or an elegant masterpiece that I am missing because of the complexity. Given the team that was involved in creating the specification, I lean towards the elegant masterpiece. But let it be known that, for

me, it's a real challenge to figure out how things are supposed to work from the documentation provided.

I will make an attempt over the next several articles to help all us better understand how Bluetooth mesh works so we can determine if it is applicable to the project on which we are working. I hope to distill down the basic concepts and provide guidelines as to how we can apply this new technology. Hopefully I don't muddy the waters and make things less clear with my own mishmash of ideas.

DEFINITION OF TERMS

Provisioner: A Provisioner on a Bluetooth mesh network is the device that manages the provisioning process. Although multiple Provisioners are allowed on the network, the method of transferring the provisioning data is implementation specific. This means that there will probably be work to switch suppliers of the Bluetooth radio in the Provisioner if you use multiple Provisioners.

Provisioning: Provisioning is the steps performed to add an unprovisioned device to a Bluetooth mesh network. The provisioning process accomplishes two goals: Authenticate the device (confirm that it is a device that is supposed to be on the network) and create the



FIGURE 1

We once worked with one of our client's designers on a process control system using IBM's OS/2 operating system. The design had an elegance that reminds me of the Bluetooth mesh specification.

secure link (keys, algorithms, randomization information) to enable the device to join the mesh network.

Provisioning Data: This is the data that is exchanged during provisioning between the Provisioner and the device that is being added to the network. Among the things it includes are the network key, the address of the device and a number (IV Index) that is used to create a unique identifier for each message.

Symmetric and Asymmetric Key Encryption: These are the two basic types of encryption. Symmetric key encryption uses the same key for both encrypting and decrypting a message. Asymmetric key encryption (also called Public key encryption) uses a public and private key pair that each party generates separately (**Figure 2**) [1]. The public key can be sent to anyone unencrypted (also called “in clear text”). They can then encrypt messages with this public key and only the holder of the private key can decrypt the message. (**Figure 3**). The Bluetooth SIG chose to use the computationally less expensive symmetric encryption during normal networking where both the sender and the receiver have the same key. But during provisioning, which generally happens only once, asymmetric encryption is used. Check out the Bluetooth SIG’s glossary of terms for additional definitions of terms—a link is provided on the *Circuit Cellar* article materials webpage [2].

STATEMENT OF THE PROBLEM

Asymmetric key encryption is the primary method used on our PCs when we connect to a secure site on the Internet. The beauty of this method is that public keys can be transmitted in clear text (unencrypted) to start a secure session. Once I have your public key, I can encrypt my messages and send them to you securely. You use your private key (which is uniquely paired with your public key) to decrypt my message. You do the same thing in reverse with any data that you want to send to me. This beautiful scheme has worked wonderfully since it was first proposed in the 1970s.

The problem for Bluetooth mesh is that, since it chose to use symmetric key encryption where both parties have the same private key, how do all parties securely get that private key? The Bluetooth mesh team created a method of distributing a unique private key that is executed before any device joins the network. This is what we are going to describe here.

At the highest level, an unprovisioned device periodically sends a beacon in clear text with its unique identifier and its capabilities as a device. The Provisioner then opens a link for that device and the device

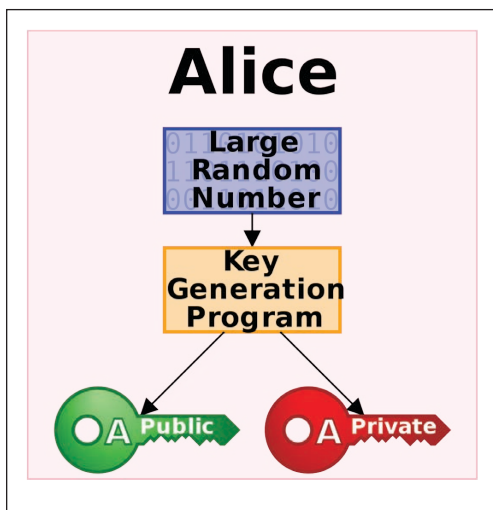


FIGURE 2 Asymmetric key encryption (also called Public key encryption) uses a public and private key pair, which each party generates separately.

acknowledges the Provisioner. Provisioning takes place when the Provisioner sends the provisioning data to the device. This data includes the private key, which is called the Network key. There is more than one type of key but we will only cover the primary private key in this article. Provisioning also authenticates that the unprovisioned device is that it says it is. Authentication, although part of provisioning, is another significant topic that is too large to cover in this article. We will look at authentication in a future article in this series. Once the provisioning data is successfully exchanged, the link is closed and the device is now part of the Bluetooth mesh network.

DRILLING DEEPER

The provisioning process consists of five phases. Phase 1 is called Beaconsing,

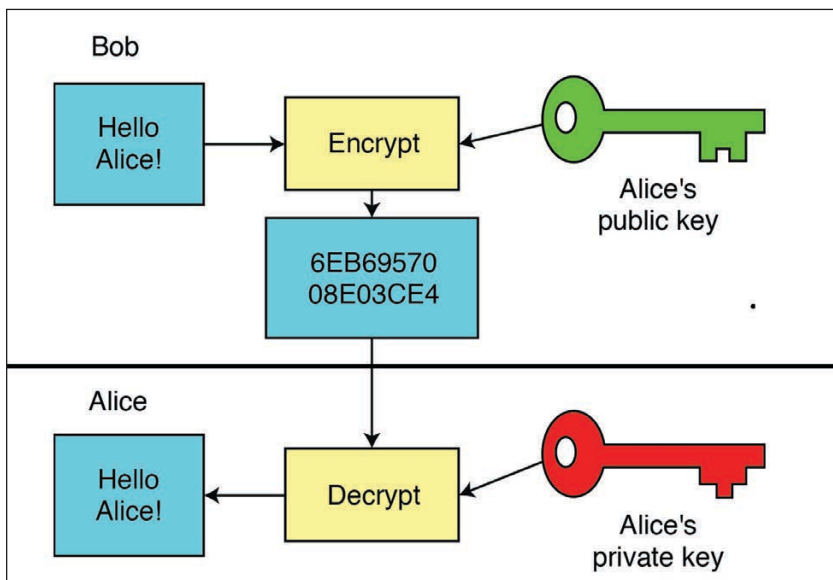


FIGURE 3 The public key can be sent to anyone unencrypted (also called “in clear text”). They can then encrypt messages with this public key and only the holder of the private key can decrypt the message.

COLUMNS

FIGURE 4

Insecure provisioning

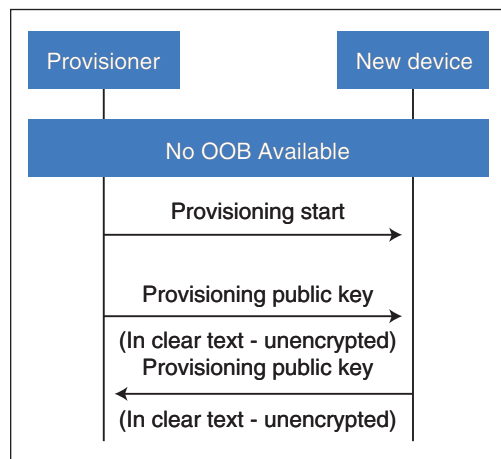
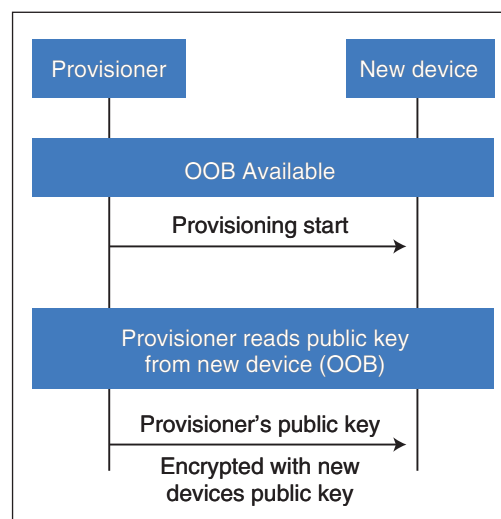


FIGURE 5

Secure provisioning



whereby the device lets the Provisioner know it is desiring to be provisioned. Phase 2 is called Invitation whereby the Provisioner invites the device to join the network. The device sends its capabilities including what encryption algorithms it supports, the type of public keys it supports and the type of Out-of-Band (OOB) capabilities it supports. Phase 3 is where the Provisioner and the device exchange their public keys. Phase 4 is where the Provisioner authenticates the device. Finally, in Phase 5 the Provisioner distributes the provisioning data.

In the case of a browser and a secure Internet site, the secure Internet site wants anyone to be able to connect securely to the site. However, not everyone has access to the site. Typically access to the site is limited by usernames and passwords.

This is not true for most mesh networks. A mesh network doesn't want to allow every

device in range to exchange the public keys in order to have a secure network connection. This is especially true with a mesh network where a lot of the work of the device is passing data from other devices to devices downstream. You wouldn't want a rogue device to be able to do this. So, for Bluetooth mesh, it would not be wise to pass the public keys in clear text (unencrypted).

With that in mind, the simple step (for PCs) of exchanging public keys is not so simple for Bluetooth mesh networks. For Bluetooth mesh networks, the passing of the public keys must be done securely to have a secure network. The Bluetooth specification allows two ways for the public keys to be exchanged. You can securely exchange the public keys through an OOB method or insecurely (**Figure 4** and **Figure 5**).

There are a number of OOB methods allowed by the specification. All of them involve user interaction. One is that the public keys are exchanged over a Near Field Communications (NFC) link. Since a lot of the Bluetooth chips include NFC capability, technically this is a viable solution with no extra recurring cost to the device. Other methods include some form of user input. Again, the specification allows a variety of input and output methods: audible, alphanumeric, visual, vibration or discrete button pushes. So, if your application can allow that, you can provide the public keys via some fairly simple I/O mechanizations—albeit requiring complicated user interaction.

REAL WORLD EXAMPLES

The Bluetooth mesh specification's statement that only the OOB method of exchanging public keys is secure is disconcerting for anyone concerned about security and simplicity of installation. We know that all of these OOB methods add significant complications for the user during provisioning. Next time we will drill into the details of the "insecure" method of passing the public keys that are used. I say that because, as I will attempt to demonstrate next time, the method is not as insecure as it might seem. For now, let's assume that you are designing a system for a customer using Bluetooth mesh. You have a customer who wants the network to be secure. The specification says that only the OOB method of passing the public keys is secure. Let's look at some real-world implementations and what options you have to create secure provisioning.

Home Automation: Imagine that you are designing a line of devices for home automation. You want the network to be secure. You want the installation to be simple. The simplest way would be to install the devices and turn them on. All devices would

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials

References [1] through [3] as marked in the article can be found there.

automatically be provisioned securely. How can this be achieved? One option to achieve this would be to completely bundle everything pre-provisioned at the factory. This removes the complexity of user involvement in the passing of the public keys, keeping the installation simple for the homeowner. The homeowner could just install the devices, turn them on and they would work. The problem is that you would be selling a system, not home automation devices. This may be acceptable in some designs but not in others.

Alternatively, the factory could pre-install all devices with the same shared public keys. No public key exchange would take place during provisioning. This would enable you to sell devices rather than systems and be able to replace faulty devices. This is similar to the default trust center link key approach of Zigbee and has some of the same pitfalls.

The NFC method is more flexible and quite simple for the user. Each device would need to be brought in close proximity (within inches) to the Provisioner as part of the installation procedure. If the device has a user interface, alphanumeric input is also a possibility although sometimes a cumbersome, error prone and time consuming process. The blinking LED, the vibrating box, the push button or the audible beep are even more complicated for the user but are open to you as designers.

Industrial Control: Imagine that you were designing some kind of lighting control for large buildings. There might be hundreds of your devices installed throughout the building. Your customers want it to be secure and they are going to want it to be simple to install and maintain. How would OOB public key exchange work in this example? Pre-provisioning at the factory is a possibility. But, as with home automation, it means that you are not selling devices but a system with a predetermined number of devices.

Requiring the installer to bring each device to the Provisioner or to use the user input or output on each device is the other option. One of our customers deemed it an unacceptable burden on the installer to use NFC OOB and require the installer to take the fixture and bring it into close proximity with the Provisioner. Even though the device had an LED and a push button on it making visual or discrete input OOB a possibility, the user input method would have been even more complicated. Requiring the installer to push the button X number of times or reading the pulse pattern on the LED was completely rejected by our customer.


We had to take a hard look at how insecure the Bluetooth provisioning process was without OOB in our environment. The method may not be as insecure as it seems. More on that next time.

ABOUT THE AUTHOR

Bob Japenga has been designing embedded systems since 1973. In 1988, along with his best friend, he started MicroTools, which specializes in creating a variety of real-time embedded systems. MicroTools has a combined embedded systems experience base of more than 200 years. They love to tackle impossible problems together. Bob has been awarded 11 patents in many areas of embedded systems and motion control. You can reach him at rjapenga@microtoolsinc.com.



CONCLUSION

One only needs to read the number of vulnerabilities that are published daily on the CERT Vulnerability website [3] to know that security is very hard to achieve. The provisioning process is key (pun intended) to creating a secure Bluetooth mesh network. Next time we'll describe the "insecure" method of passing the public keys and evaluate when this method is an acceptable risk. But of course, only in thin slices. 


Verilog HDL

With the right tools

designing a microprocessor can be easy.

Okay, maybe not easy, but certainly less complicated. Monte Dalrymple has taken his years of experience designing embedded architecture and microprocessors and compiled his knowledge into one comprehensive guide to processor design in the real world.

Monte demonstrates how Verilog hardware description language (HDL) enables you to depict, simulate, and synthesize an electronic design so you can reduce your workload and increase productivity.



cc-webshop.com

The Consummate Engineer

Improving Software Quality

By the Book

By
George Novacek

There's no doubt that achieving high software quality is a human-driven endeavor. No amount of automated code development can substitute for best practices. A great tool for such efforts is the IEEE Computer Society's Guide to the Software Engineering Body of Knowledge. In this article, George discusses some highlights of this resource, and why he has frequently consulted this document when preparing development plans.

It has been demonstrated over and over again that software quality depends 100% on human activity. Nothing else. There is no wear out, aging or some other mechanism affecting software quality

and performance. While many errors can be avoided by automating repetitive tasks, ultimately it is the lone, hard working software engineer who bears responsibility for the quality of the result.

As systems grow ever more complex, the task of competent software engineering practice is getting harder and harder. Experts agree that the only method for bringing software quality under control lies in following strict design guidelines. There's a vast body of knowledge of different topics a competent software engineer needs to grasp. In recognition of that the Institute of Electrical and Electronics Engineers' (IEEE) Computer Society issued the *Guide to the Software Engineering Body of Knowledge*. The common abbreviation of the title is SWEBOK, now in version 3 (**Figure 1**).

The SWEBOK is an exhaustive compilation of software engineering knowledge but it is not a cookbook. It identifies the multitude of topics software engineers are likely to encounter in the course of software development and points them in the right direction to find solutions. The document begins with top level characterization of the fundamental issues, then continues drilling

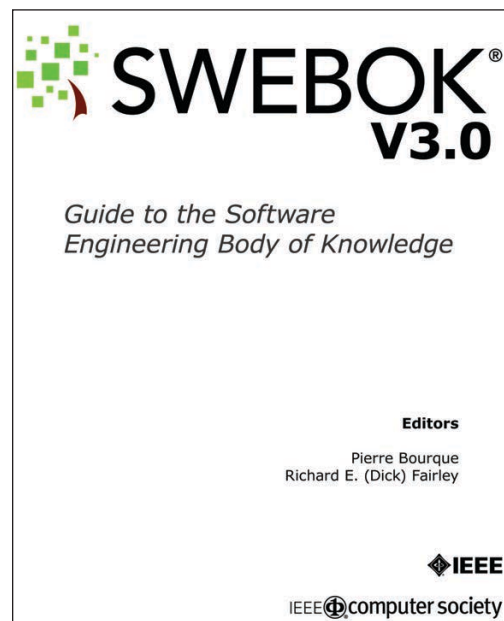


FIGURE 1

Title page of the Software Engineering Body of Knowledge (SWEBOK)

	Subject	Subchapters
1	Software Requirements	Software Requirements Fundamentals Requirements Process Requirements Elicitation Requirements Analysis Requirements Specifications Requirements Validation Practical Considerations Software Requirements Tools
2	Software Design	Software Design Fundamentals Design Key Issues SW Structure & Architecture User Interface Design Design Quality Analysis & Evaluation SW Design Notations Practical Considerations Software Requirements Tools
3	Software Construction	Software Construction Managing Construction Practical Consideration Construction Technologies SW Construction Tools
4	Software Testing	Software Testing Fundamentals Test Levels Test Techniques Test-Related Measures Test Process SW Testing Tools
5	Software Maintenance	Software Maintenance Fundamentals Key Issues in SW Maintenance Maintenance Process Requirements Analysis Techniques for Maintenance Software Maintenance Tools
6	Software Configuration Management	Management of Configuration Process Configuration Identification Configuration Control SW Configuration Status Accounting Configuration Auditing SW Release Management & Delivery Software Configuration Management Requirements Tools
7	Software Engineering Management	Initiation & Scope Definition Project Planning Project Enactment Review & Evaluation Closure SW Engineering Measurement Software Engineering Management Tools
8	Software Engineering Process	Software Process Definition Software Life Cycles Process Assessment & Improvement Software Measurement Software Engineering Process Tools

TABLE 1

Listing of subjects and their subchapters in SWEBOK

	Subject	Subchapters
9	Software Engineering Models and Methods	Modeling Types of Models Analysis of Models Software Engineering Methods
10	Software Quality	Software Quality Fundamentals Software Quality Management Process Practical Considerations Software Quality Tools
11	Software Engineering Professional Practice	Professionalism Group Dynamics & Psychology Communication Skills
12	Software Engineering Economics	SW Engineering Economics Fundamentals Life Cycle Economics Risk & Uncertainty Economic Analysis Methods Practical Considerations
13	Computing Foundations	Problem Solving Techniques Programming Language Basics Algorithms & Complexity Operating System Basics Network Communications Basics Basic Developer Human Factors Abstraction Debugging Tools & Techniques Basic Concept of a System Compiler Basics Parallel & Distributed Computing Secure Software Development and Maintenance Programming Fundamentals Data Structure & Representation Computer Organization Database Basics & Data Management Basic User Human Factors
14	Mathematical Foundations	Sets, Relation & Functions Proof Techniques Graphs & Trees Finite States Machines Numerical Precision, Accuracy & Errors Algebraic Structures Basic Logic Basics of Counting Discrete Probability Grammars Number Theory
15	Engineering Foundations	Empirical Methods & Experimental Techniques Statistical Analysis Measurement Engineering Design Modeling, Simulation and Prototyping Standards Root Cause Analysis



ABOUT THE AUTHOR

George Novacek was a retired president of an aerospace company. He was a professional engineer with degrees in Automation and Cybernetics. George's dissertation project was a design of a portable ECG (electrocardiograph) with wireless interface. George has contributed articles to *Circuit Cellar* since 1999, penning over 120 articles over the years.

down to more and more specific matters. Whether a professional software designer or a hobbyist, your familiarity with this document (available for free [1]) can only benefit you and your organization. The SWEBOK's 355 pages limit me to just a top-level overview, but I cannot recommend any stronger that you download and familiarize yourself with it.

A RICH RESOURCE

The SWEBOK is a comprehensive overview of the entire software development process. It comprises fifteen chapters, each addressing a major topic. Every chapter is subdivided into subchapters, sections, paragraphs and so forth to identify potential issues and where to find their solutions. **Table 1** shows the SWEBOK table of contents—limited to the subchapter level in this case.

I have frequently consulted this document when preparing development plans and check lists to make sure that all the pertinent issues have been addressed for the project implementation. This is not to suggest that everything listed in the SWEBOK must be a part of every design or a responsibility of every design engineer. A design of a simple embedded controller may not require the engineer's familiarity with some sophisticated mathematical processes. Not all engineers need to get involved in management issues, group dynamics or economic considerations. Each project is unique and different.

No SWEBOK topic can be taken in isolation. All the relevant issues, when applicable for the particular project, are related to each other. Therefore, you must make sure the requirements listed in each chapter

are consistent across the entire project. This comes under consideration especially during project planning. Project managers must select issues relevant to the project development and prepare a suite of plans to which members of the development team will be held to conform.

One notable feature of SWEBOK—in contrast to the numerous software development documents I've studied—is that it addresses software security in terms of its development as well as the operation.

Our industry does not always use very intuitive names, so it occasionally takes a little effort to figure out what a specific title means. Take Chapter 1, for example. There are eight topics identified, all related to the requirements. The eight sub-topics the SWEBOK authors want us to address are the requirements fundamentals, the requirements process, the requirements elicitation, analysis, the requirements specifications, requirements validation, practical considerations and tools selection. The topic meanings may be a little perplexing, but the following subchapters explain it all. While the entire Chapter 1 is focused on the software specification globally, the subchapters define the detail requirements, providing the necessary explanations and helping us not to omit anything important. Let's consider just a few selected items to get the idea of what it's all about.

SOME SELECTED TOPICS

It is acknowledged that software projects are critically vulnerable when the requirements-related activities are performed poorly. It has been established that with modern software development tools the majority of "bugs" can be ascribed to a deficient specification. With that in mind, a high-quality Software Requirements document fully defines the needs and constraints placed on a software product to solve a real-world problem. Breaking down the high-level requirements into specific activities as described in the subsections ensures nothing falls through the cracks.

Subsection 1 of the SWEBOK lists Software Requirements Fundamentals. What exactly are they? To understand, they are further broken down into six paragraphs:

1. Definition of a Software Requirement
2. Product and Process Requirement
3. Functional and Non-functional Requirements
4. Emergent Properties
5. Quantifiable Requirements
6. System Requirements and Software Requirements

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials

Reference [1] as marked in the article can be found there.


Paragraph 5 is called Quantifiable Requirements. It means that every software requirement must be stated explicitly, quantitatively, as clearly as possible, with no ambiguities left to interpretation. Vague and unverifiable requirements or those opened to interpretation or subjective judgment, such as “the software shall be reliable” are unacceptable. Instead, you must explicitly state that, for instance, that the probability of the software containing an error shall be less than 1×10^{-9} . (How we establish that is another topic.) Or, instead of calling for a “short propagation delay”, the response time needs to be stated in seconds and verifiable by test.

Consider also Chapter 5, Software Maintenance. It has five subchapters. Subsection 2 addresses Key Issues in Software Maintenance and contains four paragraphs:

1. Technical Issues
2. Management Issues
3. Maintenance Cost Estimation
4. Software Maintenance Measurement

Paragraph 3 is a nasty one. To a large degree its successful implementation relies

on your experience. There are projects where software maintenance is not required and therefore its cost per Paragraph 3 is no issue. However, the absence of such a requirement must be confirmed, so there is no ambiguity about who pays if a fix is needed. Paragraph 4 helps to identify, analyze and develop measures to be eventually applied to the correction process should future maintenance be a part of the contract. The cost of the effort and the resources to be expended to diagnose the deficiencies or the cause of a failure can only be estimated—same for the cost of potential requalification. The measure of stability is a crucial economic consideration. It encompasses the probability of encountering unexpected behavior as well as the corrective action’s testability, verification and validation. These issues, while very difficult to predict, are sometimes underestimated or ignored during project planning. Unfortunately, they seem to have the tendency to come back and bite us.

It has not been my intention to review the SWEBOK in detail. My goal was to draw your attention to this excellent document, because it can help you and your organization to improve the quality of your software. 

COLUMNS



Circuit Cellar 2018 Archive



2018

Issues # 330-341 — CD #23
Copyright 2018, KCK Media Corp.

Order yours today

cc-webshop.com

The Darker Side

The Art of Current Probing

Mindful Measuring



In his February column, Robert talked about oscilloscope probes—or more specifically, voltage measurement probes. He explained how selecting the correct probe for a given measurement and using it properly are as important as having a good scope. Here, Robert continues the discussion with another common measurement task: accurately measuring current using an oscilloscope.

By
Robert Lacoste

FIGURE 1

The EasyPIC V7 development board, used as an example for this article, draws about 118 mA from its 5 V source.

Welcome back to The Darker Side. As usual, let's start with an example. When preparing this article, I wandered around my lab and looked for some electronic stuff that could be used to illustrate what I wanted to show you. The first thing I grabbed was a nice development board from MikroElektronika, the EasyPIC V7 (**Figure 1**). For the purpose of this article, you just have to know that it's based on an 8-bit microcontroller, namely a PIC18F45K22 from Microchip Technology. The board has a zillion push-buttons, dip-switches, LEDs, 7-segment displays, LCD connectors, extension headers and so forth. It comes with demo firmware, which blinks some LEDs and increments a counter on the four 7-segment LED displays. It's powered by an external DC source ranging from 9 V to 23 V. An on-board DC-DC converter provides a stable +5 V to all circuits.

Ok. Now assume you want to measure the current consumption of this board on the +5 V power rail. You could start by simply using a multimeter. Fortunately, a jumper (J6) is present between the output of the DC-DC converter and the +5 V net. I removed this jumper and connected my Keysight U1253B multimeter between its two pins, configured in ammeter mode. The current flowed through the meter and allowed me to get a measurement. As illustrated in Figure 1, the meter reading was around 118 mA.

Job done? Not exactly. When I did this test, the meter reading was fluctuating a lot. Why? Simply because the current consumption of the board was not constant. In this example, the main cause is probably the 7-segment LED displays. The segments are constantly switched on and off by the firmware when counting, so the current is not stable over time. A similar situation appears in a lot of

projects. For example, a wireless device would draw a different current when transmitting or receiving. If you can measure not only an average value but also an actual plot of the current over time, you have a far better understanding of the design—currents for each mode, transmit and receive durations and so on.

OHM'S LAW?

Measuring current over time requires more than a basic multimeter, and the oscilloscope will be your friend. An oscilloscope measures voltages, not currents—so a kind of current-to-voltage converter will be needed. What are the possible solutions? The most straightforward is to remember Ohm's law: a current (I) flowing through a resistor (R) will generate a voltage (V) across the terminals of the resistor, simply calculated as $V = R \times I$. So, a simple shunt resistor can transform a current into a voltage, and this voltage can then be measured by an oscilloscope.

Going back to my example, a resistor could be inserted in place of the ammeter, meaning on the +5 V rail. One end of the resistor will then be at a voltage of +5 V, and the other will be at a slightly lower voltage according to Ohm's law, specifically at $5\text{ V} - (R \times I)$. The difference between these two voltages gives $R \times I$, so the current can be measured, because you know the value of R .

How to select the value of this resistor? It should be as large as possible in order to get a measurable voltage, but not too high, because you don't want to reduce the circuit operating voltage too much. Here the power voltage is 5 V, and the overall current is about 118 mA. Assuming that a 1% voltage drop caused by the measuring resistor is acceptable, this means that I need a voltage drop of about $5\text{ V} \times 1\% = 50\text{ mV}$. So, I should use a value of $R = V / I = 0.05\text{ V} / 0.118\text{ A} = 0.42\ \Omega$. I had 1.5 Ω resistors on hand, so I soldered three of them in parallel to get 0.5 Ω , and connected them on the J6 jumper header.

Now how do you actually measure the voltage drop across this resistor with an oscilloscope? If the current measurement resistor were on the 0 V line, or if the circuit under test were fully floating compared to the oscilloscope, then this would be easy. Just take a standard voltage probe, connect its ground clip to one end of the resistor and its tip to the other side, and you would measure $R \times I$. However, such a setup is impossible if the ground of the scope is connected to the ground of the board under test—either directly or indirectly. Unfortunately, this is often the case, for example, if you need to measure other signals on the board simultaneously with the same oscilloscope.

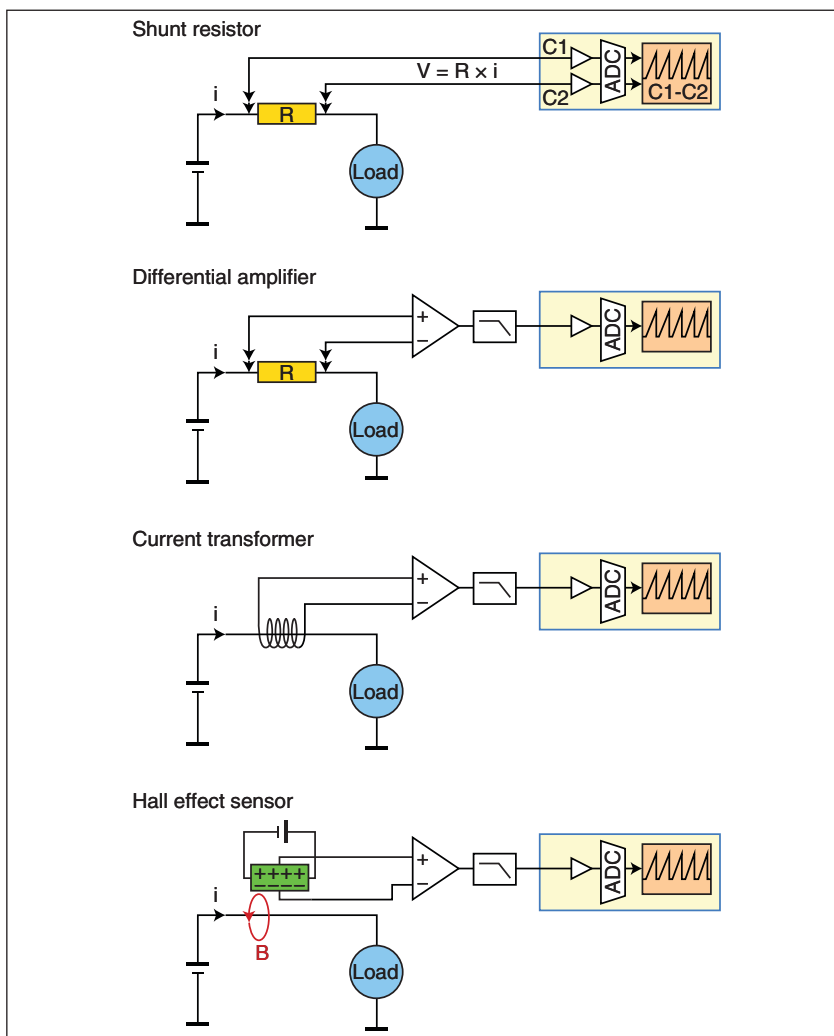


FIGURE 2

Shown here are four different ways to measure the voltage drop across a resistor with an oscilloscope.

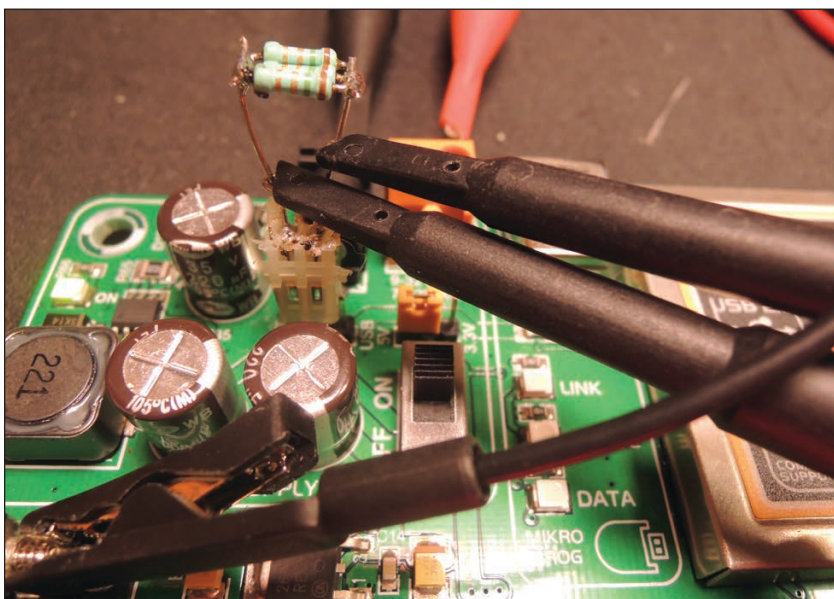
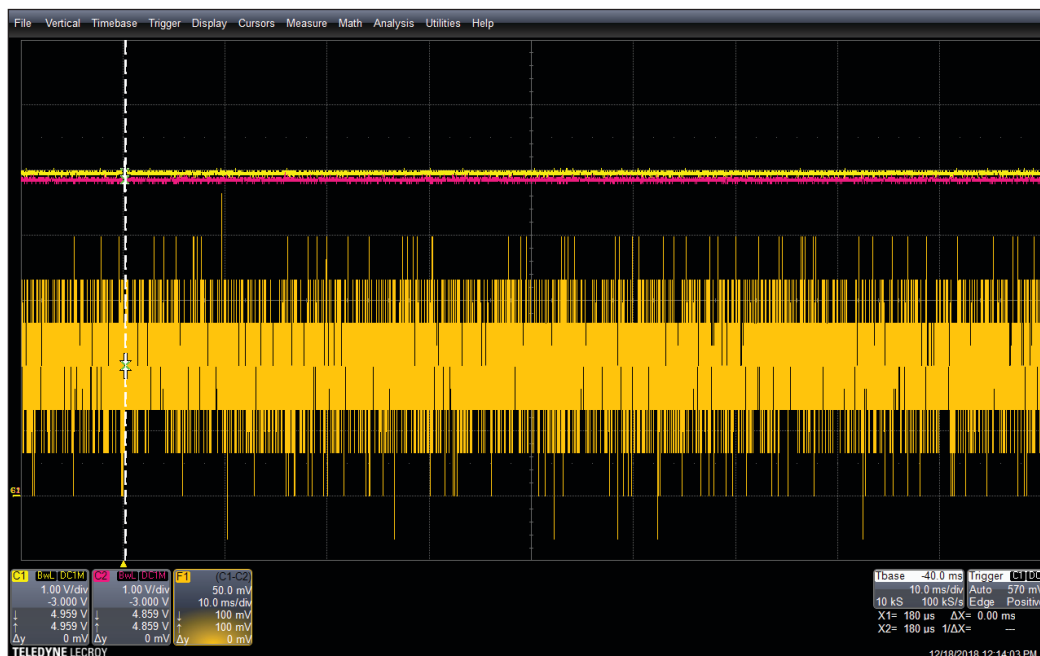


FIGURE 3

Two standard passive voltage probes were connected to a 0.5 Ω shunt resistor, giving a 50-mV voltage difference for 100 mA of current.

FIGURE 4

Both probes measure voltages close to 5 V (yellow and red curves). The plot of their calculated difference is very noisy due to the limited resolution of the oscilloscope's ADCs.



As illustrated in **Figure 2**, in such a case the easiest solution is to use two voltage probes and a dual-input oscilloscope. Each probe is connected between the system ground and one end of the current-measuring resistor, and the scope is configured to display the voltage difference between the two input channels. I tested this setup for you, as shown in **Figure 3**. The resulting plot is reproduced

in **Figure 4**, and is not pretty at all—noise!

What happened? The main cause of this bad result is that I used a digital scope. All digital scopes have fast analog-to-digital converters (ADCs) but with a low resolution. 8-bit is typical, even on high-end scopes such as the Teledyne LeCroy WaveRunner 610Zi that I used for this article. The scope digitizes each channel based on the full scale selected through the “volt-per-division” setting. Here, because the measured voltages were +5 V, the ADC full range was larger than 5 V and in fact close to 10 V, so the ADC resolution for each channel was $10\text{ V} / 256 = 40\text{ mV}$. The scope digitized each channel with a 40 mV resolution, then calculated digitally the difference between the two channels, giving a result still with a 40-mV resolution. These 40-mV quantization stairs are clearly visible in **Figure 4**. The problem is that measuring small variations of a 50-mV difference voltage with a 40-mV resolution is impossible!

DIFFERENTIAL MEASUREMENT AMPLIFIER

How do you greatly improve the measurement? By calculating the difference between the two measured voltages with an analog difference amplifier before digitization. The difference signal can then be amplified as desired, and even low-pass filtered to remove any high-frequency noise. This is how “A-B” mode used to work on analog scopes, and this exactly the job of a differential measurement amplifier—also called differential probe. These instruments are close cousins to the high-frequency voltage differential probes presented in my previous article, but are



ABOUT THE AUTHOR

Robert Lacoste lives in France, between Paris and Versailles. He has 30 years of experience in RF systems, analog designs, and high speed electronics. Robert has won prizes in more than 15 international design contests. In 2003 he started a consulting company, ALCIOM, to share his passion for innovative mixed-signal designs. Robert's bimonthly Darker Side column has been published in *Circuit Cellar* since 2007. You can reach him at rlacoste@alciom.com.

For detailed article references and additional resources go to:
www.circuitcellar.com/article-materials

RESOURCES

Aim-TTi | www.aimtti.com
Keysight | www.keysight.com
Microchip Technology | www.microchip.com
MikroElektronika | www.mikroe.com
Tektronix | www.tektronix.com
Teledyne Lecroy | www.teledynelecroy.com

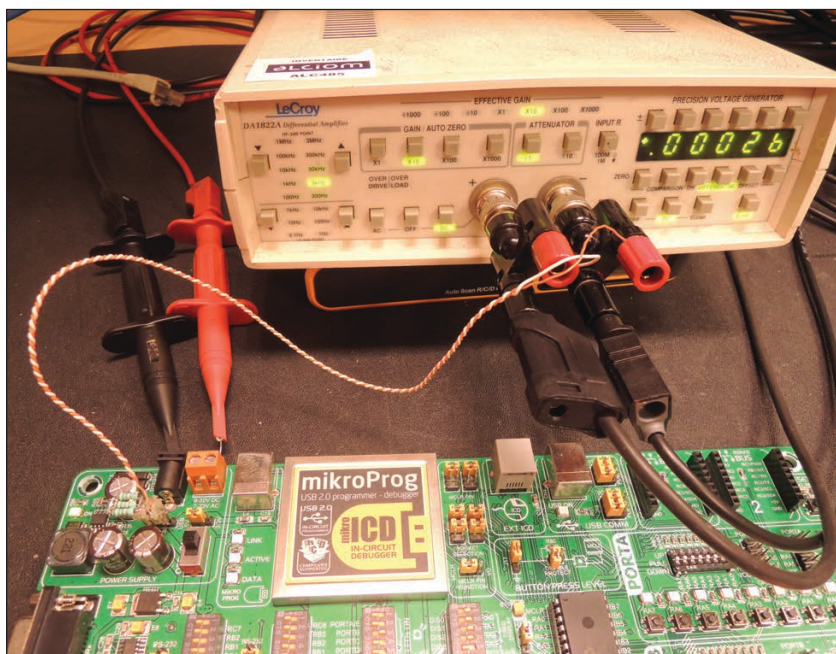
dedicated to very low differential voltages with quite low frequency bandwidths.

Of course, you could build a differential measurement amplifier by yourself, using op amps or maybe instrumentation amplifiers. If you do it, don't forget to publish your work in *Circuit Cellar*. In my case, because I'm quite lazy, I simply took a Teledyne LeCroy DA1822A differential amplifier from my lab, and connected it between the 0.5 Ω current measurement resistor and the scope input (Figure 5). Such a probe has a gain selectable from 1/1,000 to 1,000, and a low-pass filter selectable from 100 Hz to 3 MHz. Here I set the gain to 10—because it was enough to get an output voltage of 500 mV from the 50 mV across the resistor—and a bandwidth of 3 kHz, which was adequate for the measured waveforms.

Is there any improvement in the measurement when using such an amplifier? You bet there is (Figure 6). On this plot, the vertical scale is 100 mV per division, corresponding to 100 mV / (10 \times 0.5 Ω) = 20 mA per division. Now you can clearly see why the current measurement was varying. It was oscillating between 100 mA and 120 mA, with a quite complex rectangular-shaped waveform and a period of about 4.3 ms. I haven't checked the source code of the demo firmware running on this development board, but I'd bet that the LED multiplexing period is 4.3 ms.

NON-RESISTIVE CURRENT PROBES

Shunt-resistors are the most straightforward current-to-voltage converters, but they have a drawback: you need to cut the power line and add a serial resistor somewhere. Cutting the line is not always easy, and the



resistor will surely change the behavior of the circuit under test. Moreover, the resistor needs to make direct electrical contact with the circuit under test and the scope, and this could be a source of noise or even a safety issue for high-voltage or high-power systems.

Now it's time to present two other families of current-measurement probes, which don't need a resistor to be inserted in the circuit under test, and do not even make any contact with the circuit. The first is a so-called "current transformer" (Figure 2). This is basically several wire turns around a magnetic core that surrounds the wire carrying the current to be measured. Such a setup acts as a transformer, and the current could be measured through

FIGURE 5

A differential amplifier was inserted between the shunt resistor and the scope—here a Teledyne LeCroy DA1822A. The resistor was connected through a twisted pair wire to limit noise.

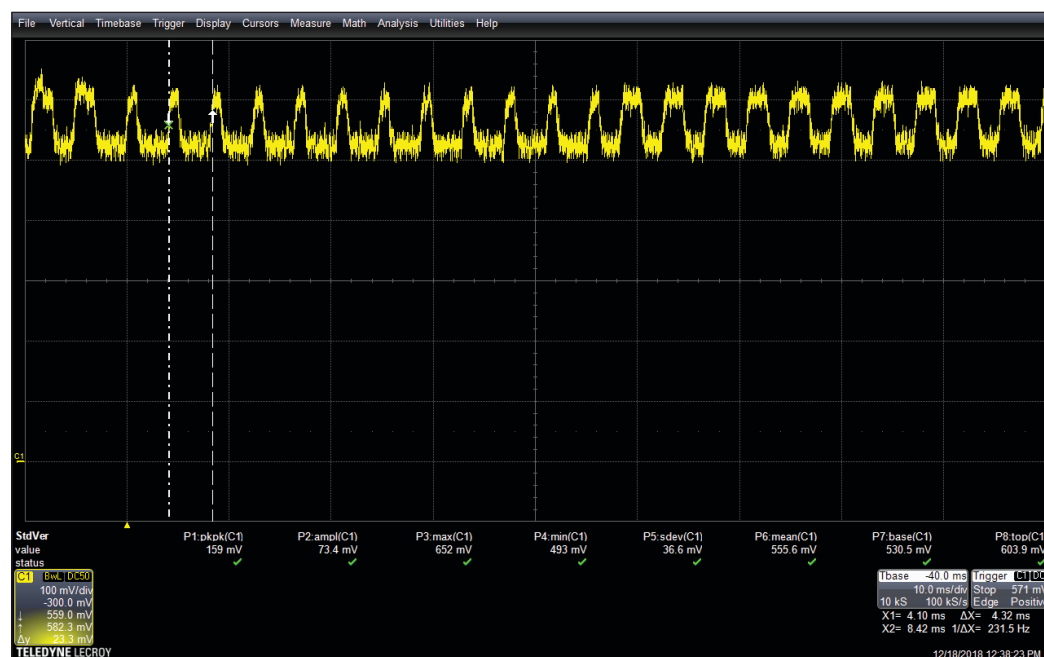


FIGURE 6

With a differential amplifier the current measurement shown in this plot is far cleaner.

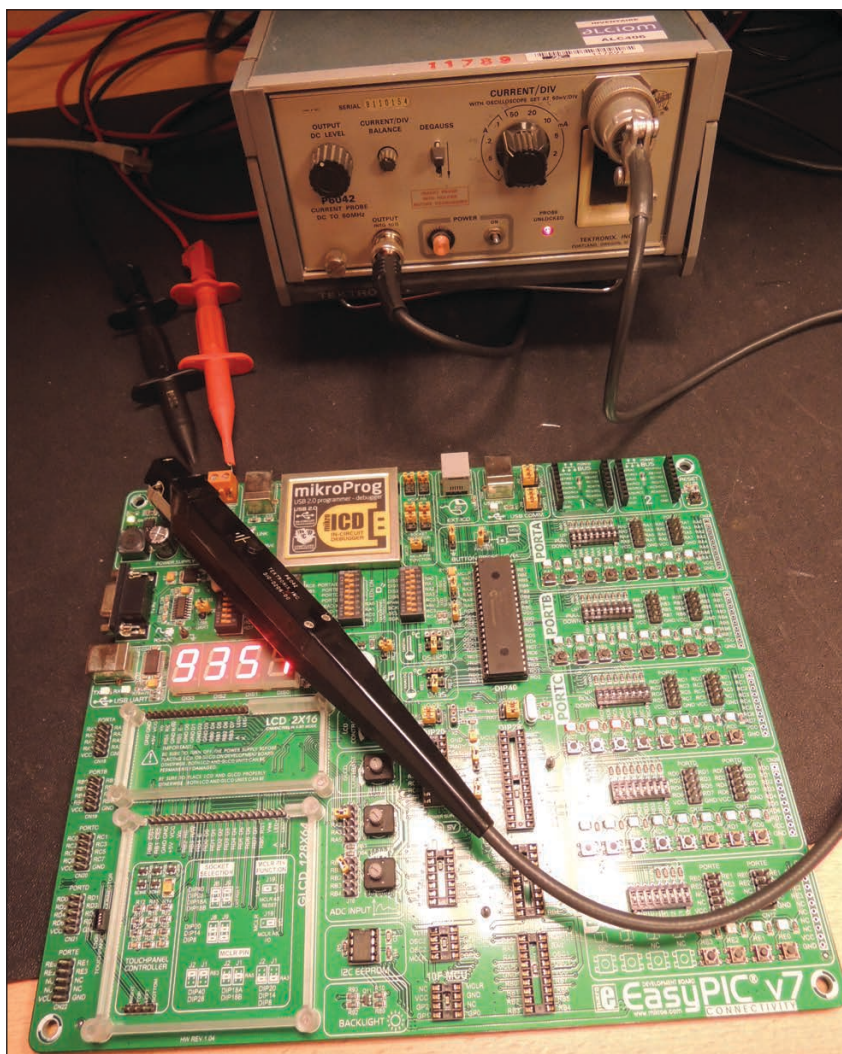


FIGURE 7
My old but trusty Tektronix P6042A current probe

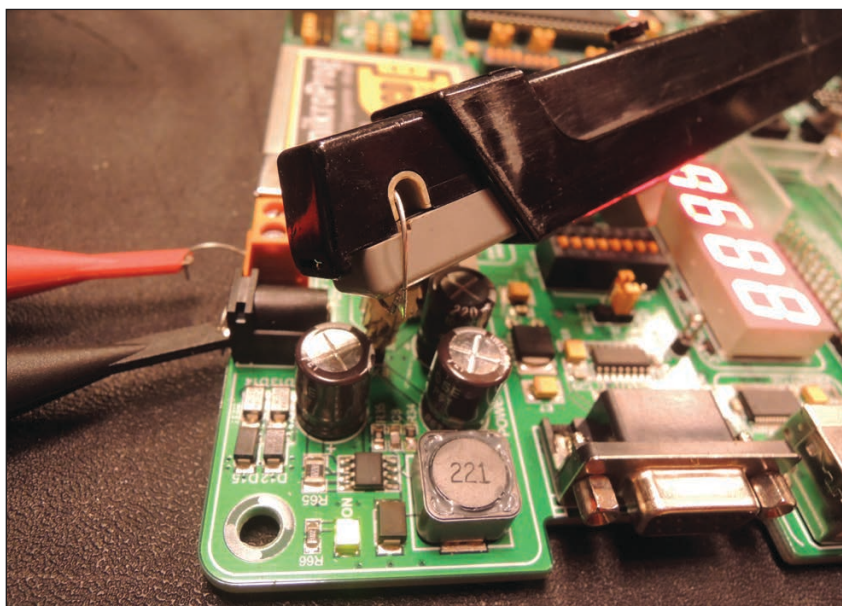


FIGURE 8
The tip of the Tektronix P6042A probe includes both a Hall-effect sensor and a current transformer.

the test wire. As with any transformer, the measurement sensitivity could be improved by using ferrite cores. This kind of current-measurement probe is often in the form of a current clamp, and doesn't need to open the circuit under test or to make any direct electrical contact with it. The disadvantage of this type of current transformer probe? It doesn't allow measurement of DC currents, as transformers only work with AC signals. In contrast, the second type of current probes—those based on the Hall effect—can measure both AC and DC currents.

What is the Hall effect about? See Figure 2 for a high-level explanation. The current circulating in the conductor under test generates a magnetic field (B) around it. A so-called "Hall sensor"—positioned close to the wire and in the same plane—is a simple conducting rectangle. A polarization current source is applied between two of its opposed sides. Due to the presence of the magnetic field (B), a voltage then appears between the two other sides, and is proportional to B . That's the Hall effect. Hall sensors have some drawbacks, mainly the need for a power source and a limited bandwidth. On the other hand, they're inexpensive and widely available.

As always, current-transformer probes or Hall-effect probes can be home-built, but it's also possible to buy one. Some are only passive probes, and others come with a companion amplifier unit. An example of an old but trusty Tektronix P6042 current probe is shown in **Figure 7**. This model was introduced in 1967, can be found on eBay for reasonable prices and is still a great device. This probe has a bandwidth from DC up to 50 MHz, and current-measurement capabilities from less than 1 mA up to 10 A. Its internal circuits are really interesting to study. That's because this probe is, in fact, a very smart combination of a Hall-effect sensor (for DC measurement) and current transformer (for high bandwidth), packaged in a clamp-on hand-held probe (**Figure 8**). Both sensors are connected in series, and are precisely matched for a good wideband response. A good article from Paul Rako details how it works—see the *Circuit Cellar* article materials webpage for link.

I connected this probe to my scope and tried to use it to measure the current consumption of the EasyPIC development board. The measurement was initially disappointing, with a very noisy result as illustrated in the top curve of **Figure 9**. What happened? The probe was not faulty, but because this probe has a bandwidth of 50 MHz, it was heavily perturbed by high frequency signals coming from the board. In particular, the DC-DC converter, which is very close to the probe, was probably a culprit.

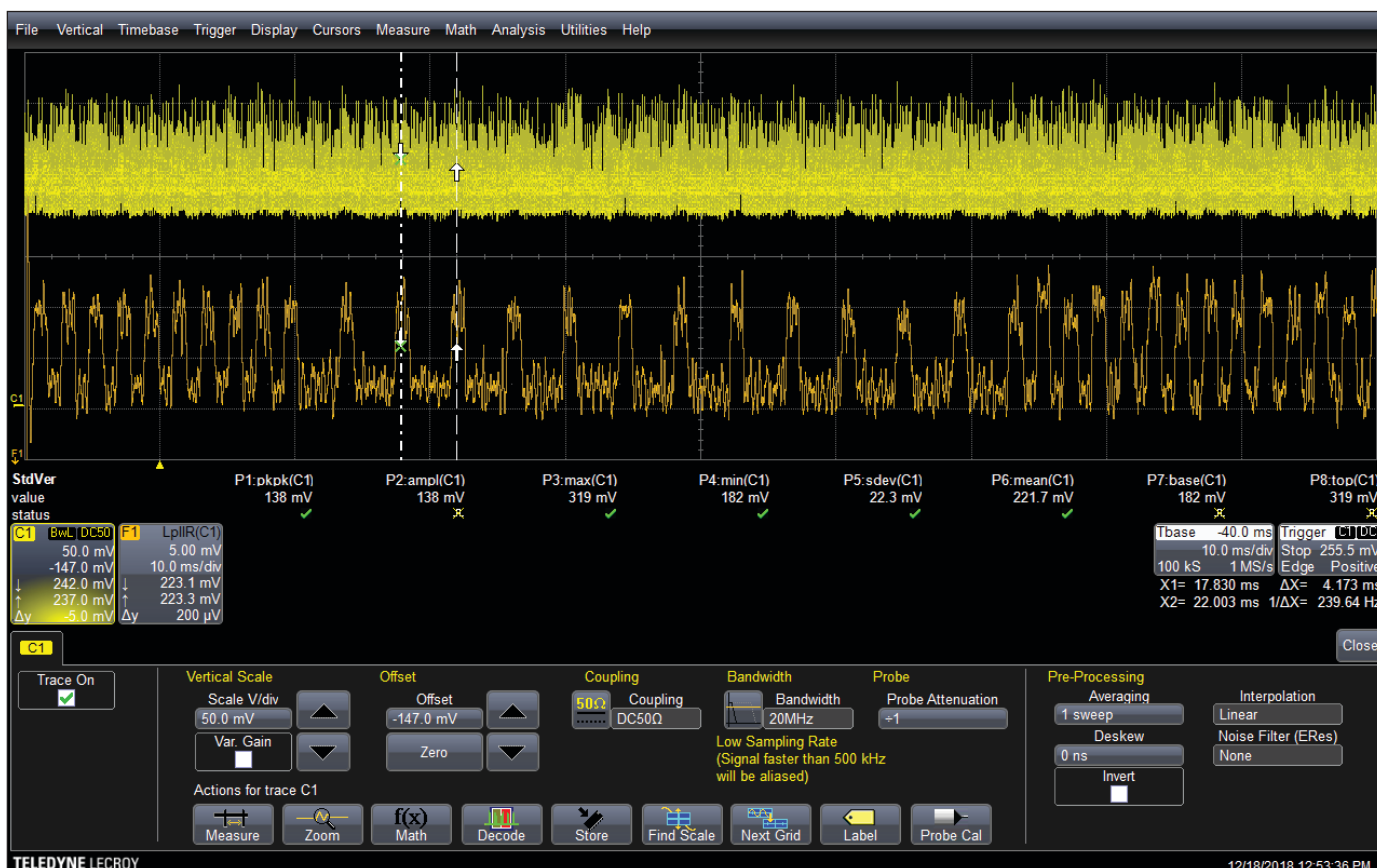


FIGURE 9 The output of the P6042A probe is very noisy (top), but the digital signal processing features of the scope allow calculation of a far cleaner low-pass filtered version (bottom).

How to improve the result and obtain a nice current signal? By adding a low-pass filter somewhere. One option would be to insert a passive low-pass filter between the current probe amplifier and scope input. The other option is to have a high-end scope with plenty of nice software features. This is the case with my WaveRunner scope. I opened the “math” menu and asked the scope to calculate a low-pass-filtered version of the input signal—using an IIR filter (infinite impulse response) with a cutoff frequency of 3 kHz. The result is the very nice second plot shown in Figure 9. Digital signal processing can be magical, can’t it?

PCB TRACK CURRENT MEASUREMENT

Current transformer probes need to surround the conductor under test, but Hall-effect ones don’t. They can simply be placed close to a wire or conductor, and will give a measurement of the current flowing through the conductor.

See, for example, **Figure 10**, where I used another nice current probe—an I-prober 520 positional current probe from Aim-TTi. This hand-held probe comes with a small amplifier, and is designed to observe currents on wires,

components leads or even PCB tracks or power planes. Its accuracy and bandwidth are lower than the Tektronix unit, but this one can be quickly moved around a circuit to check current waveforms anywhere. As an illustration, I put the probe tip on the 5 V PCB track of the EasyPIC board (**Figure 11**). The result, still using a software-based 3 kHz low-pass filter on the scope, is shown in **Figure 12**.

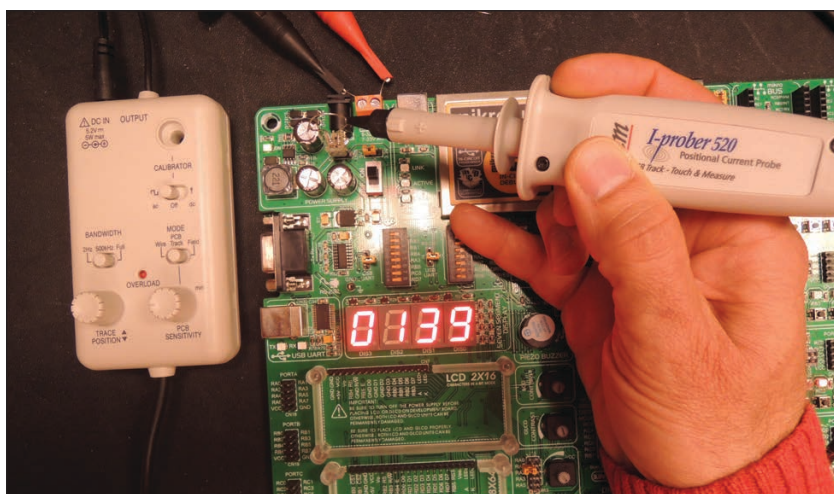



FIGURE 10 The Aim-TTi I-prober 520 and its amplifier, here simply positioned close to a wire

The current waveform was easily measured, without any cut or measurement wire added to the circuit. As shown on the plot, the DC offset of the current waveform is oscillating a little, simply because my hand was not steady during the measurement. A qualitative quick measurement like this can be invaluable.

WRAPPING UP

I hope you found that current measurement can be as easy as voltage measurement, but must be done with the proper tools and methods. A shunt resistor could be used, but must be calculated to limit the voltage drop. If the measurement is not made on the ground side, a differential amplifier is essential to transform the voltage between the two ends of the resistor into a ground-referenced voltage, which is easier to amplify, filter and measure with a scope. A differential amplifier can be home-built or bought from equipment manufacturers. Finally, both current-transformer clamps and Hall-effect sensors allow measurement of currents without any cut or shunt resistor, even directly on a PCB track.

As always, nothing is better than experimenting by yourself. Take any electronics project you've worked on—or even any battery-powered commercial stuff—and try to measure its current consumption with a scope. You can start with a shunt resistor somewhere, and then try the different methods presented in this article. You will learn a lot. And if you don't have a scope, well, buy one! Current measurement doesn't usually require more than a few megahertz of bandwidth, so even \$15 small "scopes" from China available on eBay can be used here. So, you have no excuse! Have fun! 

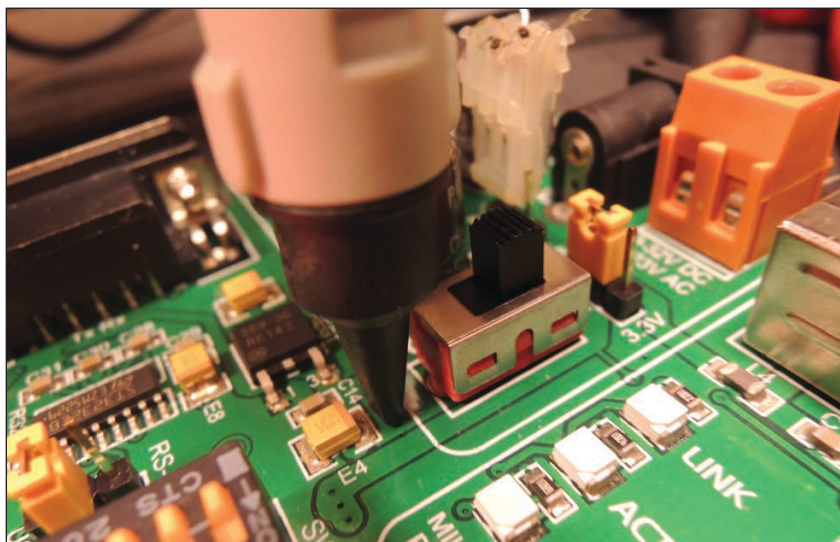


FIGURE 11

The I-prober 520 tip can also be positioned directly on a PCB track, and gives an evaluation of the current flowing through the track.

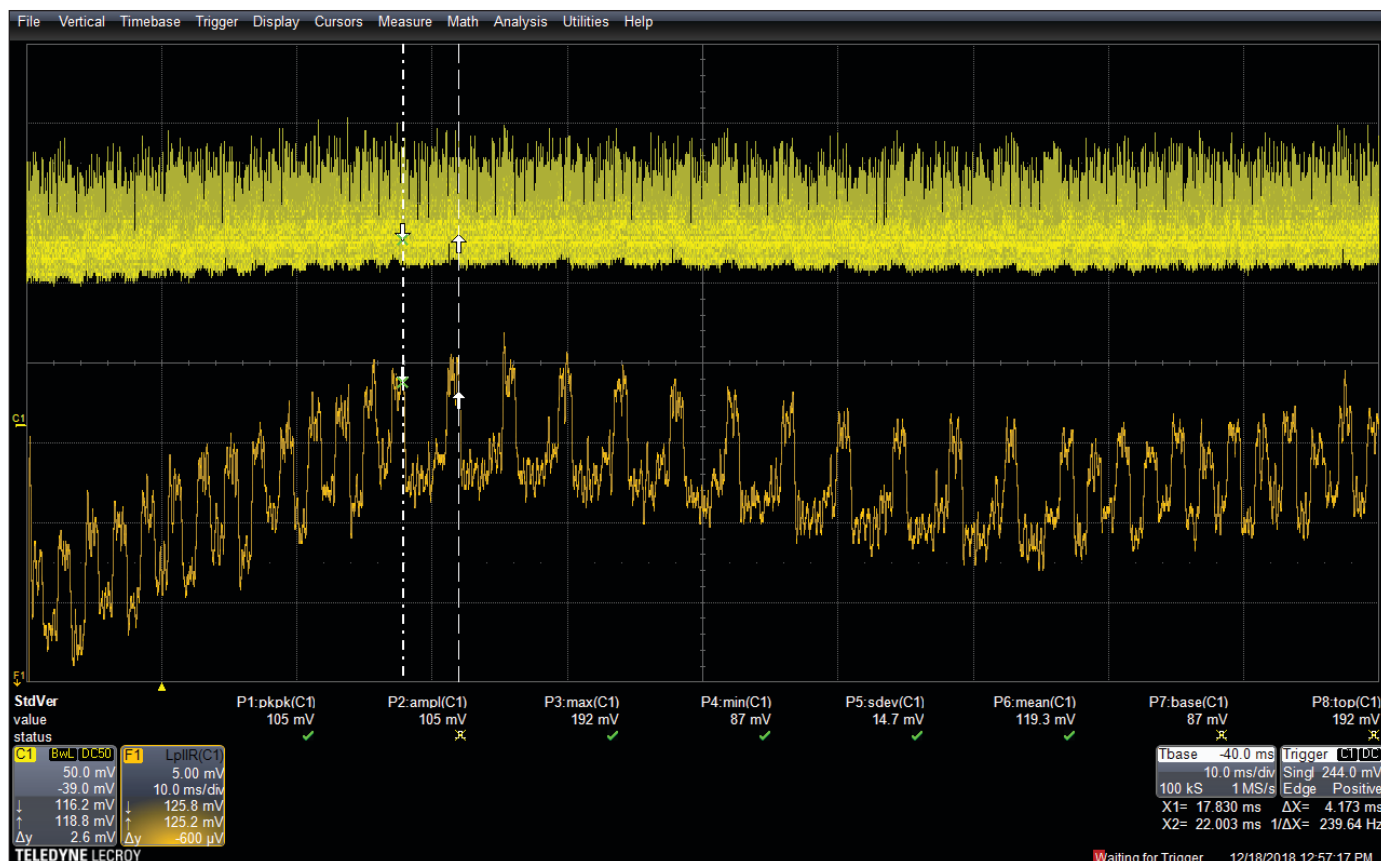


FIGURE 12

The output of the I-prober 520 shows the same waveform, but with some wander due to my unstable hand while taking the photograph.

When it comes to robotics, the future is now!



From home control systems to animatronic toys to unmanned rovers, it's an exciting time to

be a roboticist. *Advanced Control Robotics* simplifies the theory and best practices of advanced robot technologies, making it ideal reading for beginners and experts alike. You'll gain superior knowledge of embedded design theory by way of handy code samples, essential schematics, and valuable design tips.

With this book, you'll learn about:

- Communication Technologies
- Control Robotics
- Embedded Technology
- Programming Language
- Visual Debugging... and more

ADVANCED CONTROL ROBOTICS

HANNO SANDER

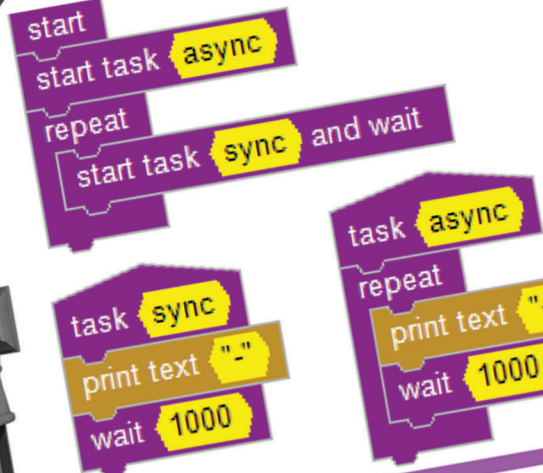
AD

Untibus e
ndipsape
berum, om
min res co
lisque eum
ad et ut ren
tis nos dolupt
volutpis as eve

Os pratis ma se
sum as sim resc
rum ne aliquo et
eat parit et que m
nis ipit accusa vole
ent magnam exorcun
veleseq uissect uest
tiur?

Obis si sinvende venis
enim dolupton et accer
tesci audions erundunt an
audandi gnimeni sintis is
dis prorunt rerem expligni
aectatus sin ea vel lum o
consed que nis dolorum aut
lorum ipiendunt et, iliquatia
taqui ad ea sim iur sequia m
utem quibus sinum remporo
adi bla nation.

HANNO SANDER



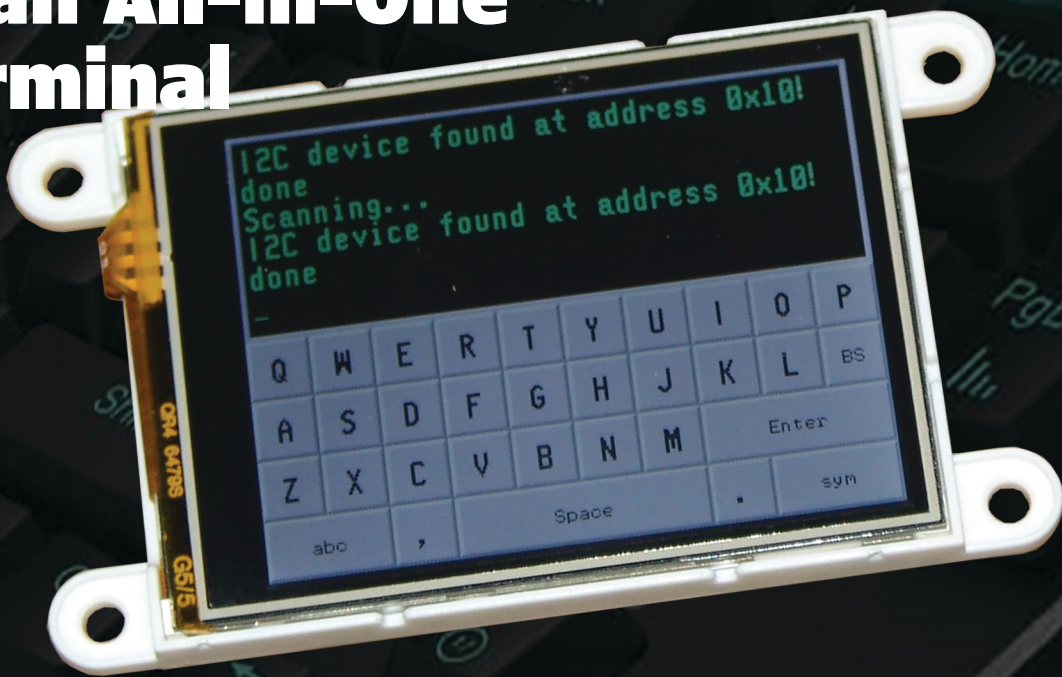
Get it today at cc-webshop.com.

From the Bench

Building an All-in-One Serial Terminal

With a Tiny Touchscreen

By
Jeff Bachiochi



Many embedded systems require at least some sort of human interface. In this project article, Jeff explains his choice of a touchscreen that met his needs. He makes use of the display's Espressif Systems ESP8266 processor and Arduino IDE support to turn the display module into a serial terminal with a serial TTL connection to other equipment.

Recently I was researching alternatives to mechanical keypads when I came across 4D Systems. 4D is an Australian company that specializes in intelligent graphics solutions integrating graphics processors with a host of display modules. Its display screens are sized from 0.9" (yup, that's not a typo!) up to 7"—many available with a resistive/capacitive touchscreen. The device I chose for this project was the Gen4-IOD-24T. At \$29, this 2.4" touchscreen display is supported by an Espressif Systems ESP8266 with Wi-Fi capability, instead of the normal graphics processor used in other models. Its current draw measures just under 200 mA at 5 V with the back lighting enabled. While I chose this for its Wi-Fi capability, I am not implementing a Wi-Fi interface for this project. I will be programming the processor using the Arduino IDE. Before I get into the project, let's take a quick tour of this product.

Figure 1 shows a PCB mounted to the back of the LCD screen. The two are contained within a plastic frame that provides convenient tabs for easy mounting. The LCD

has a diagonal display dimension of 2.4" with a resistive touchscreen applied. The micro SD card is used to hold various support files you may use in your application. These might be static text strings, fonts, images, animations/video files or general purpose storage for data logging applications. Below the SD card are the two processors, the touch controller and the ESP8266.

Those of you familiar with the ESP8266 know it's usually accompanied by a serial EEPROM (for program storage). This module has an on-board antenna. It also has a U.FL connector in case you want to use an external antenna for extended range. Connection to the device is through a 10-pin 0.5-mm pitch FPC cable. A small dongle at the opposite end of this cable provides a USB interface used in programming the module (or TTL serial). Access to power, ground, reset, TX and RX are all available on this dongle.

4D Systems gives you its basic version of Workshop4, a Microsoft Windows application that provides an integrated software development platform for the entire 4D family of processors and modules. This IDE

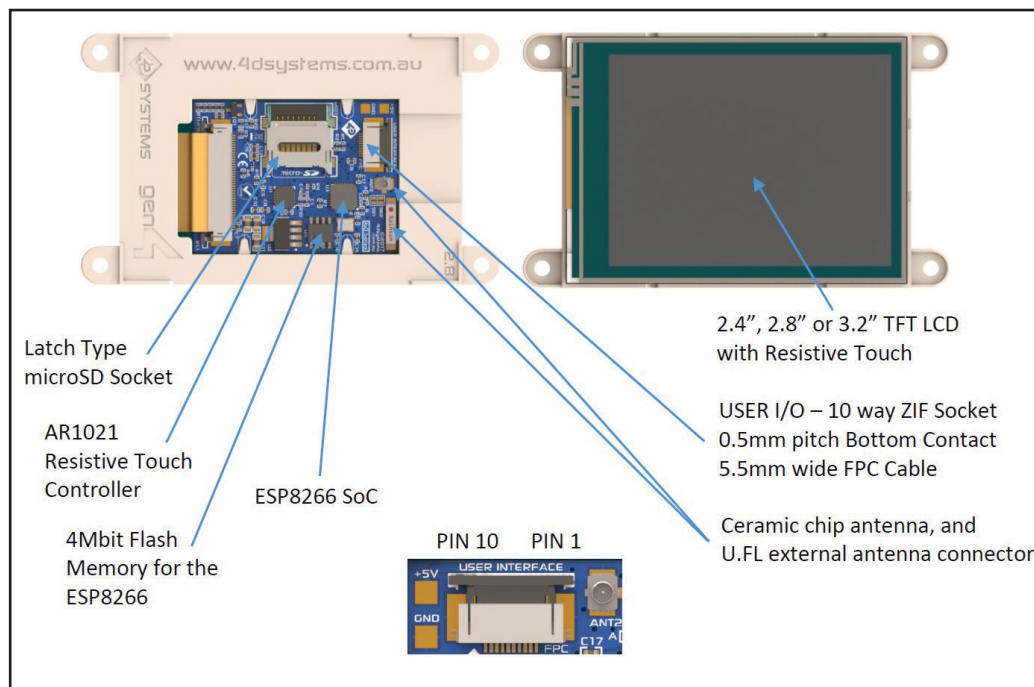


FIGURE 1

The components used on 4D systems' Gen4-IoD-24T. The newer generation of products standardize on using SMT parts and an external connection via a FPC cable.

combines an Editor, Compiler, Linker and Downloader to aid in the development of 4DGL application code. All user application code can be developed within the Workshop4 IDE or, in this case, you can use the Arduino IDE and the GFX4d Library. Workshop has some great tools for designing your application, which include Buttons, Switches, LEDs, Meters, Knobs and drawing primitives. To really customize the look, the Pro upgrade (\$70) will unlock the Smart Widget Editor, a comprehensive tool allowing you to create and animate Gauges, Sliders, Knobs and more.

ARDUINO IDE

While I could have used Workshop4 to write this entire application in 4DGL code, I wanted to show how this could be accomplished using just the Arduino IDE and the GFX4d Library. This library was developed by 4D Systems for its legacy line of Arduino products. These contained a graphics processor with a serial interface and an Arduino piggy-back interface board. The newer Gen4 modules are totally surface mount technology (SMT). The IoT modules, implemented with the ESP8266, can become a single board solution.

This project will turn the Gen4-IOD-24T module into a Serial terminal with a serial TTL connection to other equipment. Almost every project requires some kind of display interface, and most of my projects use serial I/O to collect or display data. I'll split the LCD into an upper display area and a lower keyboard area that can be used to enter data. When designing a project, I like to use a 6-pin connector that I can hang a USB dongle on. That's to get serial input in and out using my PC. This project will

```
const byte k1p[] = {
    81, 0, 0, 24, 24, 2, // Q/q
    87, 24, 0, 24, 24, 2, // W/w
    69, 48, 0, 24, 24, 2, // E/e
    82, 72, 0, 24, 24, 2, // R/r
    84, 96, 0, 24, 24, 2, // T/t
    89, 120, 0, 24, 24, 2, // Y/y
    85, 144, 0, 24, 24, 2, // U/u
    73, 168, 0, 24, 24, 2, // I/i
    79, 192, 0, 24, 24, 2, // O/o
    80, 216, 0, 24, 24, 2, // P/p
    65, 0, 24, 24, 24, 2, // A/a
    83, 24, 24, 24, 24, 2, // S/s
    68, 48, 24, 24, 24, 2, // D/d
    70, 72, 24, 24, 24, 2, // F/f
    71, 96, 24, 24, 24, 2, // G/g
    72, 120, 24, 24, 24, 2, // H/h
    74, 144, 24, 24, 24, 2, // J/j
    75, 168, 24, 24, 24, 2, // K/k
    76, 192, 24, 24, 24, 2, // L/l
    8, 216, 24, 24, 24, 1, // BS/BS (back space)
    90, 0, 48, 24, 24, 2, // Z/z
    88, 24, 48, 24, 24, 2, // X/x
    67, 48, 48, 24, 24, 2, // C/c
    86, 72, 48, 24, 24, 2, // V/v
    66, 96, 48, 24, 24, 2, // B/b
    78, 120, 48, 24, 24, 2, // N/n
    77, 144, 48, 24, 24, 2, // M/m
    13, 168, 48, 72, 24, 1, // Enter/Enter
    1, 0, 72, 48, 24, 1, // abc/ABC
    44, 48, 72, 24, 24, 2, // ./
    32, 72, 72, 96, 24, 1, // Space/Space
```

(continued)

LISTING 1

For each button, we supply 6 bytes of data: the button (ASCII value), column and row matrix position, button width and height, and font size multiplier. This listing defines the two layout arrays.

(Listing 1 continued)

```

46, 168, 72, 24, 24, 2, // ./
3, 192, 72, 48, 24, 1 // sym/sym
};
const byte k2p[] =
{
49, 0, 0, 24, 24, 2, // 1
50, 24, 0, 24, 24, 2, // 2
51, 48, 0, 24, 24, 2, // 3
52, 72, 0, 24, 24, 2, // 4
53, 96, 0, 24, 24, 2, // 5
54, 120, 0, 24, 24, 2, // 6
55, 144, 0, 24, 24, 2, // 7
56, 168, 0, 24, 24, 2, // 8
57, 192, 0, 24, 24, 2, // 9
48, 216, 0, 24, 24, 2, // 0
33, 0, 24, 24, 24, 2, // !
34, 24, 24, 24, 24, 2, // "
35, 48, 24, 24, 24, 2, // #
36, 72, 24, 24, 24, 2, // $
37, 96, 24, 24, 24, 2, // &
38, 120, 24, 24, 24, 2, // `
39, 144, 24, 24, 24, 2, // (
40, 168, 24, 24, 24, 2, // )
41, 192, 24, 24, 24, 2, // BS (backspace)
8, 216, 24, 24, 24, 1, // *
42, 0, 48, 24, 24, 2, // +
43, 24, 48, 24, 24, 2, // -
45, 48, 48, 24, 24, 2, // /
47, 72, 48, 24, 24, 2, // :
58, 96, 48, 24, 24, 2, // ;
59, 120, 48, 24, 24, 2, // +
61, 144, 48, 24, 24, 2, // =
13, 168, 48, 72, 24, 1, // Enter
2, 0, 72, 48, 24, 1, // ABC
60, 48, 72, 24, 24, 2, // <
32, 72, 72, 96, 24, 1, // Space
62, 168, 72, 24, 24, 2, // >
95, 192, 72, 24, 24, 1, // _
64, 216, 72, 24, 24, 2 // @
};

```

connect to that same interface and make the PC unnecessary. This connector matches the FTDI232 dongle—GND, CTS, VCC, TXD, RXD and RTS. We don't need the handshaking lines for this project, so two of the connections aren't necessary. The only thing you need to know when wiring your connector is whether you are connecting to Data Terminal Equipment (DTE) or Data Circuit-terminating Equipment (DCE).

For those of you who are unfamiliar with RS-232 serial, it used male connectors (DB9M or DB25M) to indicate a DTE device (like a PC) and female connectors to indicate a DCE device (for example a modem). The cable for interconnecting the two was a straight connection between a male and female connector wired pin to pin (1-1, 2-2 and so on). The sex of the connector defined the direction of the signals flowing into or out of any particular pin. This ensures that the TX of the DTE device connects to the RX of a DCE device. Schematically, most devices will have their serial output lines labeled as the signal name TX. It's good engineering practice to wire signal name TX to a TTL connector, based on its function—DTE or DCE. The connectors' pins (not the signal names) should always be labeled from a DTE point of view.

If your TX signal is going to the pin labeled TX, you know the connector is for DTE. If your TX signal is going to the pin labeled RX, you know the connector is for DCE. When you make a DTE to DCE connection, the wiring is straight through (1-1, 2-2 and so on). When you wire together two DTE devices, you will need to swap RX and TX (and other handshaking lines if used). With most of my projects being DCE, connecting an FTDI232 (DTE) is a straight-through cable. This project is a DTE, so the connection is a straight-through cable. However, when connecting this project to a FTDI232 dongle, since that is DTE to DTE, the TX and RX would need to be swapped.

Initially, all that is moot. That's because we only need to plug in a USB cable between the LCD module and a PC. The dongle on the LCD's flat ribbon cable is a TTL to USB dongle, and gets installed as a serial port. So, note the COM port number because you'll need that to connect the module with the Arduino IDE. If you need to install the GFX4d Library, you can find installation directions at github.com/4dsystems/GFX4d. If you haven't added the core board support for the ESP8266, installation instructions are included on the same page. With these installed into the Arduino IDE, you can begin an application by selecting *4D Systems gen4 IoD Range*, from the *Tools > Board* menu, along with the appropriate COM port.

With the Gen4-IOD-24T module's USB port connected to your PC, you can begin programming the application. This project will



ABOUT THE AUTHOR

Jeff Bachiochi (pronounced BAH-key-AH-key) has been writing for *Circuit Cellar* since 1988. His background includes product design and manufacturing. You can reach him at:

jeff.bachiochi@imaginethatnow.com or at:
www.imaginethatnow.com.

use the serial port (USB) as a user interface and divide the LCD screen into an upper portion (to display characters coming into the user interface) and a lower portion (to display a keyboard), allowing the user to send characters out to the user interface.

The first thing to understand is the LCD just displays graphics. The resistive touch matrix is applied to the surface of the LCD and provides the input. The coordinates of the touch matrix and the position of the graphics must be coordinated— in other words, the graphics and touch areas must coincide. This is handled by the ESP8266 and the touch controller, because there is no graphics coprocessor (as such) on this module. 4D Systems' early Arduino support for these LCD/touch modules—along with Arduino ESP8266 support—are instrumental to this module. The low bill of materials (BOM) costs allow for a very inexpensive product. So, let's begin with the most difficult function—displaying a keyboard on the LCD.

KEY TAPPING

We must begin the Arduino application by including the two libraries necessary to support this module, ESP8266 and graphics support.

```
#include "ESP8266WiFi.h"
#include "GFX4d.h"
GFX4d gfx = GFX4d();
```

The keyboard display I want to create consists of many individual buttons. The button function in the graphics library is important to this project, so let's look at its requirements.

```
drawButton(state, x, y, w, h, colorb, btext, tfont, wm, hm, tcolor);
```

where:

state = the state of the button (boolean)
x = the upper left column position (integer)
y = the upper left row position (integer)
w = the width in pixels (integer)
h = the height in pixels (integer)
colorb = background color of the button (byte)
btext = the label for the button (String)
tfont = font style (array[byte])
wm = font height multiplier (byte)
hm = font width multiplier (byte)
tcolor = font color (integer)

As you can see, there is a lot going on here. The button can be drawn in one of two states—up or down. In our case no buttons latch, so the momentary down state is only shown when your finger or a stylus is pressing on the graphic (button). The next four values place and size the button. Also note *wm* and *hm* are used to increase the size of the button; multiply the *h* and *w* by a factor greater than 1 to magnify the button's size. The remaining values choose colors, a text label and font style. To simplify things, the layout will have a standard-size button in a matrix of 10 by 4 rows. Some larger buttons use multiple matrix locations (such as the space key).

There will be two different layouts. However, the first layout is used twice—once for upper case and once for lower case. We can set up an array to hold information about each button. The first layout has 33 buttons, and the second layout has 34. For each button, we supply 6 bytes of data: the button (ASCII value), column and row matrix position, button width and height and font size multiplier. The two layout arrays are defined as shown in **Listing 1**.

We need to establish several variables in support of the keyboard. These are global variables before the `setup()` function of the Arduino application:

```
byte drawnbut[128]; // maximum number of buttons we could have in
                    // the Keyboard
int but;           // present key
int buttw;        // key returned as pressed
int lastbut;      // last key pressed
boolean skipchk;  // is this a special function key
uint16_t bcolor;  // background color
int modedelay;    // counter for special function delays
```

```

void kpabc(byte ulcase, byte bsize, int ypos,
uint16_t butcol, uint16_t tcol)
{
  String bt;
  int apos;
  int convp;
  int p[6];
  for(int n = 0; n < 33; n ++)
  {
    for(int o = 0; o < 6; o ++)
    {
      apos = (n * 6) + o;
      p[o] = k1p[apos];
    }
    if(bsize == 2)
    {
      p[1] = (p[1] / 3) * 4;
      p[2] = (p[2] / 3) * 4;
      p[3] = (p[3] / 3) * 4;
      p[4] = (p[4] / 3) * 4;
    }
    if(p[0] > 64 && p[0] < 91 && ulcase == 2)
    {
      p[0] = p[0] + 32;
    }
    bt = char(p[0]);
    if(p[0] == 13)
    {
      bt = "Enter";
    }
    if(p[0] == 8)
    {
      bt = "BS";
    }
    if(p[0] == 32)
    {
      bt = "Space";
    }
    if(p[0] == 3)
    {
      bt = "sym";
    }
    if(p[0] == 1 && ulcase == 2)
    {
      p[0] = 2;
    }
    if(p[0] == 2)
    {
      bt = "ABC";
    }
    if(p[0] == 1)
    {
      bt = "abc";
    }
    drawnbut[p[0]] = 1;
    gfx.Buttonx(p[0], p[1], p[2] + ypos,
      p[3], p[4], butcol, bt, p[5], tcol);
  }
}

```

LISTING 2 The code gets the initial keyboard displayed on the LCD.

The `setup()` function performs all the initialization. Here we start the serial port at 115,200 baud. Next, we set up the graphics, by clearing the LCD, turning on the backlight, setting the screen orientation to landscape mode and establishing the font and text size and button color. The last line in the `setup()` function will provide a display of the keyboard layout with the function call `keypad1landscape(112, bcolor, BLACK)`. This function is for layout 1 and establishes the upper left corner of the keyboard, its button color with a BLACK background.

```

void setup()
{
  Serial.begin(115200);
  gfx.begin();
  gfx.Cls();
  gfx.BacklightOn(true);
  gfx.Orientation(LANDSCAPE);
  gfx.Font(2);  gfx.TextSize(1);
  bcolor = LIGHTGREY;
  delay(100);
  keypad1landscape(112, bcolor, BLACK);
}

```

Let's take a look at what happened to accomplish this. This function calls two additional functions, `delbuttons(bcolor)` and `kpabc(1, 2, ypos, butcol, tcol)`. The first checks the whole `drawnbut[n]` array for those buttons which are drawn (`drawnbut[n]=1`) and if they are flagged as drawn (from some previous request) it erases the graphic via the `gfx.DeleteButton(n, bcolor)` graphic function. Then it marks the button as not drawn (`drawnbut[n]=0`).

```

void keypad1landscape(int ypos ,uint16_t
butcol, uint16_t tcol)
{
  delbuttons(bcolor);
  kpabc(1, 2, ypos, butcol, tcol);
}

```

Next, the alphabet matrix of buttons is drawn using the `kpabc(1, 2, ypos, butcol, tcol)` function. Note this function is passed the upper case or lower case request through its first variable =1 (lower case), along with the button size (2 times), yposition (row), button color (LIGHTGRAY), text color (BLACK). In this function, which is similar to the previous one, we will need to draw each key using the graphics function `gfx.Buttonx(p[0], p[1], p[2] + ypos, p[3], p[4], butcol, bt, p[5], tcol)`, and mark it as drawn via `drawnbut[p[0]] = 1`. After a few local variables get defined, we want to fill array `p[6]` with the values needed for the `gfx.Buttons()` function, once for each button in array `k1p[apos]`.

Do you remember the array that defined each of 33 buttons with 6 parameters `p[0] = ASCII value, p[1] = matrix column, p[2] = matrix row, p[3] = button width, p[4] = button height, and p[5] = font size multiplier`? We need to adjust some of these parameters based on the five parameters passed in the `kpabc()` function.

When `ulcase = 2` (lower case), 32 is added to each alpha ASCII character of `p[0]`, so it will be displayed as lower

sensors JUNE 25-27 expo & conference **2019**

MCENERY CONVENTION CENTER / SAN JOSE, CA

The industry's largest event dedicated to

SENSORS, CONNECTIVITY, AND SYSTEMS.

Sensors Expo & Conference is where you'll find the best of the best in the sensors industry, along with new and innovative ways to jump start your sensor solutions. Be a part of the ONLY event where technologists find opportunities and engineers innovate solutions. This year's event features three exciting days of all-new Pre-Conference Symposia, Conference Technical Sessions across 10 updated tracks, visionary Keynote Presentations, Networking Events, Exhibits, and much more!



7,000+ ATTENDEES // **330+** EXHIBITORS // **100+** SPEAKERS //
65+ CONFERENCE TECHNICAL SESSIONS // **5** PRE-CONFERENCE SYMPOSIA // **2** KEYNOTES

REGISTER TODAY!

Use code **CC100** for \$100 off Conference Passes
or a FREE Expo Hall Pass!

OFFICIAL PUBLICATION

sensors
ONLINE

INDUSTRY SPONSOR



CO-LOCATED WITH



WWW.SENSORSEXPO.COM #SENSORS19



case. When `bsize=2` (2 times), the values of `x`, `y`, `width`, `height` are adjusted, so the text is located correctly. Finally, `btext` (button text) needs to be updated with the appropriate ASCII character or special function such as BS, Enter, space, symbol, upper- and lowercase alpha.

The code shown in **Listing 2** will get the initial keyboard displayed on the LCD. Now we finish off the application with the `loop()` function. At this time we only need to loop on two functions, `checkForKey()` presses and `yield()` for the display communication to be updated.

```
void loop()
{
  checkForKey();
  yield();
}
```

Key presses are returned via the graphics function `gfx.CheckButtons()` (**Listing 3**). Note: the resistive touch areas are mapped for each key by the graphic library.

Once we have a new key, we must decide what to do about the key press. For keys that are for a special function—like the ABC, abc or sym keys—we will need to erase the keyboard and redraw a new one. For most keys we need to send the ASCII character out the serial port. And that is it—unless you want to display characters coming in through the serial port.

VIEWING TEXT

Receiving characters is a much easier process. The graphics function `gfx.TextWindow()` handles everything we need for a scrolling display of input text. We need to add the support variables for this:

```
String signOn = "Serial Terminal Gen4-IoT-24T"; // The sign on message
byte inChar; // character from the UART
uint16_t twcol; // text color
String inputString; // string of text built from the input characters
```

In the `setup()` function, options are set that define aspects of the text window before it is initialized. Finally, the text window can be drawn with the upper left corner at `column=0`, `row=0`, with a `width=320` pixels and a `height=112` pixels. The text displayed in the window will be `twcol=PALEGREEN` with `BLACK` as the text window background and `LIGHTGRAY` as the window's border.

A Sign-On message will then be displayed at the top of the text window. With a window height of 112, the window can hold six lines of text characters, because the text font is 5x8 pixels and the font magnification is 2. If you remove the upper and lower borders from the window height, you now have an inside height of approximately 100 pixels, and since each character is 16 pixels high, $100/16 = 6$ rows of characters.

```
gfx.ScrollEnable(true); // enable the scrolling function
gfx.SmoothScrollSpeed(0); // slow down scrolling
twcol = PALEGREEN; // text color
gfx.Font(2); // set the font used
gfx.TextSize(1); // set the text magnification
gfx.TextWindow(0, 0, 320, 112, twcol, BLACK, LIGHTGREY);
// draw the text window
gfx.TWcursorOn(false); // set cursor off
inputString.reserve(100); // set a bit of memory for strings
displayStringTW(); // display string in text window
```

Within the `loop()` function we add a check for serial characters.

```
checkForSerialIn(); // check for UART RX
```

Each character is displayed as received. The function `displayTW()` uses the graphic function `gfx.TWwrite(inChar)` to display the character. This is different from the `displayStringTW()` function, which uses the graphic function `gfx.TWprintln(inputString)`. Alternately, we could have saved the string until a CR was received, and then used the graphic function `gfx.TWprintln(inputString)` to display it, but I want to show characters as they are received.

```

void checkForSerialIn()
{
    inChar = 0;
    if(Serial.available() > 0)
    {
        inChar = Serial.read();
        displayTW();
    }
}

void displayStringTW()
{
    gfx.TWprintln(inputString);
}

void displayTW()
{
    gfx.TWwrite(inChar);
}

```

Sometimes you want to print text on the display at a specific location. Presently, the graphic routines don't allow for this. I wrote some code to fake out the display. In other words, it gathers positional text and substitutes data within six strings of data, which would then be printed to the display all at once. This won't be necessary with the next release of the GFX4d graphics Library. It will contain text positioning functions.

One other note if you are using this library: a couple of lines were inadvertently left out of the present GFX4d.cpp file in the library. You can fix this by adding a couple of lines to reinitialize the x and y positions. Without these, your program will crash consistently if you use this function—which I found out the hard way!

```

void GFX4d::TWcls()
{
    RectangleFilled(tctx - 3, tctx - 3,
        (tctx - 3) + (tctx + 2) - 1, (tctx - 3)
+
        (tctx + 2) - 1, tctx);
    twcurx = tctx;
    twcury = tctx;
    twxpos = 0; // added 11 19 2018
    twypos = 0; // added 11 19 2018
    for(int n = 0; n < sizeof(txtbuf); n++){
        txtbuf[n] = 0;
    }
}

```

DISPLAY DUTY

I can use my finger to tap keys on this small 2.4" display, however it is much easier to use a stylus. If you find you want to use the total area of the display, you can easily modify this and get rid of the keyboard entirely. **Figure 2** shows Gen4-IOD-24T module being used to display the results from a previous project. I added a one-shot routine to turn off the backlight 5 seconds after the last key is pressed. It also keeps track of the backlight state. So, when it sees a touch with the backlight off, it disregards that character, but turns on the backlight.

```

void checkForKey()
{
    buttw = gfx.CheckButtons();
    if(buttw != lastbut && buttw > 0)
    {
        but = buttw;
    }
    if(modedelay > 0)
    {
        modedelay = modedelay - 1;
    }
    skipchk = false;
    if(modedelay > 0 && but == 64)
    {
        skipchk = true;
    }
    if(modedelay > 0 && but == 95)
    {
        skipchk = true;
    }
    if(modedelay > 0 && but == 1)
    {
        skipchk = true;
    }
    if(modedelay > 0 && but == 2)
    {
        skipchk = true;
    }
    if(skipchk == false)
    {
        if(but > 31 && but < 127)
        {
            //gfx.TWwrite(but);
            Serial.write(but);
        }
        if(but == 13)
        {
            //gfx.TWwrite(but);
            Serial.write(but);
            gfx.ButtonUp(13);
        }
        if(but == 8 || but == 127)
        {
            //gfx.TWwrite(8);
            Serial.write(8);
        }
        if(but == 1)
        {
            keypad2landscape(112, bcolor, BLACK);
            but = 2;
            skipchk = true;
            modedelay = 30;
        }
        if(but == 2 && skipchk == false)
        {
            keypad1landscape(112, bcolor, BLACK);
            but = 1;
            skipchk = true;
            modedelay = 30;
        }
        if(but == 3 && skipchk == false)
        {
            keypad3landscape(112, bcolor, BLACK);
            modedelay = 30;
        }
    }
    lastbut = buttw;
}

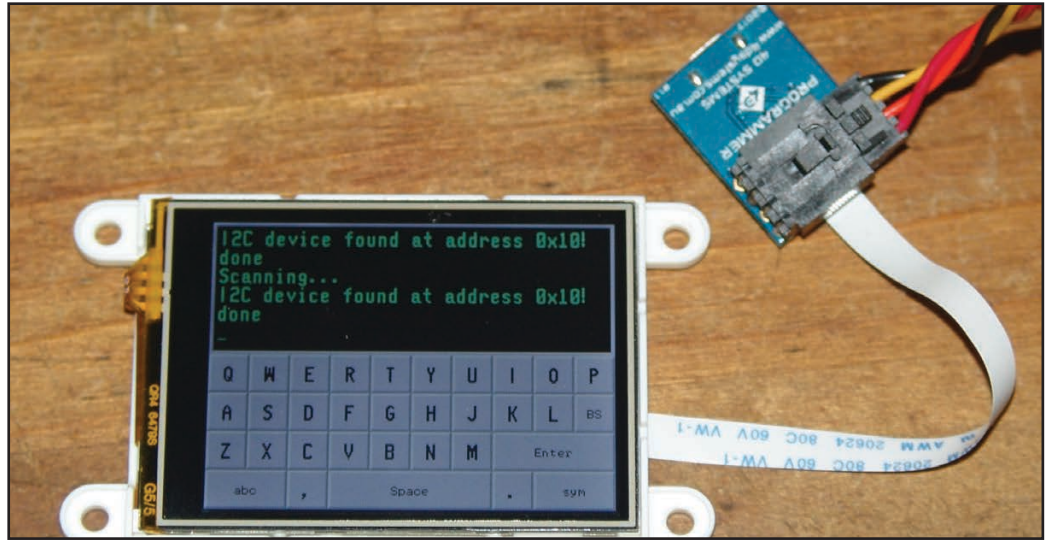
```

LISTING 3

Key presses are returned via the graphics function `gfx.CheckButtons()`. Note: the resistive touch areas are mapped for each key by the graphic library.

FIGURE 2

A photo of this project being used to display data from a previous project. Just four TTL connections are needed at the end of the FPC cable. Alternatively, the USB connector can be used.



Although I didn't show how the 4D System's Workshop IDE could be used for developing projects, you may wish to investigate this free application for its complete line of LCD modules. A true graphic controller can add functions just not attainable with the ESP8266. Each can operate in a stand-alone mode for some simple applications, or be connected to an external system to handle all the display needs.

I look forward to using this display for some

Wi-Fi projects in the future. My weather station, for example, could use a local display. This module has the dimensions that fall within the size of a single gang electrical box, which means it doesn't have to stick out like a sore thumb when mounted on the wall. Hmm... I guess I could replace many of my mechanical light switches and control all my Wi-Fi outlets and lights from each module. Too much to do, so little time, and never a dull moment.

ADuC841
Microcontroller
Design Manual
From Microcontroller Theory to Design Projects

ADuC841 Microcontroller Design Manual

From Microcontroller Theory to Design Projects

If you've ever wanted to design and program with the ADuC841 microcontroller, or other microcontrollers in the 8051 family, this is the book for you. With introductory and advanced labs, you'll soon master the many ways to use a microcontroller. Perfect for academics!

Buy it today!

cc-webshop.com

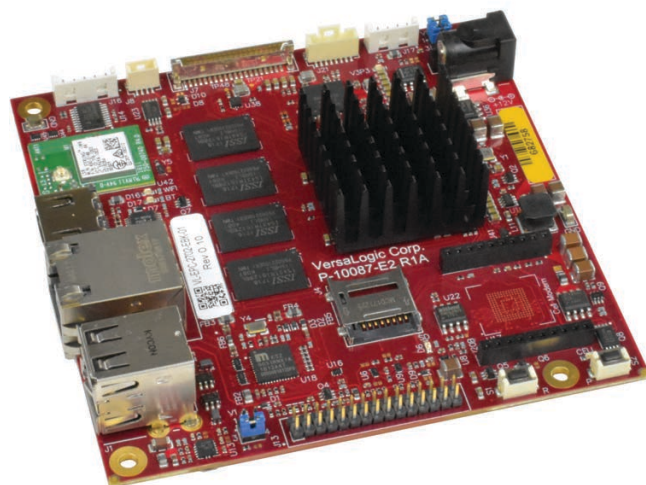
PRODUCT NEWS

i.MX6-Based SBC Offers Global Cellular Expansion

VersaLogic has announced the Swordtail SBC that features models with either the NXP i.MX6 Quad (quad core), or the i.MX6 DualLite (dual core) processors. The SBC includes on-board Wi-Fi, Bluetooth and a cellular plug-in socket. At home in hostile environments the compact 95 mm x 95 mm computer board is rated for operation at full industrial temperature range (-40° to +85°C).

Both Swordtail models feature soldered-on memory, and a variety of I/O connections. In addition to wireless capability, the on-board I/O includes a Gbit Ethernet port with network boot capability, two USB 2.0 Ports, serial I/O (RS-232), CAN Bus, microSD socket and I²C interface. The boards can accommodate up to 32 GB of on-board flash storage.

Designed and tested for industrial temperature (-40°C to +85°C) operation, VersaLogic's Swordtail also meets MIL-STD-202H specifications to withstand high impact and vibration. The Swordtail single board computers (EPC-2702), will be available Q2 2019 from both VersaLogic and Digi-Key. OEM pricing starts at \$236.



VersaLogic | www.versalogic.com

High-Temp Motor Control is Target for 32-Bit MCU Offerings

Renesas Electronics has announced the expansion of its RX24T and RX24U Groups of 32-bit MCUs to include new high-temperature-tolerant models for motor-control applications that require an expanded operating temperature range. The new RX24T G Version and RX24U G Version support

operating temperatures ranging from -40°C to +105°C, while maintaining the high speed, high functionality and energy efficiency of the RX24T and RX24U MCUs.

Software can be developed using the RX24T and RX24U CPU cards combined with the 24 V Motor Control Evaluation Kit which enables developers to create motor control applications in less time. The 32-bit RX24T and RX24U features a maximum operating frequency of 80 MHz. It is equipped with peripheral functions for motor control such as timers, A/D converter, and analog circuits that enable efficient control of two brushless DC motors by a single chip.

The RX24T G Version and RX24U G Version are available now in mass production. The RX24T covers 11 models with pin counts ranging from 64- to 100-pins and memory sizes from 128 KB to 512 KB. The RX24U covers six models with pin counts ranging from 100- to 144-pins and memory sizes from 256 KB to 512 KB.

Renesas High-Temperature-Tolerant Versions of RX24T and RX24U 32-bit MCUs for Motor Control



Renesas Electronics
www.renesas.com

4:1 Input DC-DC Converters Boast 1" x 1" Footprint

RECOM has released its REC15E-Z series of 15 W isolated DC-DC converters that featured wide input ranges at low cost in the popular 1"x1" case size. This saves a significant amount of PCB space, while the wide input ranges increase flexibility by accepting several standard bus voltages. The REC15E-Z DC-DC converters are fully-specified devices with 15 W, no minimum load, 1,600 VDC isolation, high efficiency up to 90% and low ripple/noise. The REC15E-Z series was designed for cost-sensitive applications where board space is at a premium. The wide 4:1 input ranges accept 9 V to 36 V or 18 V to 75 V to cover multiple supply options such as lead-acid or lithium

batteries or 12/24/36/48 V industrial bus voltages.

The inputs are protected against transients of up to 100 V and feature UVLO to protect batteries from being over-discharged. The single or dual outputs are continuously protected against short circuit and overload conditions and can drive high-capacitive loads. They are fully certified to industrial EMC and safety standards and come with a three-year warranty. Samples and OEM pricing are available from all authorized distributors or directly from RECOM.

RECOM | www.recom-power.com

PRODUCT NEWS

Rugged Touch Panel Computer Targets Railway System Designs

ADLINK Technology has released its latest Driver Machine Interface (DMI) touch panel computer, the DMI-1210, designed specifically for train control and driver information display. Powered by the Intel Atom x5-E3930 processor (formerly Apollo Lake) and featuring a 12.1" (4:3) high resolution color display, 5-wire resistive touch screen and securable I/O interface, the DMI-1210 can be deployed as an HMI unit for driver's desks, control panel for passenger information systems, surveillance system control/display unit or in railway diagnostics and communications applications.

The DMI-1210 is an EN 50155 certificated driver interface that offers train radio display, electronic timetable, and diagnostic display functions and additional functionality such as train data recorder. The DMI-1210 supports full range DC power input from +16.8 V to +137.5 V DC. Optional MVB, GNSS, 3G/LTE, WLAN and Bluetooth through add-on modules give system integrators the necessary tools to expand use case possibilities. With ADLINK's built-in Smart Embedded Management Agent (SEMA) management and status LEDs on the front panel, the DMI-1210 provides easy and effective health monitoring and system maintenance.

ADLINK Technology | www.adlinktech.com



45 V, Zero-Drift Op Amp Provides On-Chip EMI Filtering

Microchip Technology has announced the MCP6V51 zero-drift operational amplifier (op amp). The new device provides ultra-high-precision measurement while minimizing the increasing influence of high-frequency interference by offering a wide operating range and on-chip electromagnetic interference (EMI) filters. The self-correcting zero-drift architecture of the MCP6V51 enables ultra-high Direct Current (DC) precision, providing a maximum offset of $\pm 15 \mu\text{V}$ and only $\pm 36 \text{ nV}$ per degree Celsius ($\text{nV}/^\circ\text{C}$) of maximum offset drift. Ideal for applications such as factory automation, process control and building automation, the MCP6V51 also supports an extremely wide operating voltage range, from 4.5 V to 45 V.



With the proliferation of wireless sensors and capabilities, high-frequency interference within sensitive analog measurement is becoming a critical consideration. The additional on-chip EMI filtering within the MCP6V51 provides protection from these unwanted and unpredictable interference sources. The MCP6V51 is available now for sampling and volume production in both 5-lead SOT-23 and 8-lead MSOP packages. Prices begin at \$0.98 USD per 10,000 units for the SOT-23-5 package.

Microchip Technology | www.microchip.com

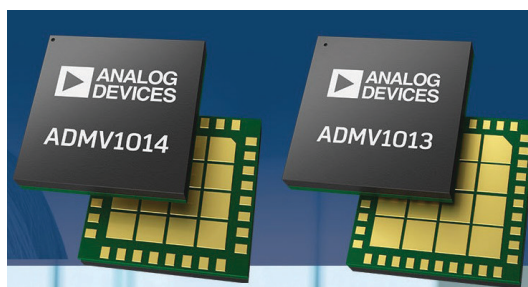
Up and Downconverter Pair Offers 24- to 44-GHz Frequency Range

Analog Devices has announced the ADMV1013 and ADMV1014, a paired highly integrated microwave upconverter and downconverter, respectively. These ICs operate over a very wide frequency range with 50 Ω -match from 24 GHz up to 44 GHz, facilitating ease of design and reducing the costs of building a single platform that can cover all 5G mm Wave frequency bands including 28 GHz and 39 GHz.

The chipset is capable of flat 1 GHz RF instantaneous bandwidth supporting all broadband services as well as other ultra-wide bandwidth transceiver applications. Each upconverter and downconverter is highly integrated, comprising I (in-phase) and Q (quadrature-phase) mixers with on-chip programmable quadrature phase-shifter configurable for direct conversion to/from baseband

(operable from DC to 6 GHz) or to an IF (operable from 800 MHz to 6 GHz). Also included on-chip are voltage variable attenuators, transmit PA driver (in the upconverter) and a receive LNA (in the downconverter), LO buffers with x4 frequency multiplier and programmable tracking filters. Most programmability functions are controlled via an SPI serial interface.

The ADMV1013 is offered in a 40-pin, 6 mm x 6 mm LGA, and the ADMV1014 is in a 32-pin, 5 mm x 5 mm LGA package. Pricing per 1,000 each for the ADMV1013 starts at \$90.79 in a 40-pin, 6 mm x 6 mm CSP package. Pricing per 1,000 each for the ADMV1014 starts at \$88.37 in a 32-pin, 5 mm x 5 mm CSP package.



**Analog Devices
www.analog.com**

IDEA BOX

The Directory of PRODUCTS & SERVICES

AD FORMAT:

Advertisers must furnish digital files that meet our specifications (circuitcellar.com/mediakit).

All text and other elements MUST fit within a 2" x 3" format.
E-mail adcop@circuitcellar.com with your file.

For current rates, deadlines, and more information contact Hugh Heinsohn at 757-525-3677 or Hugh@circuitcellar.com.

ALL ELECTRONICS

Surplus & New Parts & Supplies Since 1967

LEDS · CONNECTORS · RELAYS
SOLENOIDS · FANS · ENCLOSURES
MOTORS · WHEELS · MAGNETS
PC BOARDS · POWER SUPPLIES
SWITCHES · LIGHTS · BATTERIES
and many more items...

We have what you need for your next project.

Discount Prices
Fast Shipping

www.allelectronics.com

CCS Inc. PREMIER MICROCHIP THIRD PARTY PARTNER

C Compiler for PIC® MCU

- Easy to learn - Powerful Tools
- Built-in functions for Rapid Development
- Optimized for Unique PIC® Architecture
- Easy code Portability between devices

www.ccsinfo.com/CC419

PIC® MCU is a registered trademark of Microchip Technology Inc. in the U.S. and other countries.

Technologic Systems

Single Board Computer

TS-7250-V2
1GHz ARM Computer with Customizable FPGA-Driven PC/104 Connector and Several Interfaces at Industrial Temp

www.embeddedARM.com

Circuit Cellar 2018 Archive

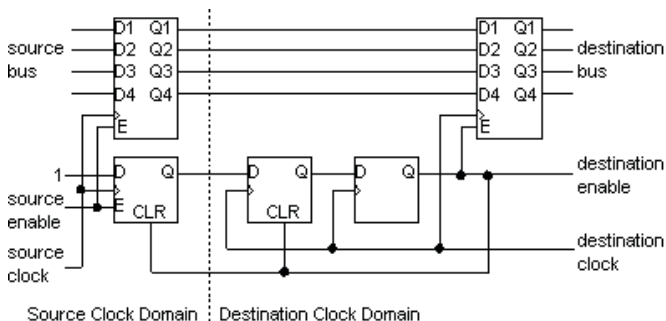
Order yours today

cc-webshop.com

Issues # 330-341 - CD #23
Copyright 2018, KCK Media Corp.

Answer 1— The obvious multi-bit synchronizer built with dual pipeline registers does indeed address the problem of metastability, but it leaves open the possibility that different data bits in the bus will change at different clock edges in the destination domain. This means that the bus will occasionally contain “invalid” values that are a mix of bits from the previous value and bits from the new value.

Answer 2— In systems where the data changes less often than every 3-4 clocks of the destination domain, you can use a single pipeline register to capture the data. The trick to avoiding metastability is to enable this register only when the data isn't changing. For this, you create a separate “update” signal that gets passed in the usual way from the source domain to the destination domain. One possible implementation is shown in Figure 1.



This implementation presumes that there is an “enable” or “valid” signal that flows in parallel with the data in each clock domain anyway, and this circuit creates an enable pulse in the destination domain for each one that occurs in the source domain. Note that two of the flip-flops have an asynchronous clear function.

Answer 3— Ethernet relies on message-level error checks to detect collisions, which means that the two messages that

TEST YOUR EQ

Contributed by David Tweed

collided are both lost and must eventually be retransmitted. Since there's a nonzero chance that a collision can happen on every (re-)transmission, there's no guaranteed upper bound on the delivery time of any particular message.

On the other hand, CAN transmitters monitor the physical medium on a bit-by-bit basis, and if two of them happen to start sending at the same time, the first one to send a “recessive” bit at the same time the other is sending a “dominant” bit will recognize this at once and abort its message. The other transmitter continues. None of the other devices on the network are aware of the collision, and that message experiences no delay. After that, the transmitter that experienced the collision is free to try again, and it won't risk a collision with that same message again.

Even if it experiences a collision with a different higher-priority message, there is a definite limit on the number of those that can occur, and therefore, there's a definite upper bound on the delivery time of the message. This makes CAN suitable for “hard” real-time applications for which ordinary Ethernet cannot be used—although there are now higher-level protocols that have been invented that can be used with Ethernet to mitigate this problem.

Answer 4— Since there's no requirement for collisions to be detectable within one bit time, the raw Ethernet bit rate can be much higher, which means that the average network throughput can be correspondingly higher. CAN typically maxes out at about 1 Mbps (on small networks), while Ethernet starts at 10 Mbps and goes up from there.

For more information:
circuitcellar.com/category/test-your-eq/

TESTS YOUR EQ

CC Vault

Unlock the power of embedded design.

A vault of need-to-know information in the fields of embedded hardware, embedded software, and computer applications

Order yours today! cc-webshop.com

This pocket-sized vault comes fully loaded with every issue of Circuit Cellar magazine and serves as an unparalleled resource for embedded hardware and software design tips, schematics, and source code.

*CC Vault is a 16-GB USB drive.

The Future of Robotics

Dealing with Challenges and Tradeoffs in Motion Control



By
James English, President
and CTO, Energid Technologies

The global motion control industry saw trends similar to the machine vision market with an estimated 8% growth in 2018. Within this broader market stability, robot motion control has continued a focus on innovation to drive growth in new industries and applications. Robot motion control—especially for complex articulated arms—is expected to transcend the overall motion control market in coming years.

Motion control software moves robot arms through the action of rotation and sliding joints, and mobile robots through locomotion and steering. This controlled motion enables robot tasks, which are done with tools (end effectors) on the robot. Task can be manipulative, as when using a gripper, or sensory, as when positioning and capturing data with a camera. These two concepts—movement and tasks—are defining components for using a robot.

CONTROLLING THE JOINTS

As illustrated in **Figure 1**, the tools/end effectors do the work while the joints are controlled, and the relationship between the two is complex. An equation describing the placement of a probe held by a robot arm can take pages and pages of trigonometric functions. This is the easy direction. Going the other way—calculating the control solution of how to place the joints to achieve a desired tool position—may not even have an equation. It may only be solvable iteratively.

Robots like the one shown in **Figure 2** have more actuators than the minimum needed for grasping a screwdriver. This actuator redundancy empowers a robot but complicates motion control. If you consider the human body, our extra joints enable many ways—an infinite number, in fact—to take the leftover sandwich out of the refrigerator, and exploiting the redundancy lets us reach around milk cartons, balance, and move smoothly to reduce joint stress and avoid joint limits. This takes lots of brainpower, however. Robots with redundancy benefit from having the potential to move with the same smooth, efficient control, but that takes lots of computer processing power.

When there are multiple ways for a robot to accomplish a task, the chosen way should have special qualities, such as maximizing distance from a collision, improving strength, minimizing time, avoiding workspace limits, reducing power consumption and improving accuracy. The best motion will usually be a combination of these—and other—pure qualities.

Motion control must also incorporate constraints. Robot joints have speed and acceleration limits. Actuators have maximum torque or force. Physical parts of the robot cannot overlap in space, and joint limits cannot be exceeded. These are constraints imposed by

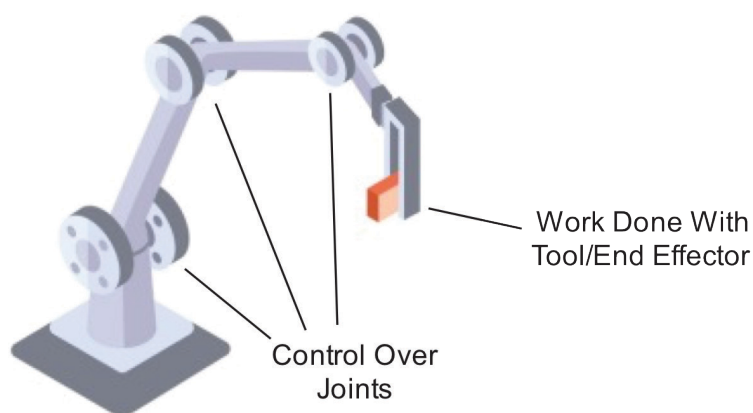


FIGURE 1

The motion control challenge: The joints are controlled while the tool (or end effector) does the work, and the relationship between the two is complex.

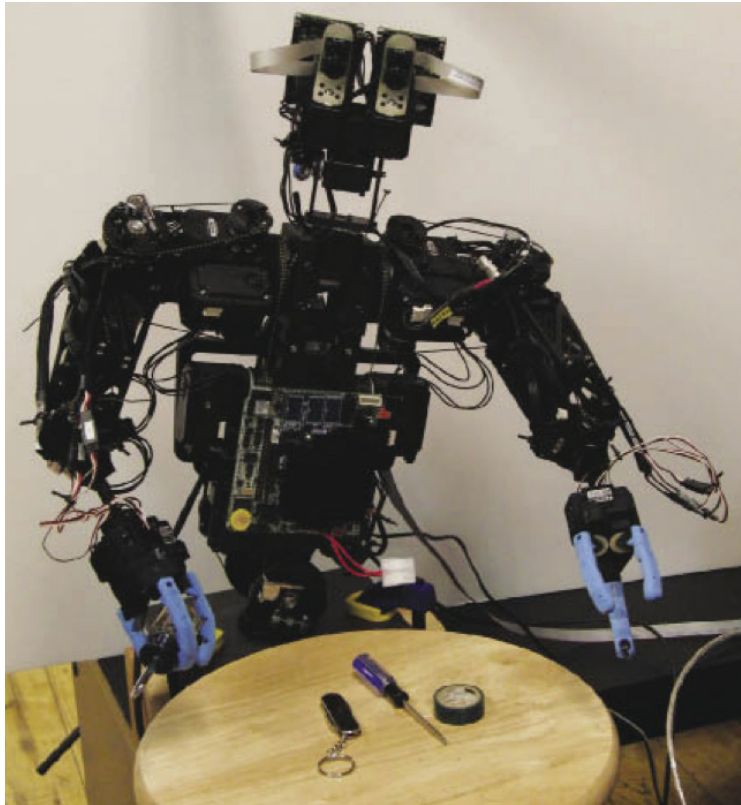


FIGURE 2

Some robots have more degrees of freedom than the minimum required to do a task. These kinematically redundant robots are powerful but can be hard to control.

the physical reality of the robot and the world. The desired tasks, constraints and optimizations combine to make robot motion control a challenge.

REAL-TIME PROCESSING

A variety of mathematical techniques have been developed to address the challenge. While special-purpose equations can be used, a common technique applies the so-called manipulator Jacobian matrix. The Jacobian is a mathematical object that describes tool velocity as a function of joint velocity in a simplified way. It sidesteps the complicated direct calculation of positions. It has a simplified form, so it is easier to invert to solve the control problem. The only drawback is that it works with velocities rather than positions. Positioning using the Jacobian requires algorithmic feedback techniques.

Though the Jacobian can almost always be defined, calculated and inverted for control, challenges remain


for a complete solution. Jacobian-based control is local in nature, and some problems have to be solved globally. Global control relates to large movements with flexibility in the path so long as the endpoints are correct, while local control relates to precisely defined—and usually small—movements. Many robot tasks are performed using a combination of global and local control, and how the optimizations are selected and implemented depend on qualities of the robot, its task and its environment.

Managing higher time derivatives of joint position is also important. Many robots today generate full paths offline before motion starts. Offline path generation allows the use of the future states of the robot in calculations about earlier states. This helps in limiting the higher derivatives of motion that can cause vibration, such as jerk, the derivative of acceleration. The drawback of this approach, however, is that knowledge is incomplete before motion begins. Once the robot starts on a pre-calculated path, it cannot respond to environmental and user-input changes. Sometimes a tradeoff must be made, using slower and more cautious motion to control higher derivatives in real time.

Playing into this tradeoff is the speed of calculation of the control algorithms. When applied in real time, speed is critical. Exploring multiple alternatives—time step by time step—and choosing the best is a powerful algorithmic approach. Faster implementations allow more alternatives and improved control.

There is a chasm between algorithm existence on paper or in demonstration and its practical use because making an algorithm usable is itself difficult. Implementations need to be robust and even rare problems need to be addressed. The implementation must accommodate inevitable deviations in the robot type, the environment, and tasks. And it must easily integrate with other software.

PRACTICAL ROBOTIC SOLUTIONS

Addressing these needs today are multiple free-open-source and commercial software packages. One commercial example is Energid Technologies' Actin, which controls arms in areas such as manufacturing, medical, and energy applications. A prominent example of the use of Actin is in bin picking, where one part at a time must be removed from a random pile of parts. Bin picking requires motion control that is fast and smooth while avoiding collisions with the bin holding the parts and with other parts in the bin. Advanced motion control enables this robotic bin picking to be practical, just one example of the way motion control advancement will help robotics across many industries. 

James English is the president and chief technical officer of Energid Technologies. Specializing in automatic remote robotics, James leads project teams in the development of complex robotic, machine vision and simulation systems. Prior to founding Energid, his background was in software development in the engineering and aerospace industries where he held key R&D positions with Raytheon and MAK Technologies. He has authored many journal and conference papers and multiple patents related to the control and simulation of robotic systems.

RESOURCES

Energid Technologies | www.energid.com

STRONG FOUNDATION

Whether you are an
EMS, CM or OEM,
let our bare boards be the foundation
you build your reputation upon!

Technology:

Up to 50 Layers
Any Layer HDI
Sequential Lamination
Blind / Buried Vias
Laser Drilling / Routing
Heavy Copper

Materials:

Fr4
Metal Core
Isola
Rogers
Polyimide - Flex
Magtron

**We will make only what is needed,
when it's needed,
and in the amount needed.**

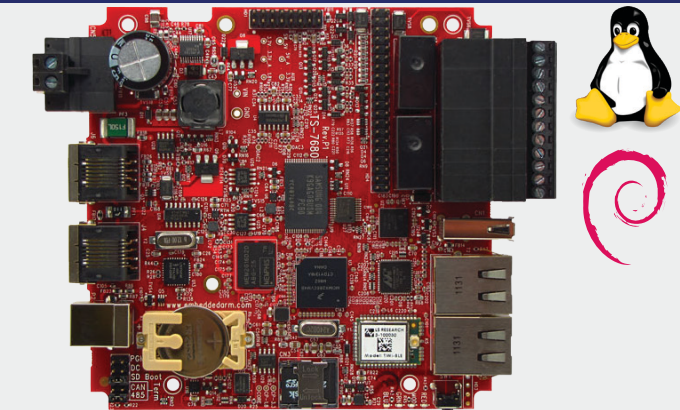
You no longer have to worry about long shelf life
or tie your capital in bare board inventory.

Accutrace[®] inc.

www.PCB4u.com sales@PCB4u.com

SAM & ITAR Registered UL E333047 ISO 9001 - 2008

FROM THE DEEP BLUE SEA TO THE WILD BLUE YONDER



TS-7680

Low Power Industrial
Single Board Computer with
WiFi and Bluetooth

\$159
Qty 100

The TS-7680 is designed to provide extreme performance for applications demanding high reliability, fast boot-up/startup, and connectivity at low cost and low power. Because there are so many features packed on to one single board computer you will see a reduction in payload weight since there is no need for additional boards, micro-controllers, or peripherals.

Rated for industrial temperature range of -40°C to $+85^{\circ}\text{C}$ the TS-7680 is deployed in fleet management, pipeline monitoring, and industrial controls and is working in some of the most demanding places on Earth.

The TS-7680 will help you perform at your very best in a variety of critical missions.



Made in USA
with Global Parts

 **Technologic**
Systems