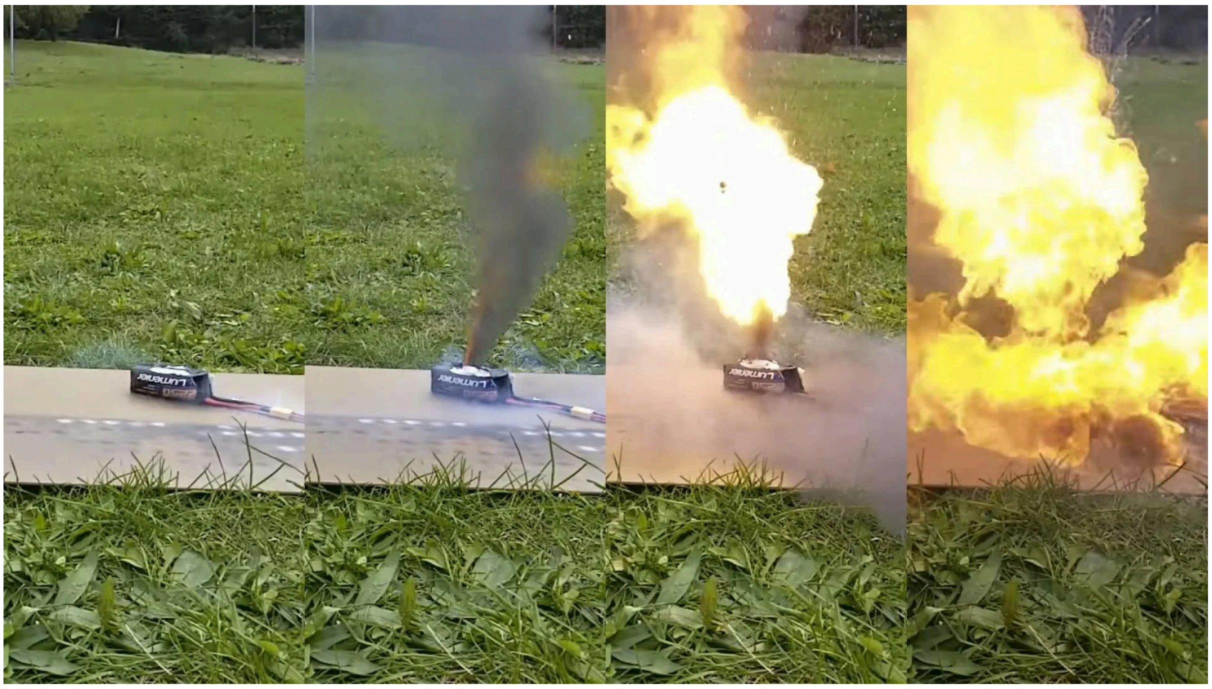
Title: **Explosive Gadgets: Analyzing the Mysterious Events in Lebanon**

Introduction:

Recent media reports have stirred significant intrigue regarding a series of explosions involving gadgets in Lebanon on September 17 and 18, 2024. Approximately 5,000 pagers reportedly exploded in a coordinated incident, followed by similar reports of smartphones and other devices experiencing spontaneous combustion. While lithium batteries pose inherent risks, the scale and nature of these events raise critical questions. How could such widespread gadget malfunctions occur? This article delves into the science behind lithium batteries, explores the feasibility of remote activation, and examines the potential for tampering or modification of devices, all while questioning the veracity of subsequent reports.

Media reports

Network_support · bastyon.com 17 min read

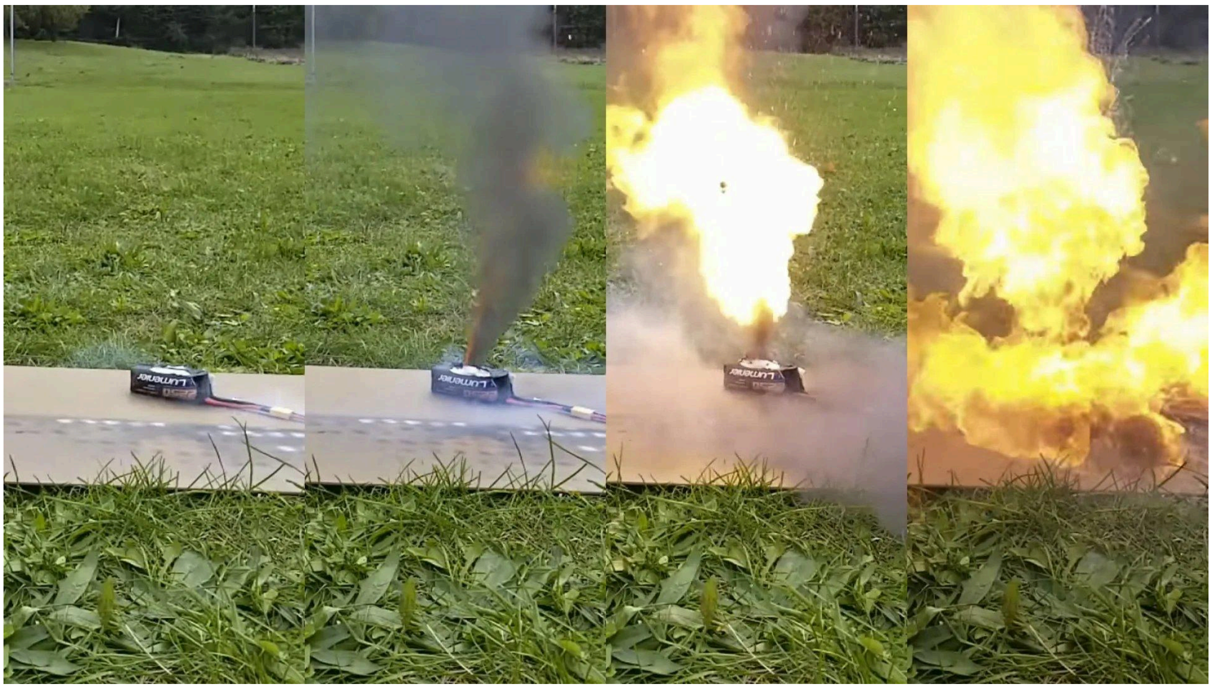


In connection with the latest events in Lebanon on September 17 and 18, 2024, I would like to address the question - how is this even possible? That gadgets explode?! There is always an answer to everything, especially in the exact sciences such as chemistry, physics and programming.

Media reports

On the 17th, according to media reports, about 5,000 pagers were blown up at one point in Lebanon. And on the 18th, there were reports of explosions and spontaneous combustion of other devices, such as smartphones, walkie-talkies, and stations with solar panels.

Yes, the lithium battery itself is quite dangerous under certain conditions. But with such force, as was seen in the footage on the network, it certainly does not explode, especially such a small battery - like on a pager. Even if the battery on the device is damaged, in the worst case it will burn out very quickly, perhaps with a small pop, but certainly not explode as footage from the scene showed us. This can be easily checked on Utrupe by searching “How does a lithium battery explode.”



Here in the photo is a battery fire, I want to note that the battery is quite large, but at the same time it did not explode like a grenade, but burned out very quickly!

Let's assume that a batch of 5,000 pagers were modified in advance. This is indicated by the force of the explosion. Most likely, in the typical sense, a bomb was not planted there - since this would have been quickly discovered even during the purchase of equipment. My opinion on this matter is the following:-

The battery consists of an anode and a cathode containing lithium. This element oxidizes fairly quickly and burns well. Between the anode and cathode there is an electrolyte. Previously, these were organic solvents, now they have been replaced by polymers and batteries have become “Lithium polymer”. These polymers are similar to plastic and also burn well. But they don't explode like a bomb! Apparently, the special services produced special batteries in which part of the polymer was replaced with an explosive. At the same time, the functionality of the gadget remains intact. And when heated to a certain temperature, an explosion occurs. You can still believe this.

But the next day (18th) there were media reports that other gadgets, such as telephones, walkie-talkies, and solar panels, began to explode. However, there is no video evidence. I don’t consider a couple of videos where a phone battery burns out in a workshop as evidence. Such videos are not uncommon. People bring their phones for repair, some of them have swollen batteries - out of order. And such batteries periodically ignite on their own all over the planet. And videos of burnt-out cars are not at all a fact that they were caused by phones.

Also in the case of pagers, it can be assumed that a remote signal was sent to detonate. But on the 18th they reported both walkie-talkies and solar panels. After all, you definitely cannot send a signal to a walkie-talkie or solar panel via the Internet or cellular network to activate hidden capabilities. Therefore, the events of the 18th are either an invention of people who decided to disrupt the hype on the topic. Or a ridiculous cover for Hezbolah’s failure.

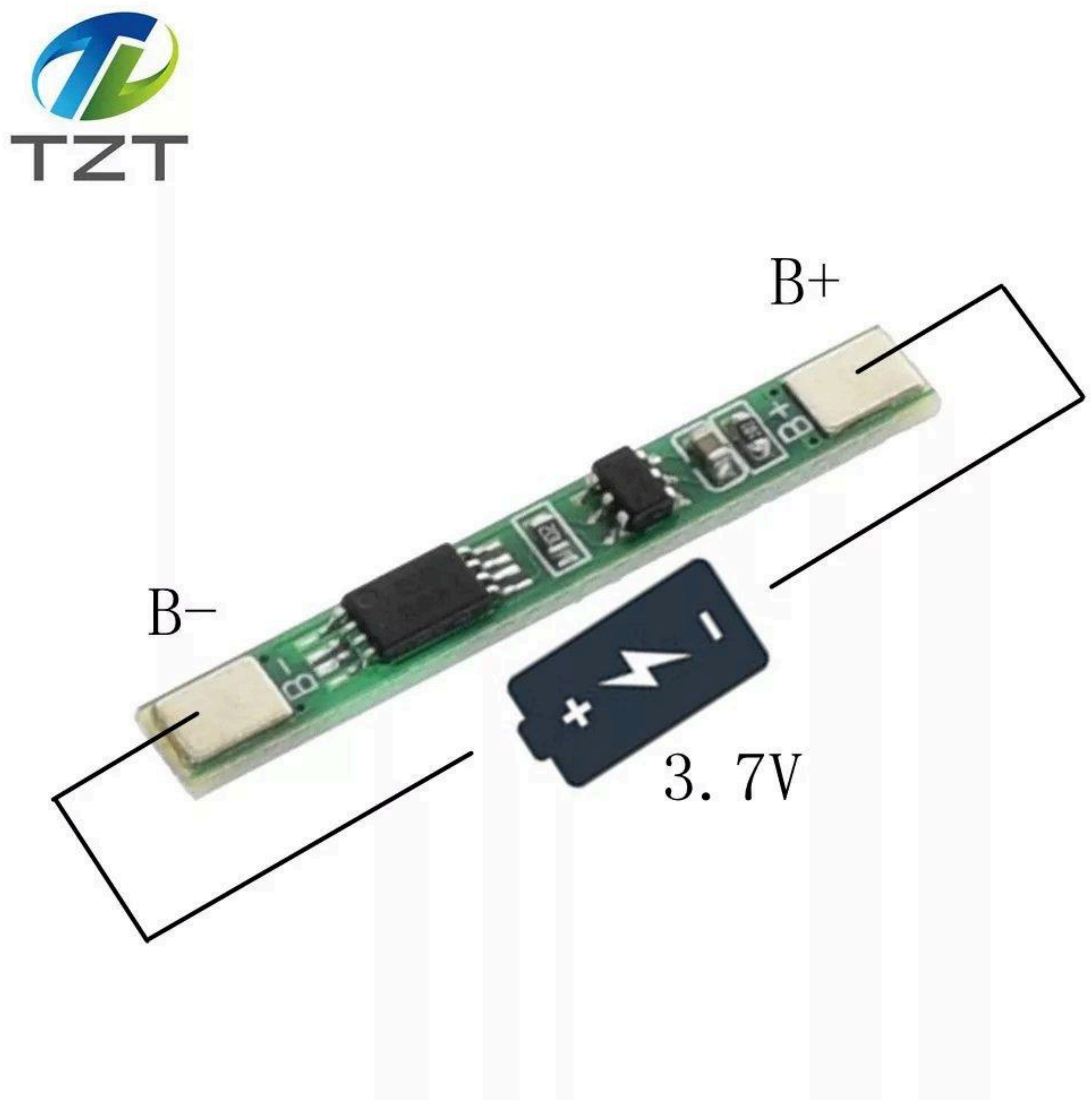
As a result, we have a batch of modified pagers that were actually bombs and were activated remotely and a bunch of fakes on the 18th.

What is a Li battery and how does it work?

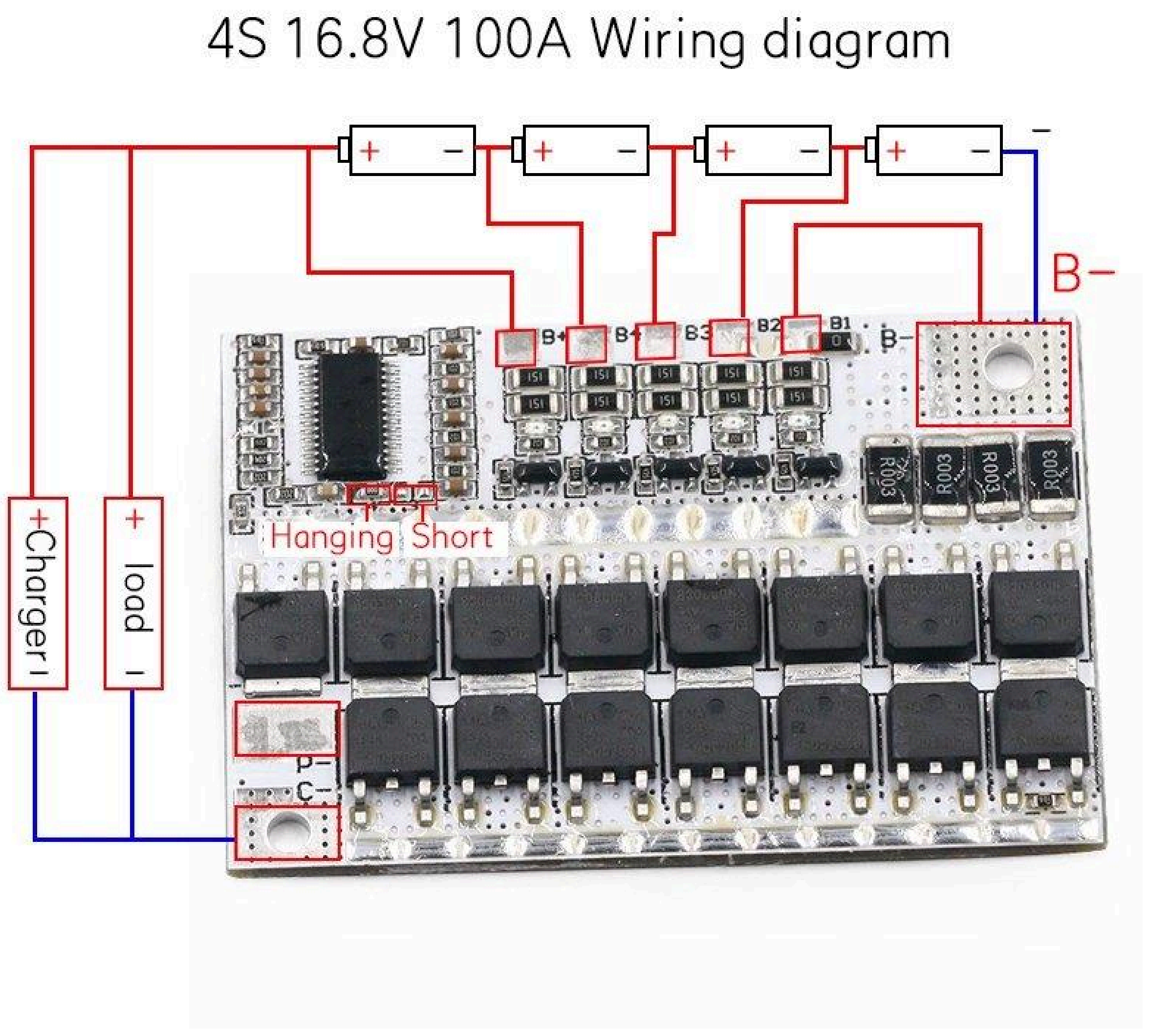
But now the analysis of the question begins - is it even possible to blow up a smartphone remotely?

A lithium battery is a dangerous thing under certain conditions. But in real life, you and I don’t see smartphones or other equipment with lithium batteries breaking down so often with such consequences.

The thing is that on any battery there is a small board called BMC. This board controls the charge and discharge voltage, as well as the current. It may have additional capabilities for temperature control, balancing of cans, recharging cans with low current and other functions. There are even so-called "Smart BMCs" that have the ability to transmit information via Bluetooth or the Internet, for example, to a smartphone. Such things are installed, for example, on electric bicycles to monitor the battery.



Example BMC for one battery.

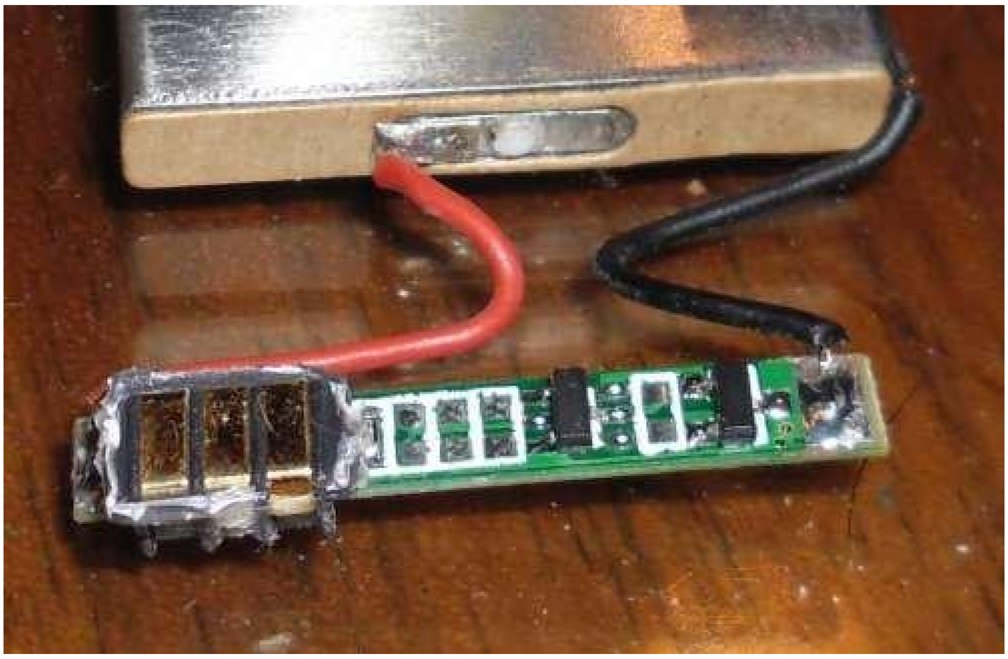


An example of a BMC for self-assembly of a battery from Li cells.

BMC is essentially a fuse located on the battery. Which does not allow the battery to fail! This is why we do not see constant spontaneous combustion of equipment with lithium batteries.

In theory, this should have been the end of the article. But a spherical hacker cannot sleep peacefully at night in a vacuum... And he is very interested in finding more information. We will do reverse engineering.

The beginning of a real investigation



BMC smartphone battery.

The BMC of a smartphone battery is a very simple device, without firmware or the possibility of external control/influence. In case of emergency situations, excess current / voltage, the BMC simply breaks the contact between the battery and the device. Thus protecting the device. BMC is a safety switch at its core.

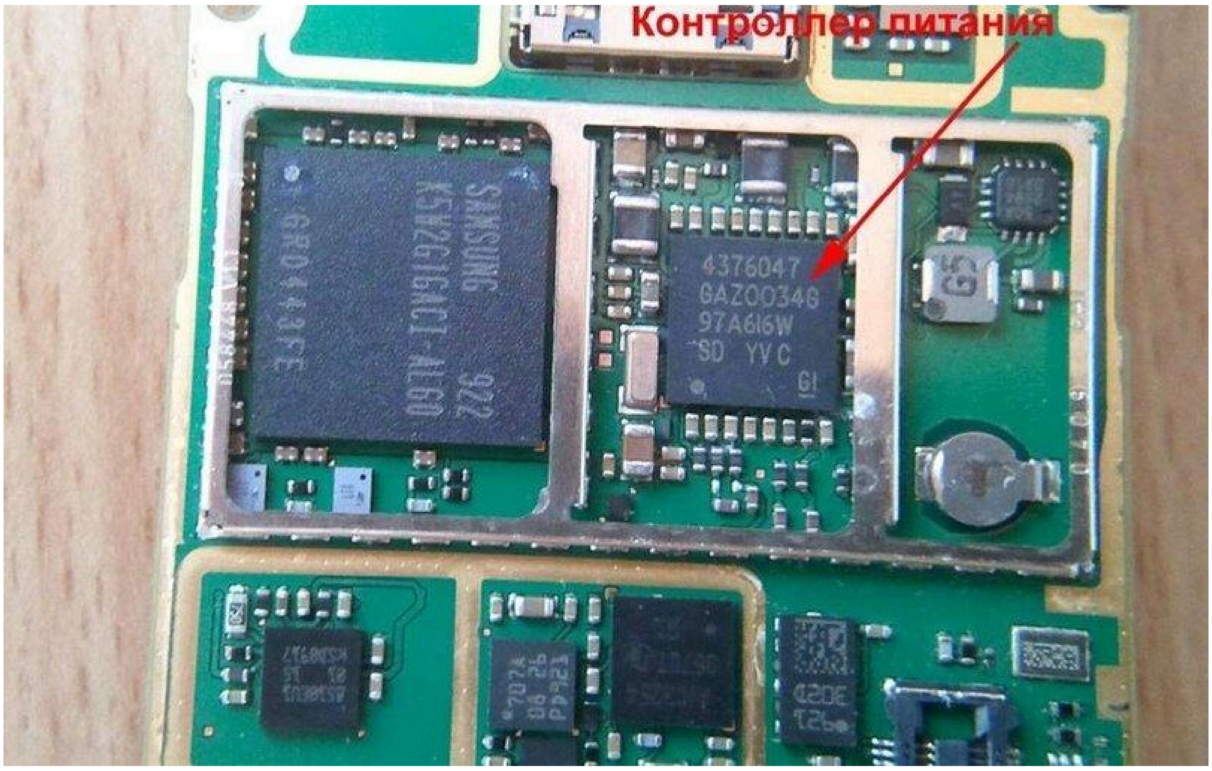
If anyone remembers, before, when batteries were removed from smartphones, they had 3 or 4 contacts. The purpose of the two contacts is completely clear. This is a plus and a minus. Why do you need more contacts?

The third contact is a temperature sensor. As a rule, an ordinary thermistor (a resistor that changes its resistance depending on temperature) is soldered between the contact (-) and an additional third contact. Or between two additional contacts.

Using this sensor, the smartphone understands that the battery is overheated and it is worth stopping its use (Safety shutdown of the device) or reducing the load on it.

Thanks to this, modern smartphones with a fast charging function understand how quickly the battery can be charged - so that it does not fail.

BMC is the battery emergency fuse. But not only the BMC board, but the so-called “Power Controller” is responsible for the operation of the battery in a smartphone. At its core, it is a small separate chip on the board that is responsible for powering the entire gadget.



Briefly, its main functions are as follows:

- Battery status monitoring.
- Providing protection for the battery from overcharging, deep charging, and various types of damage.
- Optimizing battery performance.
- Sending information about the status and operation of the battery to the phone.
- Battery temperature monitoring.

Why do you need a power controller in your phone?

Thanks to the power controller, smartphone users do not need to monitor the charging process, since this chip completely controls it. First of all, it regulates the required voltage so that it does not exceed its maximum or minimum value (overcharging or deep discharging does not occur).

In addition, the controller regulates the voltage and changes it during charging itself, depending on how discharged or charged the battery is. For example, the process of charging a phone to 80% is much faster than from 80% to 100%.

But the most useful function of the controller during charging is the ability to leave the phone with the charger connected after the charge level has reached 100%. The owner does not need to worry about the phone overheating or the battery exploding. He can leave the device charging overnight as the controller automatically reduces the voltage and activates the recharging phase.

Temperature control

Charging at temperatures that are too high (above 45 degrees) or too low (below 0 degrees) not only reduces battery performance, but is also potentially harmful to battery health. Most controllers have a function for monitoring battery temperature, the task of which is to send data about it to the phone. That, in turn, will either display a corresponding warning on the screen or turn off to prevent damage to the phone and battery.

First guesses

Let's think about how you can overheat a phone battery so that it spontaneously ignites. As we found out earlier, there is a BMC fuse on the battery itself. It is responsible for protection against abnormal voltage and current. But the BMC board itself cannot do anything about the temperature sensor.

Suppose - if you somehow programmatically start influencing the gadget so that it starts to overheat the battery, without exceeding the specified load according to the BMC - in theory it is possible, since the BMC is not responsible for the temperature, but only measures it. This prompts the idea that in this case you still need to influence the smartphone's charge controller in order to do this.

Let's look at the charge controller in detail. For example, I took the BQ25968 battery charge controller for Xiaomi Poco X3 NFC, Redmi Note 10T 5G.

Link to datasheet - https://www.alldatasheetru.com/datasheet-pdf/view/1286966/TI/BQ25968.html

As you can judge from the physical dimensions of the chip, there is a very large circuit inside.

9.3 Feature Description

9.3.1 Charging System

BQ25960 is a single-cell high efficiency switched cap charger, used in parallel with a switching mode charger. A host must set up the protections and alarms on BQ25960 prior to enabling the BQ25960. The host must monitor the alarms generated by BQ25960 and communicate with the smart adapter to control the current delivered to the charger.

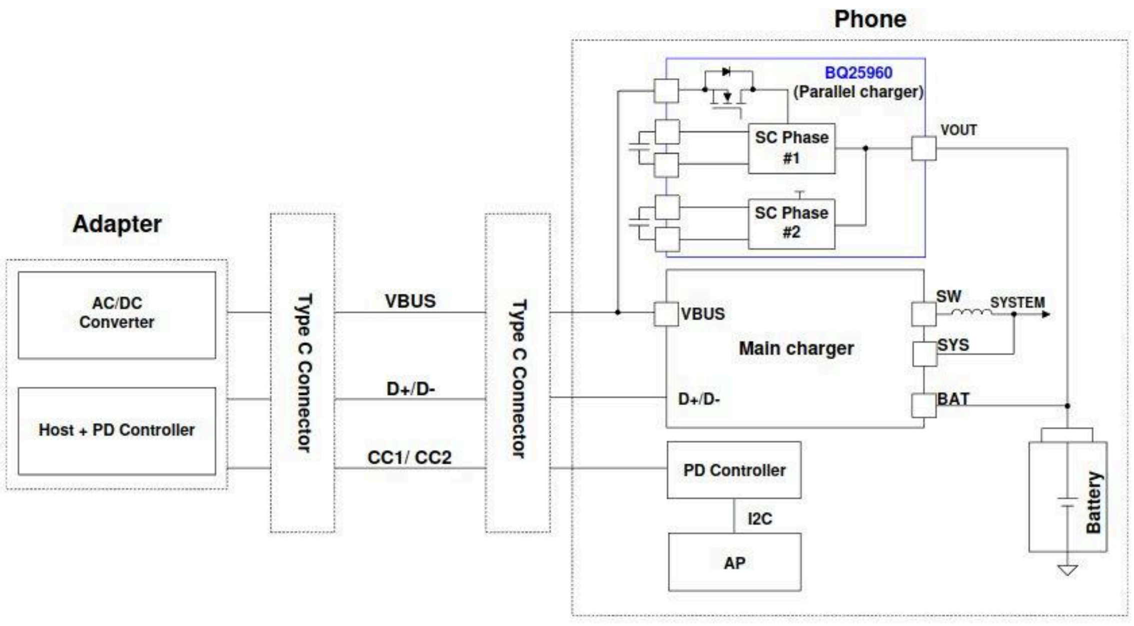


Figure 9-1. BQ25960 System Diagram

General connection

TEXAS INSTRUMENTS
www.ti.com

BQ25960
SLUSE08 – FEBRUARY 2021

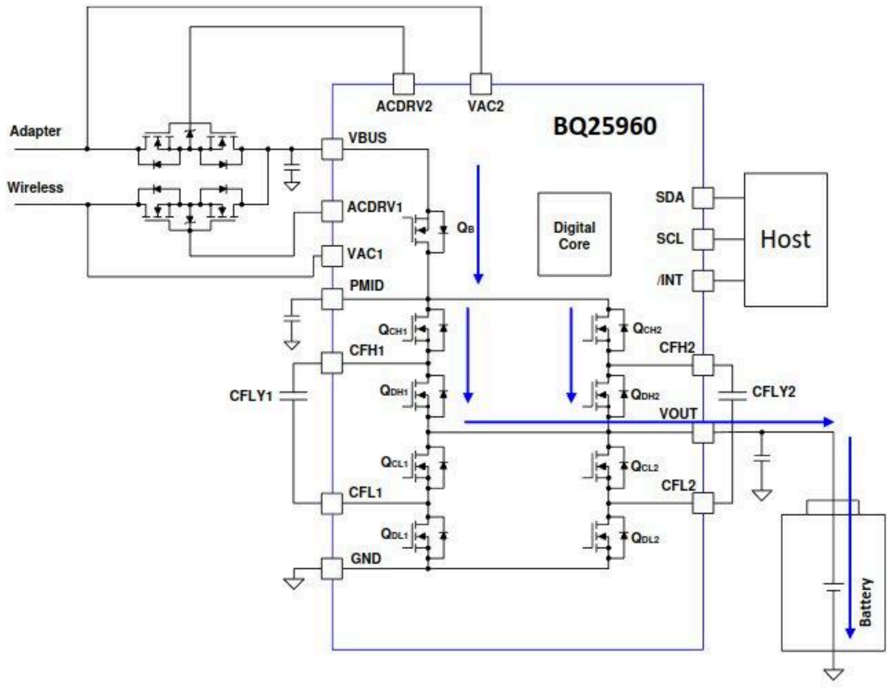
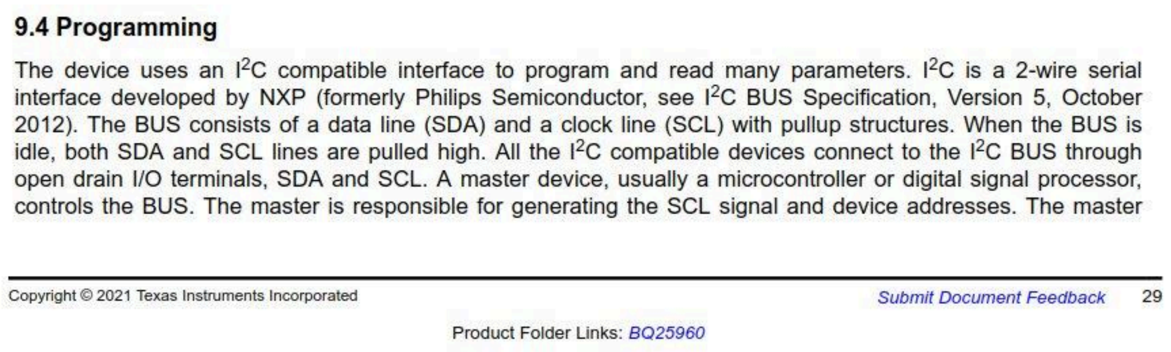


Figure 9-7. BQ25960 Bypass Mode

Internal general arrangement

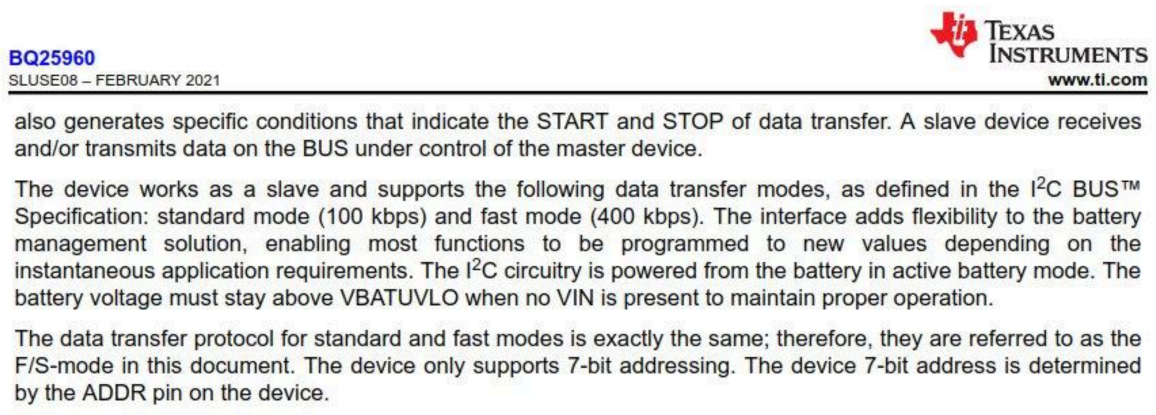
The fun begins next, when considering the full functionality. Fortunately, the datasheets write about this in black and white.



9.4 Programming

The device uses an I^{2}C compatible interface to program and read many parameters. I^{2}C is a 2-wire serial interface developed by NXP (formerly Philips Semiconductor, see I^{2}C BUS Specification, Version 5, October 2012). The BUS consists of a data line (SDA) and a clock line (SCL) with pullup structures. When the BUS is idle, both SDA and SCL lines are pulled high. All the I^{2}C compatible devices connect to the I^{2}C BUS through open drain I/O terminals, SDA and SCL. A master device, usually a microcontroller or digital signal processor, controls the BUS. The master is responsible for generating the SCL signal and device addresses. The master

Copyright © 2021 Texas Instruments Incorporated



also generates specific conditions that indicate the START and STOP of data transfer. A slave device receives and/or transmits data on the BUS under control of the master device.

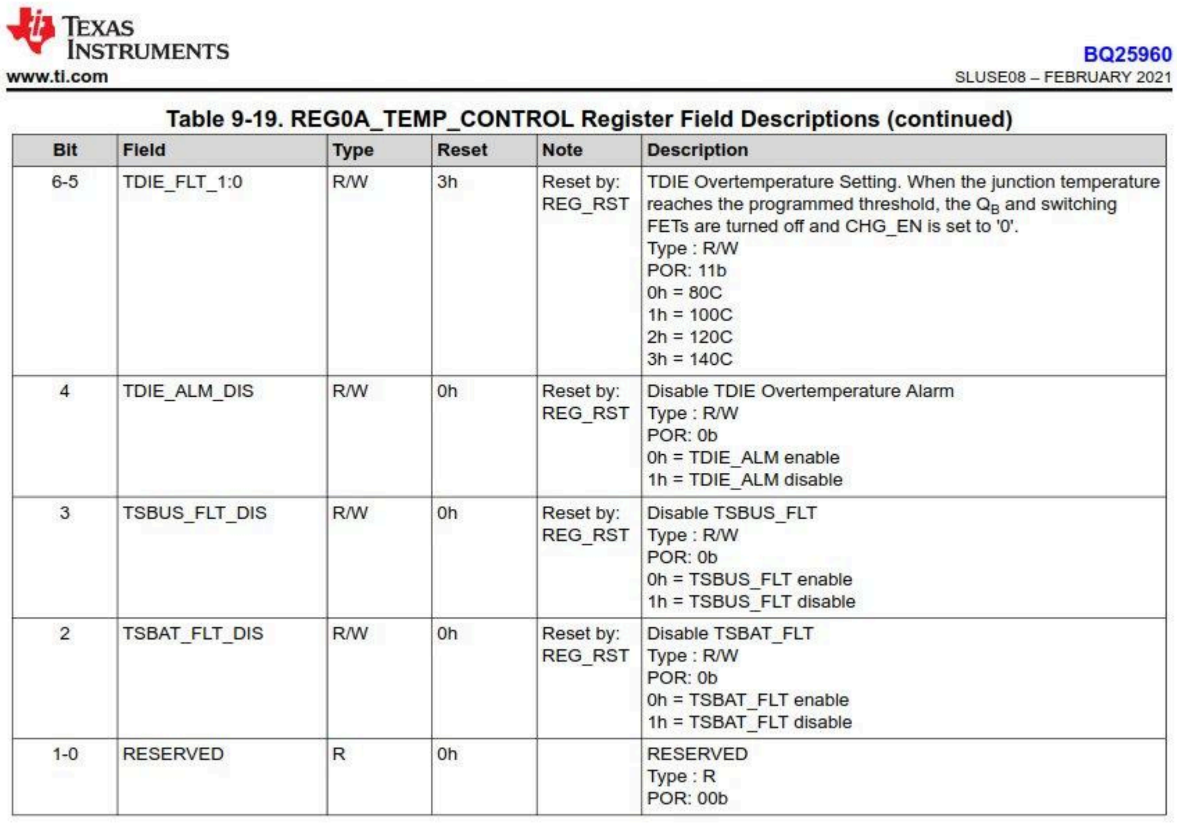
The device works as a slave and supports the following data transfer modes, as defined in the I^{2}C BUS™ Specification: standard mode (100 kbps) and fast mode (400 kbps). The interface adds flexibility to the battery management solution, enabling most functions to be programmed to new values depending on the instantaneous application requirements. The I^{2}C circuitry is powered from the battery in active battery mode. The battery voltage must stay above VBATUVLO when no VIN is present to maintain proper operation.

The data transfer protocol for standard and fast modes is exactly the same; therefore, they are referred to as the F/S-mode in this document. The device only supports 7-bit addressing. The device 7-bit address is determined by the ADDR pin on the device.

Translation:- The device uses an I2C compatible interface to program and read many parameters. I2C is a two-wire serial interface developed by NXP (formerly Philips Semiconductor, see I2C Bus Specification Version 5, October 2012). The BUS consists of a data line (SDA) and a clock line (SCL) with pull-up structures. When BUS is to the side, the SDA and SCL lines rise high. All I2C compatible devices are connected to the I2C BUS through open drain I/O terminals, SDA and SCL. The host device, usually a microcontroller or digital signal processor, controls the BUS. The master is responsible for generating the SCL signal and device addresses. The BQ25960 also generates specific conditions that indicate START and STOP of data transmission. The slave device receives and/or transmits data on the BUS under the control of the master device. The device operates as a slave and supports the following data transfer modes as defined in the I2C BUS™Specification: standard mode (100 kbit/s) and fast mode (400 kbit/s With). The interface adds flexibility to the battery management solution, allowing most functions to be programmed to new values depending on the immediate application requirements. The I2C circuit is powered by the battery in active battery mode. Battery voltage must remain above VBATUVLO when VIN is missing to maintain proper operation. The data transfer protocol for standard and fast modes is exactly the same;

therefore, they are referred to as F/S mode in this document. The device only supports 7-bit addressing. The 7-bit device address is determined by the ADDR pin on the device.

And then we look at the registers themselves:-


TEXAS INSTRUMENTS

www.ti.com

BQ25960

SLUSE08 – FEBRUARY 2021

Table 9-19. REG0A_TEMP_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Note	Description
6-5	TDIE_FLT_1:0	R/W	3h	Reset by: REG_RST	TDIE Overtemperature Setting. When the junction temperature reaches the programmed threshold, the Q_B and switching FETs are turned off and CHG_EN is set to '0'. Type : R/W POR: 11b 0h = 80C 1h = 100C 2h = 120C 3h = 140C
4	TDIE_ALM_DIS	R/W	0h	Reset by: REG_RST	Disable TDIE Overtemperature Alarm Type : R/W POR: 0b 0h = TDIE_ALM enable 1h = TDIE_ALM disable
3	TSBUS_FLT_DIS	R/W	0h	Reset by: REG_RST	Disable TSBUS_FLT Type : R/W POR: 0b 0h = TSBUS_FLT enable 1h = TSBUS_FLT disable
2	TSBAT_FLT_DIS	R/W	0h	Reset by: REG_RST	Disable TSBAT_FLT Type : R/W POR: 0b 0h = TSBAT_FLT enable 1h = TSBAT_FLT disable
1-0	RESERVED	R	0h		RESERVED Type : R POR: 00b

We see that there is a register responsible for temperature control. And it has R/W - this means that it can be rewritten. Valid value including 1h = Disable. This means you can programmatically turn off battery temperature control.

It turns out that it is still possible to bypass the protection of the BMC board at the hardware level. Let me remind you that the BMC on a cell phone battery only protects against abnormal voltages and currents. But he doesn't monitor the temperature! In theory, this is monitored by the battery charge controller, but it can be reprogrammed and the protection can be turned off!

Software part of the question

Now, in order to find out how deep the rabbit hole is, we will need to unearth the software features of the Android OS.

Access rights

And this is where the real “Sedition” begins. When studying Unix-like operating systems in more detail, which includes Android, you first need to understand how the file system is mounted and what access rights are.

In Android, when the device is turned off, there is no single file system on its flash drive as in the usual Windows. Instead, you will see a bunch of RAW images, each of which carries part of the overall file system. When you turn on the gadget, these images are mounted to the root point called Root. It will already contain all the kernel files, the OS itself and all your data in the Data folder.

What users call their OS is actually just a folder ./Data/ It contains all your data, programs, settings, phone book, etc. And what users call “Phone Internal Memory” is actually just one directory at ./Data/Media/0

For example, when you do a full factory reset on your phone, it actually deletes the Data folder.

So, you, as a regular phone user, do not have the right to go beyond the Data folder. Well, no way at all! All your files and programs can only be accessed within the internal contents of the Data folder.

Root access

The words “Root access” refer to special rights for a regular user, which allow him to go beyond the Data folder and interact with other root folders. This is where the name Root comes from - translated as "Root". Root access means that the user can go to the root directory.

Root access itself can be obtained by modifying the phone software. In my profile there are instructions on how to get Root access to Bastyon Mobi. But not all manufacturers provide this opportunity.

And thanks to Root access, I can view all the files on the phone, which is what I’m writing about in this article. Personally, I have been using phones only with Root access for more than 10 years, so I have an idea of how it all works.

But at its core, the OS itself has Root access initially. That is, programs from the ./System folder can completely control the phone.

Search for configuration files

What I love about Android is its openness! Even proprietary software on Android is not at all difficult to read without special programming knowledge. The thing is that this is a Unix-like system, which means the “Everything is a file” rule works. This means that the entire OS consists of text files. But already assembled code is not always text. However, it can still be "Parsed" even on the fly and viewed. The funny thing is that no one is hiding anything inside Android! Everything is written openly! And to use and understand all this, you don’t need to have an academic degree. The Enthusiast level of knowledge is enough.

To do this, I will use my favorite program from Chinese developers MTmanager. By the way, this is the only program that I bought in my life! I paid as much as 20 bucks for a lifetime license. Although almost all functions of the program are available for free. When you first open this program, you might think that this is an ordinary two-window file root manager. But this program can decompile any application, make changes to it and put it back together! Android Studio right in your pocket.

Let's start our search with the simplest. Many enthusiasts know that you can calibrate the battery by deleting the batterystats.bin file located in ./Data/System. This file records the charge and discharge cycles of the battery. Based on this data, Android OS predicts the state of the battery and how long the charge will last. But there are times when incorrect data is written to this file and because of this, the OS turns off the phone ahead of time, because it thinks that the battery is low, or vice versa. The battery is completely drained, but the OS thinks that there is still 20% left. To avoid such situations, you can delete this file, reboot the phone, the system will create it again and start statistics from scratch. Which in theory could help. But this is not exactly what we are looking for.

Further digging through the phone files, in ./System/Flamework/Flamework-res.apk/res/xml - you can find records about battery capacity and processor clock speeds and other settings and data.

And in ./Vendor/Etc you can find a bunch more processor configuration files. By the way, I remember once bloggers raised a hype that some phones, when running “Benchmarkers”, deliberately increase clock frequencies, so, in such phones in this folder you can find “WhiteList” files in which this is written and you can read it all yourself!

It becomes clear that the Android OS has configuration files responsible for setting up the battery, processor and more... In theory, you can make it so that the battery is overheated by an abnormal load and causes a fire.

But everything is not so simple here, because Android 13+ has protection against changing system files. If you try to change at least one file in ./System or ./Vendor, the phone will no longer turn on and will go into BootLoop.

Three Essences of Control

Now we need to understand who can gain access and how.

Let us remember what was written earlier - that the user and all his programs cannot interact with system objects! This means that not one program, not one virus - which the user picks up - cannot do anything with the system. So you need to look in the system itself.

The first entity of phone control is the Android OS itself. Pure Android AOSP itself is a good clean OS. It has few functions, but also no software bookmarks. And this can be checked in the source code, because AOSP is Open Source.

The second entity of control is the phone manufacturers themselves. Such as Samsung, Asus, Xiaomi and others. They take AOSP and overhaul it. They are introducing a huge number of changes and new features. And this is no longer open information. In addition, they add their own programs to the phone, which are unknown what they can do and are located in the system partitions. In fact, they have Root access initially.

The third control entity is Gapps. Google is not at all interested in distributing Android for free. Therefore, almost all phones have a pre-installed Google software package - Gapps. When you first turn on the phone, during its setup, you will be told that Google has the right to install programs at its discretion, and there will also be a point nearby - that even if Bluetooth is turned off, it will still work. And third, Google's license agreement is hundreds of pages long. And until you agree with all of this, you will not be able to turn on your smartphone! And it all works from the ./System folder.

Total

To summarize, we have a bunch of different software in the system folders of the factory phone, which does not know what it does and is closed source. "Hello" in the form of surveillance, remote control and undocumented functions may arrive.

Many users are afraid of viruses and other things. But it is important to understand that nothing happens just like that. Security systems in the modern world are very good. Yes, there are errors in any code that can be exploited by hackers or special services. But as a rule, in 99.9% of cases of hacking of any systems, the user himself is to blame. I downloaded and installed a program downloaded from an unknown site, or opened an archive with a virus, or went to an infected site.

But there are known cases of hacking when the user does not need to do anything at all! An example of such hacker software is "Pegasus" from Israel. But this is a very murky matter; I don’t believe that this is possible without the participation of Google itself. If this software is sold to intelligence agencies around the world, who is stopping Google itself from buying a copy through a front man and closing all the security holes?! This suggests that Google itself is a participant in this...

Or the story of how Israel used AI to carry out bombing attacks, and then it turned out that this AI took information from Google and was processed at Google data centers...

In fact, nothing happens on a regular factory phone without Google's knowledge. A striking example of this is the Sberbank application. When installing it, it will ask for two permissions, to access contacts and to access the internal memory. If access to contacts can be denied, then access to files is required - without this, the Sber application will not work. And when you give this permission, the Sber application will launch the “Antivirus” type every 15 minutes. Scan all your files and don’t know what to do with the scan results.

But when installing Sber on Mobi, you don’t have to give permission to access files, but the application will work. And the most interesting thing is that if you still give access to the files, the antivirus will not start. Miracles! Is it true?

This is an example of how Google gives some apps more access than others. And then we wonder how they listen to us, and then show advertising... And here we don’t need to equate everything with Android itself. Besides it, there are other “Control Entities” on the smartphone.

P.S.

A little about Bastyon Mobi. It is based on Lineage OS. Which in turn is modified by AOSP enthusiasts. It is open source and can be verified by any specialist.

Secondly, each user can get Root access to Mobi and personally view all the files on the phone.

Third, in the Mobi firmware there is only one “Essence of Control” - the OS itself. There are no closed-source programs on the phone, especially in the system sections.

In general, after September 17, you and I live in a new reality. Where there is almost no security and the virtual world intersects with the real one. Apparently BigTech corporations can do much more than I previously thought. And the only way out is to use “Open Software” on all your gadgets. Only this can protect you and your data in our turbulent times.