Maxim Integrated MAX31343SHLD# Real Time Clock - Review

created by Gough Lui on Jun 6, 2021 12:21 PM, modified on Jun 19, 2021 6:23 PM

RoadTest: Maxim Integrated MAX31343SHLD# Real Time Clock

Buy Now

Evaluation Type: Independent Products
Was everything in the box required?: Yes
Comparable Products/Other parts you considered: Other RTCs from Maxim, STMicroelectronics, Abracon, MicroCrystal and Seiko Epson Devices with integrated timebases.
What were the biggest problems encountered?: No significant problems encountered.

Detailed Review:

Scoring	
Product Performed to Expectations:	9
Specifications were sufficient to design with:	10
Demo Software was of good quality:	10
Product was easy to use:	10
Support materials were available:	10
The price to performance ratio was good:	10
TotalScore:	59 / 60

Maxim MAX31343 MEMS-based 5ppm RTC Shield Evaluation Kit RoadTest Review

By Gough Lui – May-June 2021

A Real-Time Clock (RTC) integrated circuit is commonly used with microcontrollers and CPUs to provide it with the current date, time, calendar day and other features such as alarms and occasionally RAM storage. Such peripherals are commonplace amongst electronic devices, with the majority of the RTCs on the market relying on an external 32.768kHz quartz crystal for its timebase with a few integrating the crystal in-package.

The MAX31343 is a departure from this norm, applying modern MEMS technology in an RTC claiming 5ppm accuracy. The use of MEMS is supposed to confer benefits such as a smaller size, low-power consumption, increased shock resistance, reduced ageing/drift and improved timing accuracy/stability. This particular IC features additional features such as an inbuilt configurable trickle charger, configurable low-voltage thresholds, a wide operating voltage (1.6V to 5.5V), temperature-compensated programmable square wave output, uncompensated programmable clock output, one interrupt, two alarms and 64-byte RAM storage.

Thanks to element14 and Maxim Integrated for choosing me to be a RoadTester of this MEMS-based RTC through their MAX31343SHLD evaluation kit. I hope you find this review to be informative, entertaining and useful. If you did, feel free to share this review, leave a like, bookmark, comment, question or rating, and thanks for reading!

EDIT - Apologies for issues with loading images - this is currently under investigation. It seems some users can load all images, others may have a few broken images.

Introduction to RTCs & Market Survey

A real-time clock (RTC) integrated circuit is not something that we often think about until we encounter a system that doesn't have any *sense of time*. This could include certain microcontrollers and single-board computers – including the Arduino and Raspberry Pi. In standalone applications without internet connectivity, it could be important to know the *calendar* day, date and time to be able to timestamp logs for troubleshooting or data logging to make them easier to interpret. It can also be important to *measure* timing intervals to ensure actions are performed periodically or at set times (e.g. like an alarm clock). Finally, it could even be useful as a memory to store configuration parameters where they can easily be reset – for example, the CMOS configuration in most PCs and laptops. After all, knowing the time is important for many reasons – from validating SSL certificate validity to speeding up GNSS satellite acquisition.

For many years, the RTC chip landscape had been dominated by the likes of Dallas Semiconductor whose heritage is widely visible in the venerable series of DSxxxx chips which are ubiquitous and still in use today. But what you might not know is that Dallas Semiconductor was acquired by Maxim Integrated in 2001 with the former branding being phased out by 2007. Many of these chips rely on tuning-fork style 32.768kHz quartz crystals for their time base, frequently external to the IC but also later integrated into package.

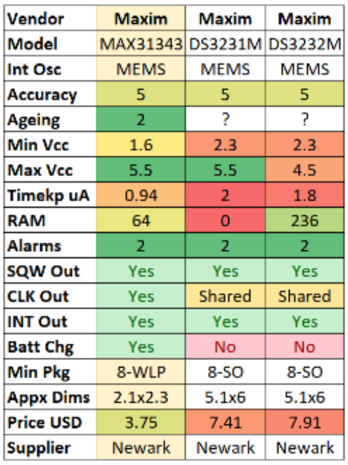
The main downsides of quartz crystal references include their physical size, mechanical fragility to shock, ageing or drift over time and energy consumption which sets a barrier to low-power operation. The use of quartz crystals is well established and their characteristics are generally well understood, including their temperature drift characteristics which depend on the geometry of the crystal. This has resulted in a few innovations in the market – the use of temperature-compensated crystal oscillators (TCXO) and digitally temperature-compensated crystal oscillators (DTCXO) which extend the operating temperature range by compensating for temperature drift in either an analog or digital way, maintaining tighter accuracies. To reduce power consumption, some RTCs instead use an elaborate arrangement of an R-C oscillator for their primary source of time (which would be very inaccurate on its own), disciplined to a quartz crystal oscillator which only runs intermittently (to correct for the R-C oscillator error). Others have ensured they can be used with crystals featuring higher ESRs to reduce power dissipation in the crystal.

The development of micro-electromechanical systems (MEMS) provides another potential solution by enabling the possibility of fabricating such oscillator systems at small scale in silicon, reducing costs and improving integration. Indeed, the existence of MEMS oscillators to provide higher-frequency system clocks is something I have *heard* of, as some high-end DACs use them for their superior stability, but also because the iPhone's MEMS oscillator had been observed to be affected by helium leaks from MRI machines. The MAX31343 is a RTC chip that uses similar MEMS technology to provide an inexpensive, high-accuracy real-time clock and low-frequency clock source in a compact all-integrated package with improved shock resistance and low power consumption.

To understand the relative strengths and weaknesses of the MAX31343, a market survey was undertaken on 28th May 2021 to compare different RTC solutions on the market. As a potential purchaser of the MAX31343, I focused only on RTCs that used an I^{2}C interface, that had an integrated time-base and is presently listed on either electronics component vendor sites or the manufacturer's website as being an in-production part that was recommended for new designs.

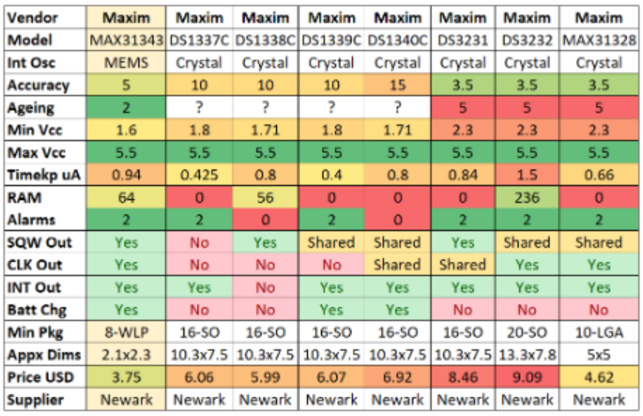
Unfortunately, because not all manufacturers specify their RTCs in the same way, there are **quirks** which need to be kept in mind when reading this market survey. Specifically, for a more compact presentation, abridged part numbers are used. Where a part is available in multiple packages, the smallest package with an integrated crystal is chosen. Some manufacturers do not provide any figures for ageing and other manufacturers provide one-year figures instead of ten-year figures. As a result, the ageing parameter may not be directly comparable. For accuracy, the worst figure is presented, however some manufacturers only provide typical values which makes comparisons a bit difficult. Some vendors do not clearly specify number of alarms and RAM capacity which may have been inferred from a quick glance through the I^{2}C registers. The operating voltage range is also specified differently, as some manufacturers claim low operating voltages which do not guarantee clock start-up.

Disclaimer: *The data compiled in this market survey is provided in good faith on a best-effort interpretation of data from manufacturer's datasheets. Differences in measurement conditions and methodologies may not be highlighted and no verification of on-paper specifications with real-world performance is made. Not all parameters and features are compared. Differences between vendor terminology may lead to errors or omissions. Prospective purchasers must perform their own independent research to assess the suitability of parts for their intended application. I cannot be held responsible for any errors, omissions or damages incurred (regardless of how they were incurred) from the use or inability to use the data tabulated below.*



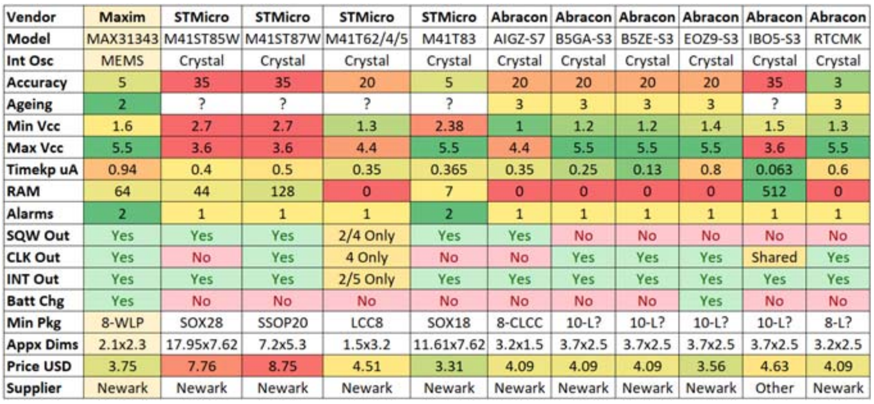
Vendor	Maxim	Maxim	Maxim
Model	MAX31343	DS3231M	DS3232M
Int Osc	MEMS	MEMS	MEMS
Accuracy	5	5	5
Ageing	2	?	?
Min Vcc	1.6	2.3	2.3
Max Vcc	5.5	5.5	4.5
Timekp uA	0.94	2	1.8
RAM	64	0	236
Alarms	2	2	2
SQW Out	Yes	Yes	Yes
CLK Out	Yes	Shared	Shared
INT Out	Yes	Yes	Yes
Batt Chg	Yes	No	No
Min Pkg	8-WLP	8-SO	8-SO
Appx Dims	2.1x2.3	5.1x6	5.1x6
Price USD	3.75	7.41	7.91
Supplier	Newark	Newark	Newark

If you're looking for a **MEMS-based RTC**, you don't have many options at this present time with **all** of them coming from Maxim Integrated. All three units claim 5ppm accuracy, but the MAX31343 is able to provide a wider operating voltage range, lower timekeeping current, CLK out, smaller package and significantly lower price. The DS3232M provides a larger RAM of 236 bytes compared to the 64 bytes of the MAX31343, however, this is unlikely to matter much in most applications. It makes the MAX31343 the clear stand-out amongst MEMS-based RTCs.



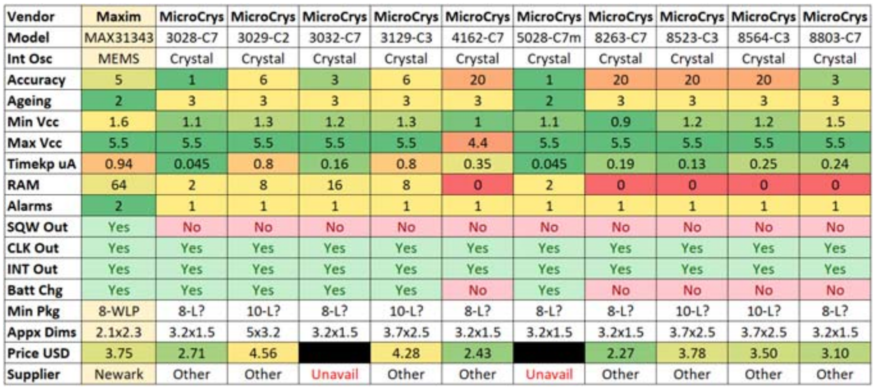
Vendor	Maxim	Maxim	Maxim	Maxim	Maxim	Maxim	Maxim	Maxim
Model	MAX31343	DS1337C	DS1338C	DS1339C	DS1340C	DS3231	DS3232	MAX31328
Int Osc	MEMS	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal
Accuracy	5	10	10	10	15	3.5	3.5	3.5
Ageing	2	?	?	?	?	5	5	5
Min Vcc	1.6	1.8	1.71	1.8	1.71	2.3	2.3	2.3
Max Vcc	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5
Timekp uA	0.94	0.425	0.8	0.4	0.8	0.84	1.5	0.66
RAM	64	0	56	0	0	0	236	0
Alarms	2	2	0	2	0	2	2	2
SQW Out	Yes	No	Yes	Shared	Shared	Yes	Shared	Shared
CLK Out	Yes	No	No	No	Shared	Shared	Yes	Yes
INT Out	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
Batt Chg	Yes	No	No	Yes	Yes	No	No	No
Min Pkg	8-WLP	16-SO	16-SO	16-SO	16-SO	16-SO	20-SO	10-LGA
Appx Dims	2.1x2.3	10.3x7.5	10.3x7.5	10.3x7.5	10.3x7.5	10.3x7.5	13.3x7.8	5x5
Price USD	3.75	6.06	5.99	6.07	6.92	8.46	9.09	4.62
Supplier	Newark	Newark	Newark	Newark	Newark	Newark	Newark	Newark

Looking at a broader selection of **Maxim Integrated RTCs**, it seems that there are newer crystal-based RTCs which offer improved initial accuracy of 3.5ppm but with a worse (ten-year) drift figure of 5ppm, compared to the 5ppm initial and 2ppm drift of the MEMS-based MAX31343. Many of the crystal RTCs are able to offer a lower timekeeping current consumption than the MAX31343, which is slightly surprising. The MAX31343 shows clear benefits in the wider operating voltage, larger RAM (except vs DS3232), more outputs, smaller package size and lower cost. On the whole, even when faced with a plethora of crystal-based RTC choices, the MEMS-based MAX31343 is very desirable.



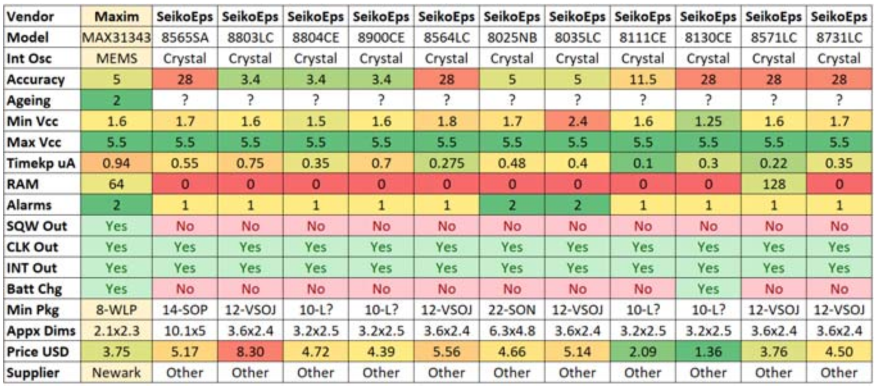
Vendor	Maxim	STMicro	STMicro	STMicro	STMicro	Abracon	Abracon	Abracon	Abracon	Abracon	Abracon
Model	MAX31343	M41ST85W	M41ST87W	M41T62/4/5	M41T83	AIGZ-S7	B5GA-S3	B5ZE-S3	EOZ9-S3	IBO5-S3	RTCMK
Int Osc	MEMS	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal
Accuracy	5	35	35	20	5	20	20	20	20	35	3
Ageing	2	?	?	?	?	3	3	3	3	?	3
Min Vcc	1.6	2.7	2.7	1.3	2.38	1	1.2	1.2	1.4	1.5	1.3
Max Vcc	5.5	3.6	3.6	4.4	5.5	4.4	5.5	5.5	5.5	3.6	5.5
Timekp uA	0.94	0.4	0.5	0.35	0.365	0.35	0.25	0.13	0.8	0.063	0.6
RAM	64	44	128	0	7	0	0	0	0	512	0
Alarms	2	1	1	1	2	1	1	1	1	1	1
SQW Out	Yes	Yes	Yes	2/4 Only	Yes	Yes	No	No	No	No	No
CLK Out	Yes	No	Yes	4 Only	No	No	Yes	Yes	Yes	Shared	Yes
INT Out	Yes	Yes	Yes	2/5 Only	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Batt Chg	Yes	No	No	No	No	No	No	No	Yes	No	No
Min Pkg	8-WLP	SOX28	SSOP20	LCC8	SOX18	8-CLCC	10-L?	10-L?	10-L?	10-L?	8-L?
Appx Dims	2.1x2.3	17.95x7.62	7.2x5.3	1.5x3.2	11.61x7.62	3.2x1.5	3.7x2.5	3.7x2.5	3.7x2.5	3.7x2.5	3.2x2.5
Price USD	3.75	7.76	8.75	4.51	3.31	4.09	4.09	4.09	3.56	4.63	4.09
Supplier	Newark	Newark	Newark	Newark	Newark	Newark	Newark	Newark	Newark	Other	Newark

Looking further out, lets compare the MAX31343 with **STMicroelectronics and Abracon offerings**. With regards to accuracy, only the Abracon RTCMK is able to beat the MAX31343, claiming 3ppm accuracy with 3ppm drift versus 5ppm and 2ppm respectively. The RTCMK also offers a wider operating voltage range, but loses the RAM, SQW out and battery charging capabilities and has a larger package. Across the board, **all** of these RTCs were able to offer lower timekeeping current, with a few being slightly smaller in footprint area. The MAX31343 still seems the preferable option, remaining unmatched for outputs and competitive on price when accuracy is concerned.



Vendor	Maxim	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys	MicroCrys
Model	MAX31343	3028-C7	3029-C2	3032-C7	3129-C3	4162-C7	5028-C7m	8263-C7	8523-C3	8564-C3	8803-C7
Int Osc	MEMS	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal
Accuracy	5	1	6	3	6	20	1	20	20	20	3
Ageing	2	3	3	3	3	3	2	3	3	3	3
Min Vcc	1.6	1.1	1.3	1.2	1.3	1	1.1	0.9	1.2	1.2	1.5
Max Vcc	5.5	5.5	5.5	5.5	5.5	4.4	5.5	5.5	5.5	5.5	5.5
Timekp uA	0.94	0.045	0.8	0.16	0.8	0.35	0.045	0.19	0.13	0.25	0.24
RAM	64	2	8	16	8	0	2	0	0	0	0
Alarms	2	1	1	1	1	1	1	1	1	1	1
SQW Out	Yes	No	No	No	No	No	No	No	No	No	No
CLK Out	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
INT Out	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Batt Chg	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No
Min Pkg	8-WLP	8-L?	10-L?	8-L?	10-L?	8-L?	8-L?	8-L?	10-L?	10-L?	8-L?
Appx Dims	2.1x2.3	3.2x1.5	5x3.2	3.2x1.5	3.7x2.5	3.2x1.5	3.2x1.5	3.2x1.5	3.7x2.5	3.7x2.5	3.2x1.5
Price USD	3.75	2.71	4.56		4.28	2.43		2.27	3.78	3.50	3.10
Supplier	Newark	Other	Other	Unavail	Other	Other	Unavail	Other	Other	Other	Other

Comparing with **MicroCrystal's offerings**, the MAX31343 has some stiff competition. Their best modules (RV-3028-C7, RV-5028-C7 Medical) are able to reach accuracy levels of 1ppm initial and 2-3ppm (one-year) drift with timekeeping currents as low as a twentieth of the MAX31343, a wider operating voltage, a lower price (3208-C7) and a slightly smaller area. They lose the SQW out feature and most of the RAM, but it makes the 3028-C7 a very attractive proposition. The other offerings either have higher prices, lower accuracy, fewer capabilities or larger footprints making them less attractive. In this case, I think the MAX31343 has some very stiff competition in the form of the RV-3028-C7 based on performance, but it must be remembered that the RV-3028-C7 is a traditional crystal-based RTC and not MEMS-based.



Vendor	Maxim	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps	SeikoEps
Model	MAX31343	8565SA	8803LC	8804CE	8900CE	8564LC	8025NB	8035LC	8111CE	8130CE	8571LC	8731LC
Int Osc	MEMS	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal	Crystal
Accuracy	5	28	3.4	3.4	3.4	28	5	5	11.5	28	28	28
Ageing	2	?	?	?	?	?	?	?	?	?	?	?
Min Vcc	1.6	1.7	1.6	1.5	1.6	1.8	1.7	2.4	1.6	1.25	1.6	1.7
Max Vcc	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5
Timekp uA	0.94	0.55	0.75	0.35	0.7	0.275	0.48	0.4	0.1	0.3	0.22	0.35
RAM	64	0	0	0	0	0	0	0	0	0	128	0
Alarms	2	1	1	1	1	1	2	2	1	1	1	1
SQW Out	Yes	No	No	No	No	No	No	No	No	No	No	No
CLK Out	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
INT Out	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Batt Chg	Yes	No	No	No	No	No	No	No	No	Yes	No	No
Min Pkg	8-WLP	14-SOP	12-VSOJ	10-L?	10-L?	12-VSOJ	22-SON	12-VSOJ	10-L?	10-L?	12-VSOJ	12-VSOJ
Appx Dims	2.1x2.3	10.1x5	3.6x2.4	3.2x2.5	3.2x2.5	3.6x2.4	6.3x4.8	3.6x2.4	3.2x2.5	3.2x2.5	3.6x2.4	3.6x2.4
Price USD	3.75	5.17	8.30	4.72	4.39	5.56	4.66	5.14	2.09	1.36	3.76	4.50
Supplier	Newark	Other	Other	Other	Other	Other	Other	Other	Other	Other	Other	Other

Finally, we round out the comparison with the final big-name RTC module manufacturer that meets our market survey criteria, **Seiko Epson Devices**. Across the board, there are only a few products that can meet or exceed the initial accuracy of the MAX31343, however, they are all more expensive, have larger footprints, no RAM and fewer capabilities. It should be noted that the MAX31343 does lose out across-the-board on timekeeping current, with every one of Seiko Epson's offerings besting them on this metric. However, all things considered, I'd say the MAX31343 is still comfortably more desirable.

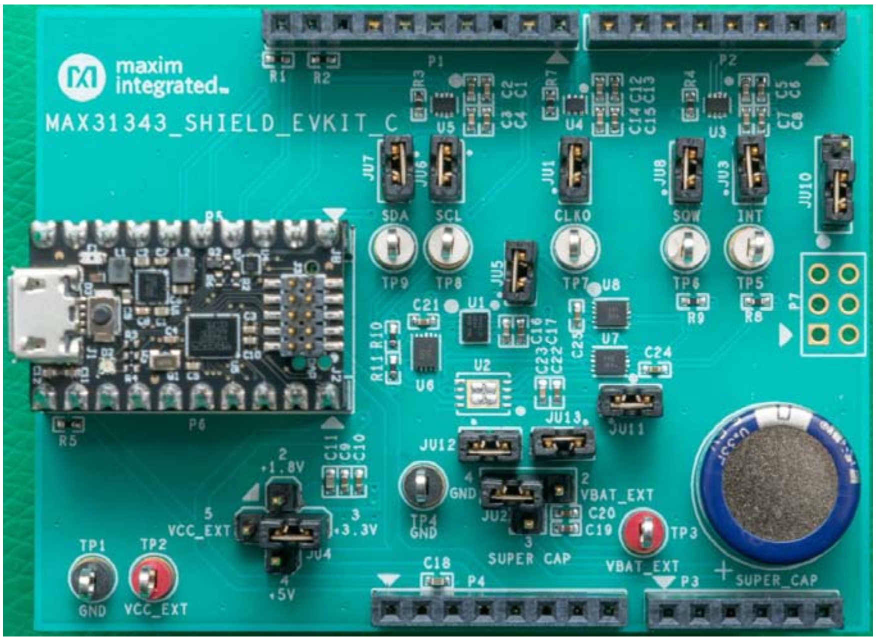
In conclusion, the MAX31343 occupies a rather unique space in the market, being one of very few MEMS-based RTCs available. While it has low power consumption compared to the prior MEMS solutions, its power consumption is not all that special when compared with the rest of the quartz crystal-based RTCs. However, it does offer a low cost and set of capabilities that is rarely matched or exceeded by most of the products on the market. The only exception would be MicroCrystal's RV-3028-C7 which seems to be more accurate, more power efficient, with a wider operating voltage, lower price and slightly smaller footprint area. However, in return, you do lose the SQW feature, most of the RAM and any of the physical robustness benefits that a MEMS solution would confer. All in all, the MAX31343 is an extremely desirable RTC chip.

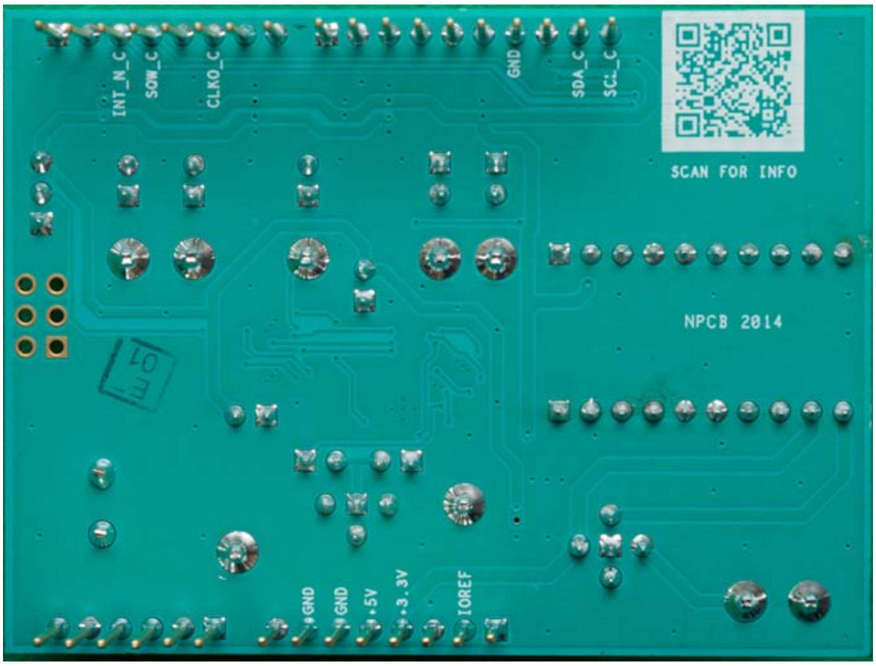
Unboxing

The MAX31343SHLD evaluation kit comes packaged in a brown cardboard box with a Maxim Integrated label affixed to the corner to identify its part number. The kit is currently listed at US$121.88 which seems a little expensive to me, but seems to be the norm for low-volume evaluation kits.

The box opens up along the seam on the rear like a set of double-doors to reveal the included items and a bit of pink anti-static foam providing some impact protection.

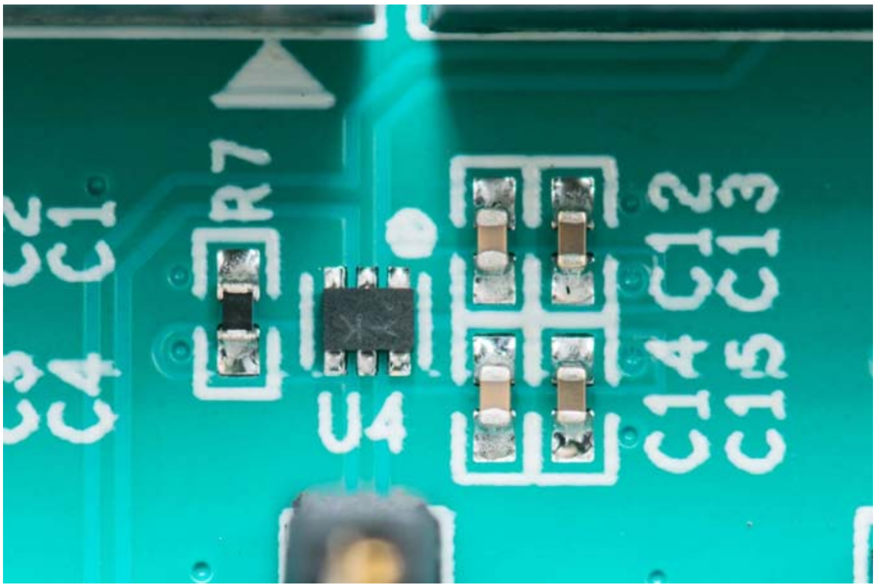
As with most Maxim evaluation kits, there is not much included – the resealable anti-static shielding bag contains the shield itself and a white 3ft USB-A to USB Micro-B cable to connect the kit to your computer.

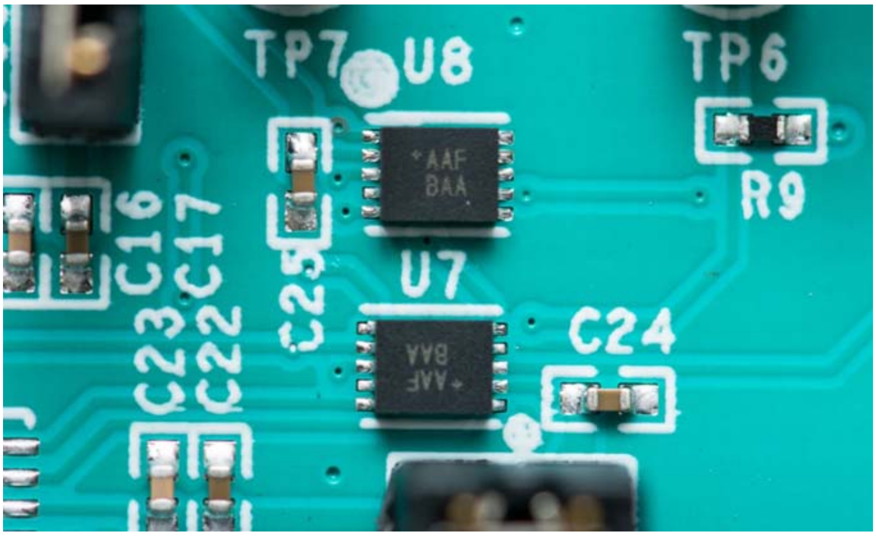




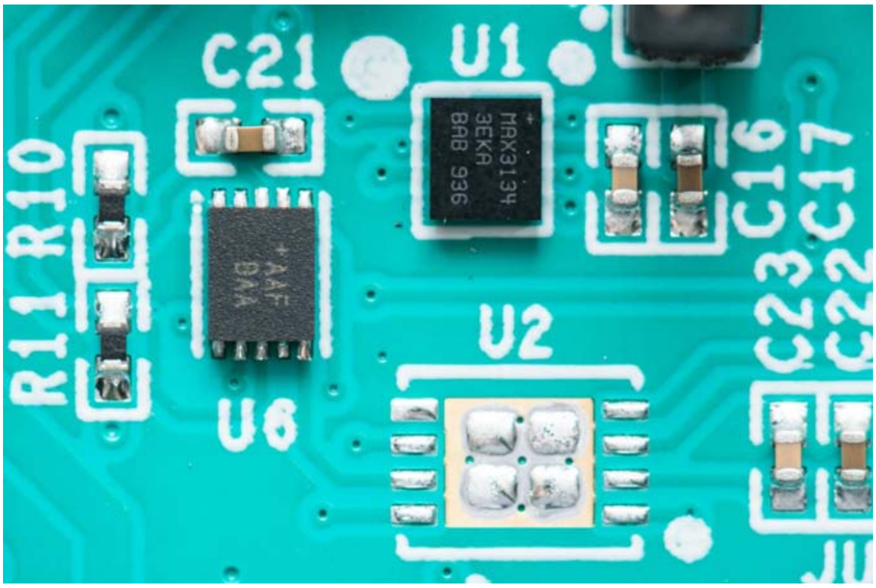
While I felt the kit to be a bit expensive, looking at the design of the board, the reasons begin to be apparent. The board appears to be a high-quality four-layer PCB with a Maxim-specific green solder resist and gold-plated finish that is very distinctive. The board has the form factor of an Arduino shield so that the MAX31343 RTC chip can be evaluated using Arduino-compatible platforms, but ships with a pre-mounted and pre-programmed MAX32625 PICO microcontroller in a “ready-to-go” configuration which certainly adds a little to the cost, in return for convenience.

The design of the board is an absolute delight to behold. It is expertly executed with plenty of jumpers allowing for isolation and re-routing of signals, selection of different Vcc and backup power sources. Colour-coded test point loops are also provided for monitoring of signals and supply of external Vcc and backup power. To accommodate the possibility of evaluating the MAX31343 in the larger TDFN-8 package, the U2 footprint is provided, but so are jumpers surrounding the chip allowing power to be routed to whichever footprint is used without affecting measurement integrity. Signal lines are routed through the provided MAX14689 DPDT analog switches (U6-U8) instead, which adds to the bill-of-materials.



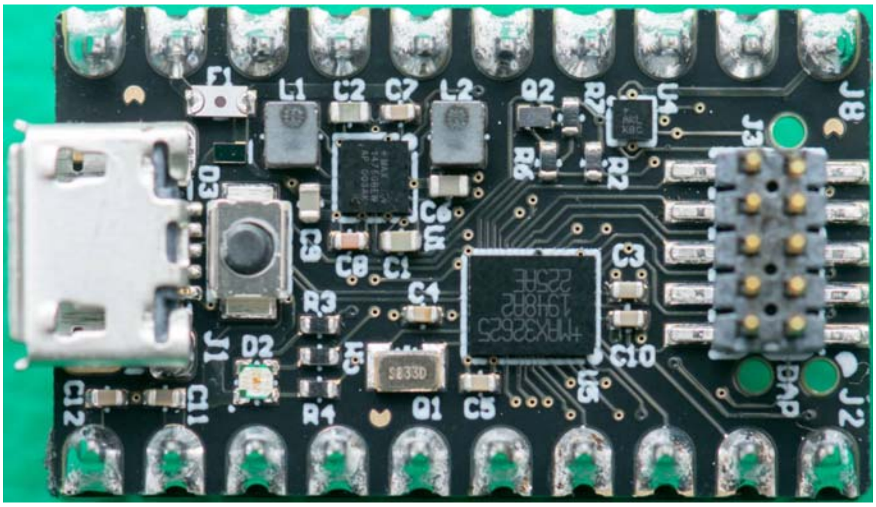


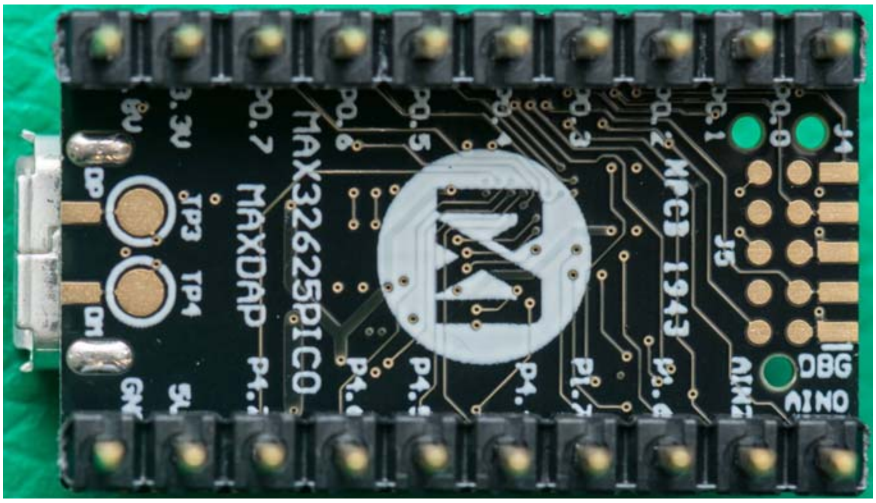
The RTC is also supported by an assortment of NLSX4401 and NLSX4373 level-shifters which handle all signals to and from the chip, allowing the Vcc selection to be completely decoupled from the external system's I/O voltage which is taken from the IOREF pin. The level of forethought in the design allows for easier evaluation of the RTC under different Vcc conditions even if you do not possess a host capable of communicating at the test Vcc.

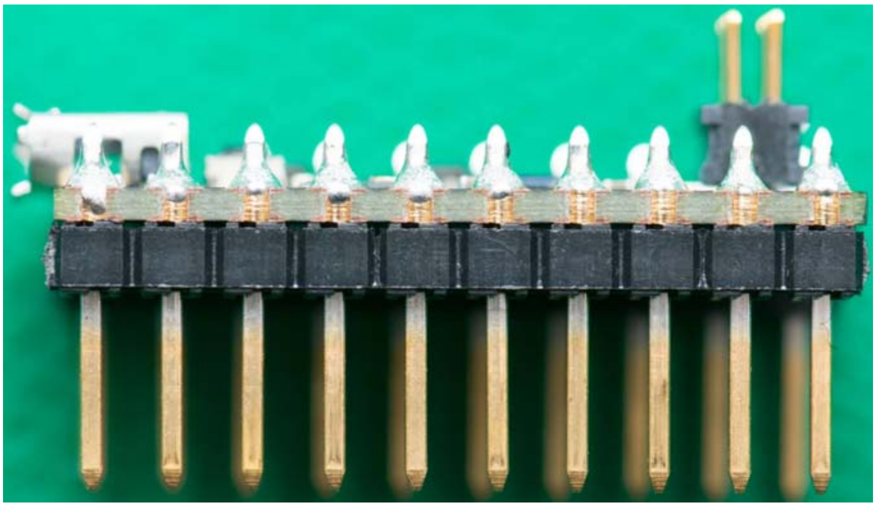


While the design is very much commendable for its flexibility and versatility for evaluating the RTC, it is unfortunate they did not also solder a second MAX31343 RTC chip onto the board (U2) in the TDFN-8 package given the price of the evaluation kit. It would also make great sense because the inclusion of analog switches (U6-U8) and some of the jumpers are essentially redundant until a second RTC chip is mounted. The chip itself would have only added a few dollars to the BOM cost, but would have allowed for both packages to be evaluated. As it stands, soldering in a TDFN-8 package could prove tricky as the surrounding components may be heat-damaged in a reflow attempt. Another notable note is that the use of the I2C SDA/SCL and IOREF pins on the Arduino shield footprint mean that an Arduino Uno R3 or newer footprint is necessary to be compatible with this shield (or creative rewiring is necessary). Older Atmega328P-based boards would need the pins bent out of the way and IOREF patched to 5V, with the SCL/SDA patched to A4/A5 pins to function.

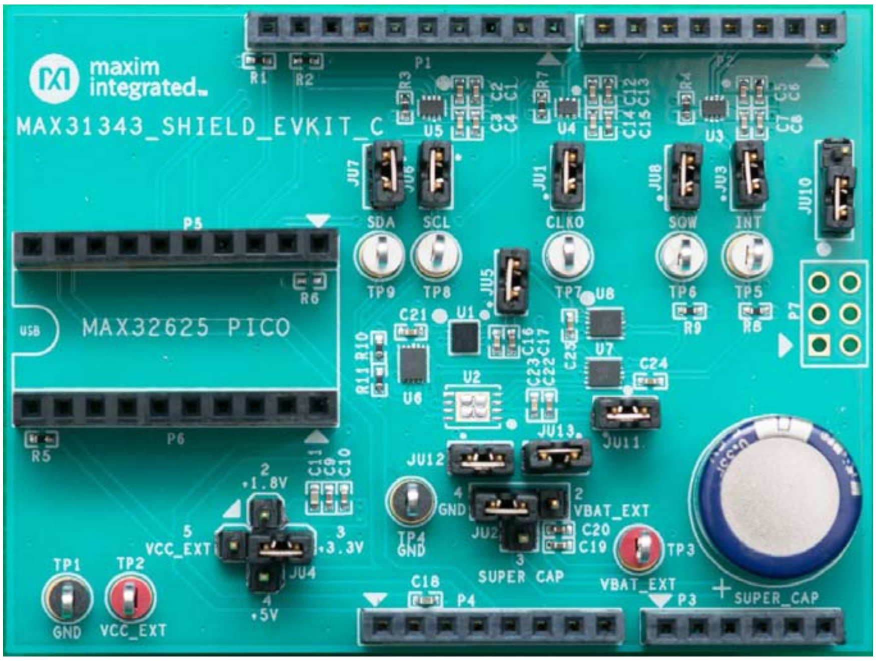
The underside of the shield is not wasted either with the pin connections identified and a QR code link printed in the silkscreen. The soldering, in general, was excellent with only minor flux residues visible on the headers for the MAX32625 PICO.

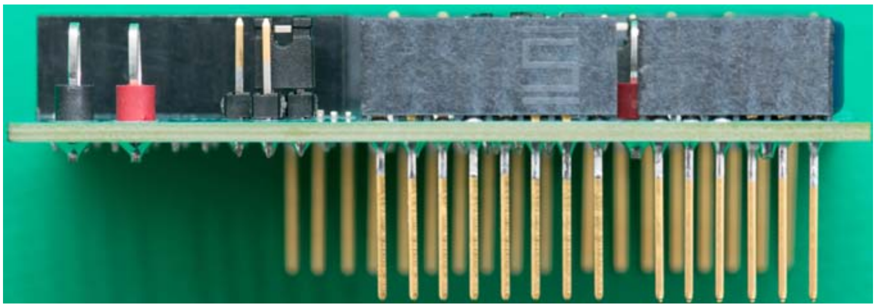




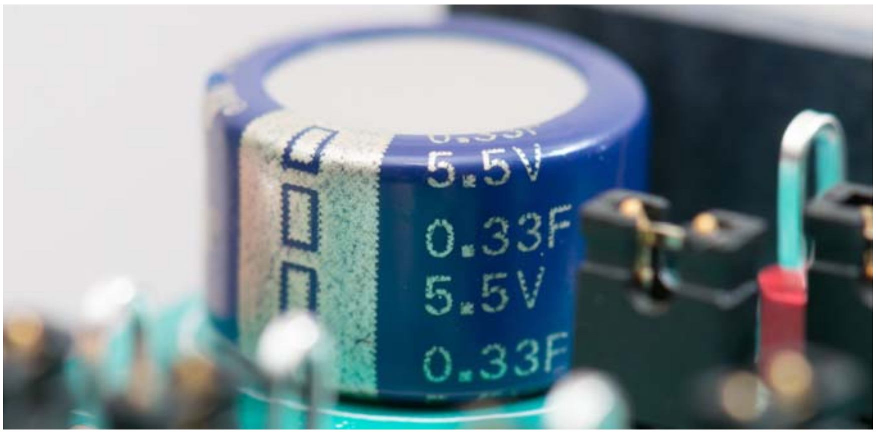


The MAX32625 PICO is on a very compact, densely packed black PCB. It's amazing to think there are two switching converters onboard to provide 3.3V and 1.8V in addition to 5V from USB. The microcontroller is a capable ARM Cortex-M4, with a DAPLink port shoehorned into a board that interfaces with two rows of standard ten SIL 2.54mm header pins.

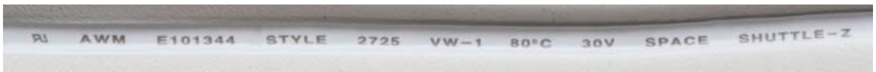




The shield comes pre-populated with stackable headers which allows for stacking additional shields on top. The profile of the components on the board are carefully chosen such that when the PICO is unmounted, they would not foul the connection of other shields. Quality components are also used in the construction – the Samtec header pins really show there has been attention to detail.



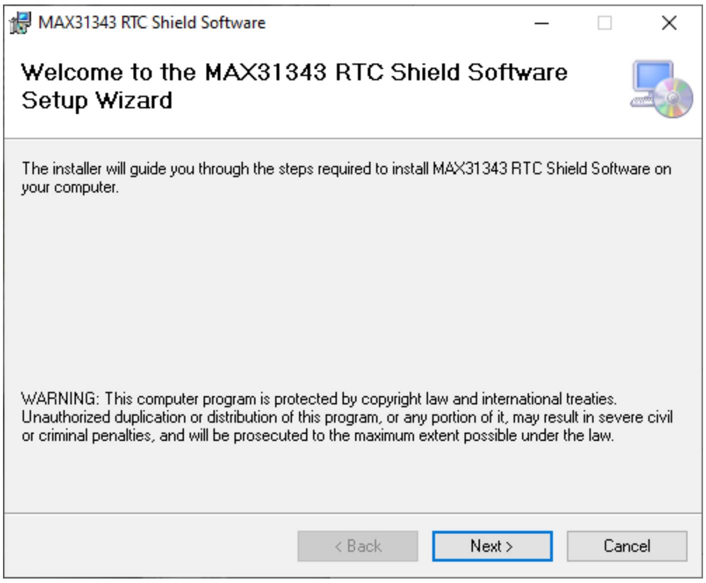
Whereas most RTCs on the market are evaluated with primary lithium cells or Ni-MH cells used as the backup power source, this board uses an Eaton 0.33F 5.5V supercapacitor instead. The use of a supercapacitor would have added cost to the BOM, but is required for systems which are expected to charge the backup power source from a Vcc rail as low as 1.8V. Conventional rechargeable batteries don't fit the operational voltage envelope well, so supercapacitors are a good alternative as it permits the evaluation of the onboard trickle charger functionality across the full operational voltage range.



As for the cable – there really isn't all that much to say about it as the cable does not identify its internal wire gauges. This is unlikely to pose a problem, as the evaluation kit is expected to use very little power overall. Documentation and other details are available from Maxim's website – not including printed material seems an environmentally friendly choice.

Initial Setup, Software & Documentation

Getting started is easy as the supplied board is pre-assembled and pre-configured to run. It should be as simple as connecting the USB cable between a computer running Windows 10 and the PICO microcontroller on the board, downloading the GUI software, installing it and running it.



Welcome to the MAX31343 RTC Shield Software Setup Wizard

The installer will guide you through the steps required to install MAX31343 RTC Shield Software on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

MAX31343 RTC SHIELD Software

MAX31343 RTC SHIELD Software , Version 1.0

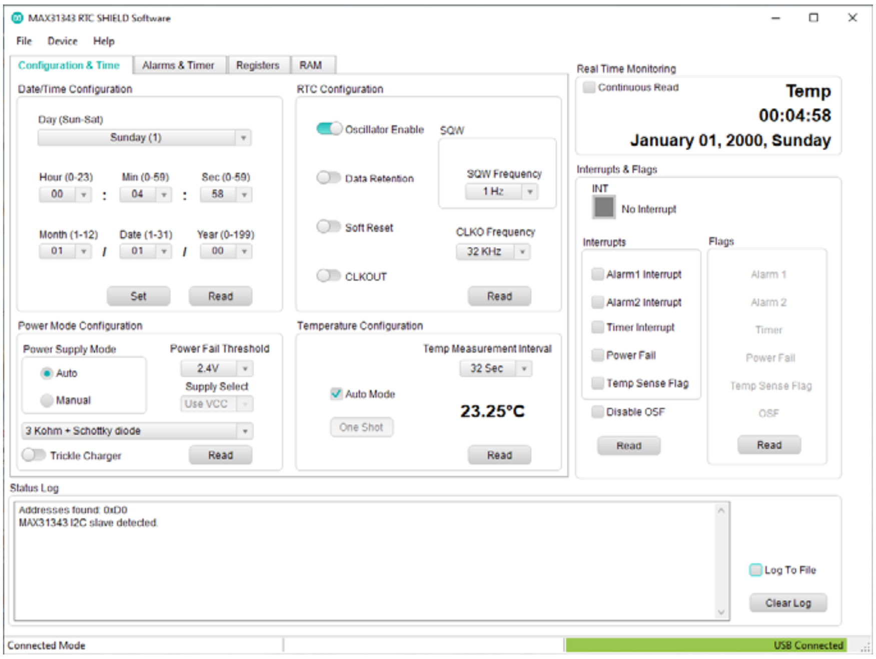
© Maxim Integrated Products, Inc.
All rights reserved.

Website: www.maximintegrated.com

Support: support.maximintegrated.com

□ Disable Splash Screen

This process barely took me a minute, as the software download is nice and compact at just 703kB. The software itself seems very straightforward, but in case some guidance is necessary, there is also a user guide dedicated to the software.



Date/Time Configuration

Day (Sun-Sat): Sunday (1)

Hour (0-23): 00 : Min (0-59): 04 : Sec (0-59): 58

Month (1-12): 01 / Date (1-31): 01 / Year (0-199): 00

RTC Configuration

● Oscillator Enable
○ Data Retention
○ Soft Reset
○ CLKOUT

SQW: SQW Frequency: 1 Hz

CLKO Frequency: 32 KHz

Real Time Monitoring

□ Continuous Read

Temp
00:04:58
January 01, 2000, Sunday

Interrupts & Flags

INT: No Interrupt

Interrupts:
□ Alarm1 Interrupt
□ Alarm2 Interrupt
□ Timer Interrupt
□ Power Fail
□ Temp Sense Flag
□ Disable OSF

Flags: Alarm 1, Alarm 2, Timer, Power Fail, Temp Sense Flag, OSF

Power Mode Configuration

Power Supply Mode: ● Auto ○ Manual

Power Fail Threshold: 2.4V

Supply Select: Use VCC

3 Kohm + Schottky diode

○ Trickle Charger

Temperature Configuration

Temp Measurement Interval: 32 Sec

☑ Auto Mode

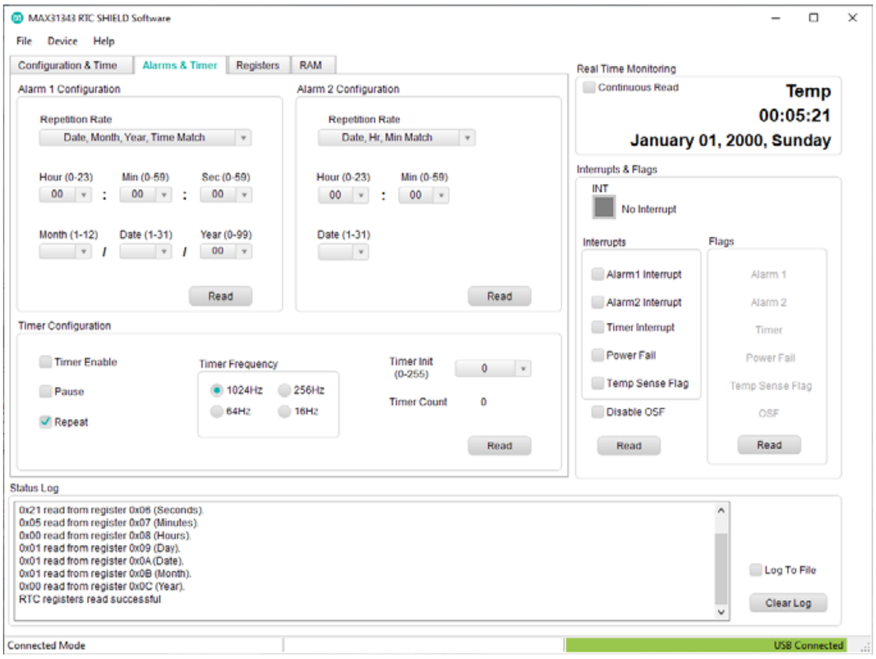
23.25°C

Status Log

```
Addresses found: 0xD0
MAX31343 I2C slave detected.
```

□ Log To File

Connected Mode | USB Connected



Alarm 1 Configuration

Repetition Rate: Date, Month, Year, Time Match

Hour (0-23): 00 : Min (0-59): 00 : Sec (0-59): 00

Month (1-12): / Date (1-31): / Year (0-99): 00

Alarm 2 Configuration

Repetition Rate: Date, Hr, Min Match

Hour (0-23): 00 : Min (0-59): 00

Date (1-31):

Real Time Monitoring

□ Continuous Read

Temp
00:05:21
January 01, 2000, Sunday

Interrupts & Flags

INT: No Interrupt

Interrupts:
□ Alarm1 Interrupt
□ Alarm2 Interrupt
□ Timer Interrupt
□ Power Fail
□ Temp Sense Flag
□ Disable OSF

Flags: Alarm 1, Alarm 2, Timer, Power Fail, Temp Sense Flag, OSF

Timer Configuration

□ Timer Enable
□ Pause
☑ Repeat

Timer Frequency: ● 1024Hz ○ 256Hz ○ 64Hz ○ 16Hz

Timer Init (0-255): 0

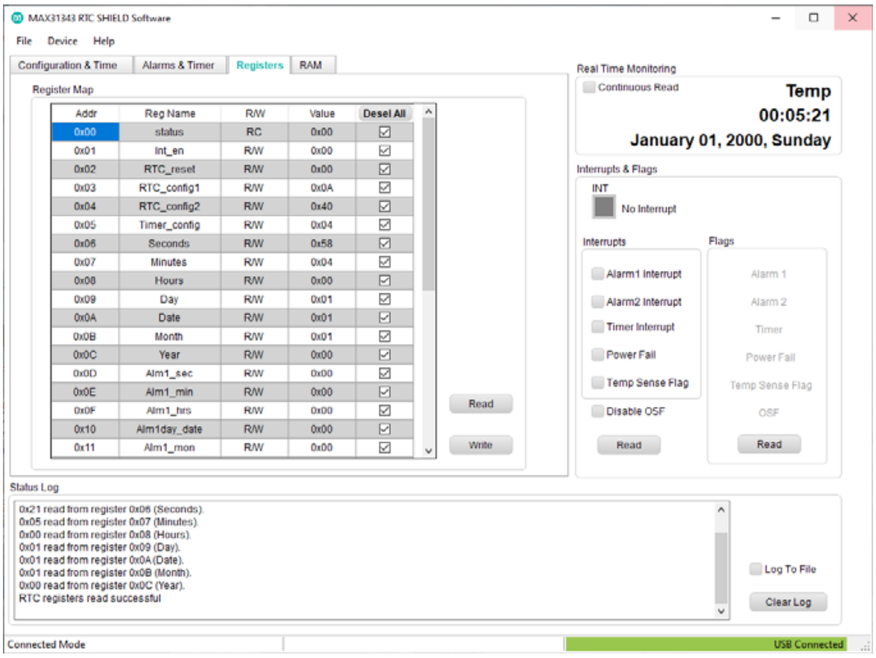
Timer Count: 0

Status Log

```
0x21 read from register 0x06 (Seconds).
0x05 read from register 0x07 (Minutes).
0x00 read from register 0x08 (Hours).
0x01 read from register 0x09 (Day).
0x01 read from register 0x0A (Date).
0x01 read from register 0x0B (Month).
0x00 read from register 0x0C (Year).
RTC registers read successful
```

□ Log To File

Connected Mode | USB Connected



Register Map

Addr	Reg Name	R/W	Value	Desel All
0x00	status	RC	0x00	☑
0x01	Int_en	R/W	0x00	☑
0x02	RTC_reset	R/W	0x00	☑
0x03	RTC_config1	R/W	0x0A	☑
0x04	RTC_config2	R/W	0x40	☑
0x05	Timer_config	R/W	0x04	☑
0x06	Seconds	R/W	0x58	☑
0x07	Minutes	R/W	0x04	☑
0x08	Hours	R/W	0x00	☑
0x09	Day	R/W	0x01	☑
0x0A	Date	R/W	0x01	☑
0x0B	Month	R/W	0x01	☑
0x0C	Year	R/W	0x00	☑
0x0D	Alm1_sec	R/W	0x00	☑
0x0E	Alm1_min	R/W	0x00	☑
0x0F	Alm1_hrs	R/W	0x00	☑
0x10	Alm1day_date	R/W	0x00	☑
0x11	Alm1_mon	R/W	0x00	☑

Real Time Monitoring

□ Continuous Read

Temp

00:05:21

January 01, 2000, Sunday

Interrupts & Flags

INT ■ No Interrupt

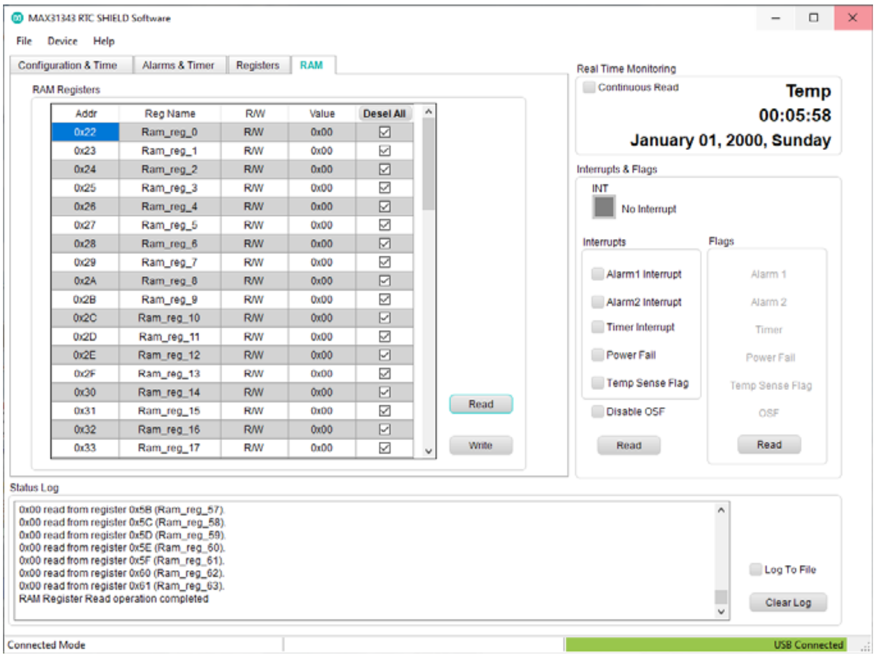
Interrupts: □ Alarm1 Interrupt, □ Alarm2 Interrupt, □ Timer Interrupt, □ Power Fail, □ Temp Sense Flag, □ Disable OSF

Flags: Alarm 1, Alarm 2, Timer, Power Fail, Temp Sense Flag, OSF

Status Log

```
0x21 read from register 0x06 (Seconds).
0x05 read from register 0x07 (Minutes).
0x00 read from register 0x08 (Hours).
0x01 read from register 0x09 (Day).
0x01 read from register 0x0A (Date).
0x01 read from register 0x0B (Month).
0x00 read from register 0x0C (Year).
RTC registers read successful
```

Connected Mode | USB Connected



RAM Registers

Addr	Reg Name	R/W	Value	Desel All
0x22	Ram_reg_0	R/W	0x00	☑
0x23	Ram_reg_1	R/W	0x00	☑
0x24	Ram_reg_2	R/W	0x00	☑
0x25	Ram_reg_3	R/W	0x00	☑
0x26	Ram_reg_4	R/W	0x00	☑
0x27	Ram_reg_5	R/W	0x00	☑
0x28	Ram_reg_6	R/W	0x00	☑
0x29	Ram_reg_7	R/W	0x00	☑
0x2A	Ram_reg_8	R/W	0x00	☑
0x2B	Ram_reg_9	R/W	0x00	☑
0x2C	Ram_reg_10	R/W	0x00	☑
0x2D	Ram_reg_11	R/W	0x00	☑
0x2E	Ram_reg_12	R/W	0x00	☑
0x2F	Ram_reg_13	R/W	0x00	☑
0x30	Ram_reg_14	R/W	0x00	☑
0x31	Ram_reg_15	R/W	0x00	☑
0x32	Ram_reg_16	R/W	0x00	☑
0x33	Ram_reg_17	R/W	0x00	☑

Real Time Monitoring

□ Continuous Read

Temp

00:05:58

January 01, 2000, Sunday

Interrupts & Flags

INT ■ No Interrupt

Interrupts: □ Alarm1 Interrupt, □ Alarm2 Interrupt, □ Timer Interrupt, □ Power Fail, □ Temp Sense Flag, □ Disable OSF

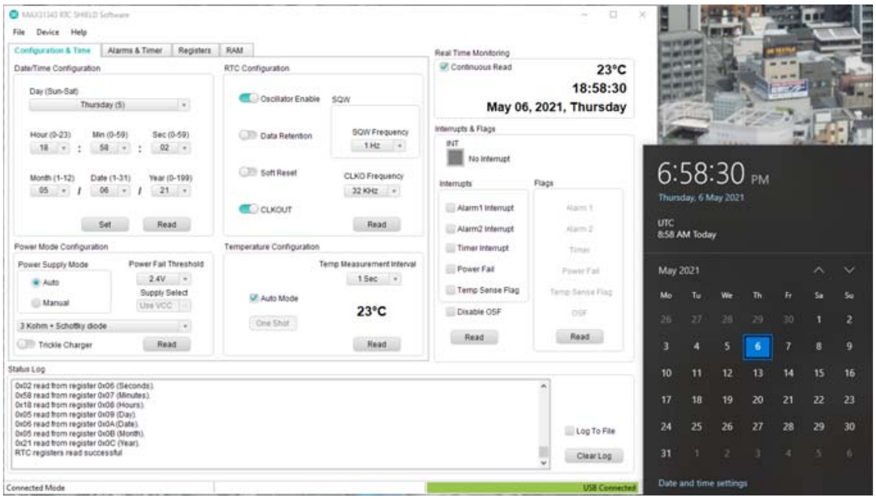
Flags: Alarm 1, Alarm 2, Timer, Power Fail, Temp Sense Flag, OSF

Status Log

```
0x00 read from register 0x5B (Ram_reg_57).
0x00 read from register 0x5C (Ram_reg_58).
0x00 read from register 0x5D (Ram_reg_59).
0x00 read from register 0x5E (Ram_reg_60).
0x00 read from register 0x5F (Ram_reg_61).
0x00 read from register 0x60 (Ram_reg_62).
0x00 read from register 0x61 (Ram_reg_63).
RAM Register Read operation completed
```

Connected Mode | USB Connected

The software consists mainly of four tabs which allow you to comprehensively alter the configuration and read-back values from the RTC. The configuration and time tab allows for basic configuration of date, time, power supply mode, trickle charger, clock/square wave outputs, temperature conversion intervals and interrupts. The alarms and timer tab allows for configuration of both alarms and the timer function. The registers tab allows for direct reading and writing of values into all registers, allowing a very "raw" level of access. Finally, the RAM tab allows for reading and writing the 64-byte RAM memory locations on the chip.



Date/Time Configuration

Day (Sun-Sat): Thursday (5)

Hour (0-23): 18 : Min (0-59): 58 : Sec (0-59): 02

Month (1-12): 05 / Date (1-31): 06 / Year (0-199): 21

RTC Configuration

● Oscillator Enable

○ Data Retention

○ Soft Reset

● CLKOUT

SQW — SQW Frequency: 1 Hz

CLKO Frequency: 32 KHz

Power Mode Configuration

Power Supply Mode: ● Auto ○ Manual

Power Fail Threshold: 2.4V

Supply Select: Use VCC

3 Kohm + Schottky diode

○ Trickle Charger

Temperature Configuration

Temp Measurement Interval: 1 Sec

☑ Auto Mode

23°C

Real Time Monitoring

☑ Continuous Read

23°C

18:58:30

May 06, 2021, Thursday

Interrupts & Flags

INT ■ No Interrupt

Interrupts: □ Alarm1 Interrupt, □ Alarm2 Interrupt, □ Timer Interrupt, □ Power Fail, □ Temp Sense Flag, □ Disable OSF

Flags: Alarm 1, Alarm 2, Timer, Power Fail, Temp Sense Flag, OSF

Status Log

```
0x02 read from register 0x06 (Seconds).
0x58 read from register 0x07 (Minutes).
0x18 read from register 0x08 (Hours).
0x05 read from register 0x09 (Day).
0x06 read from register 0x0A (Date).
0x05 read from register 0x0B (Month).
0x21 read from register 0x0C (Year).
RTC registers read successful
```

Connected Mode | USB Connected

6:58:30 PM

Thursday, 6 May 2021

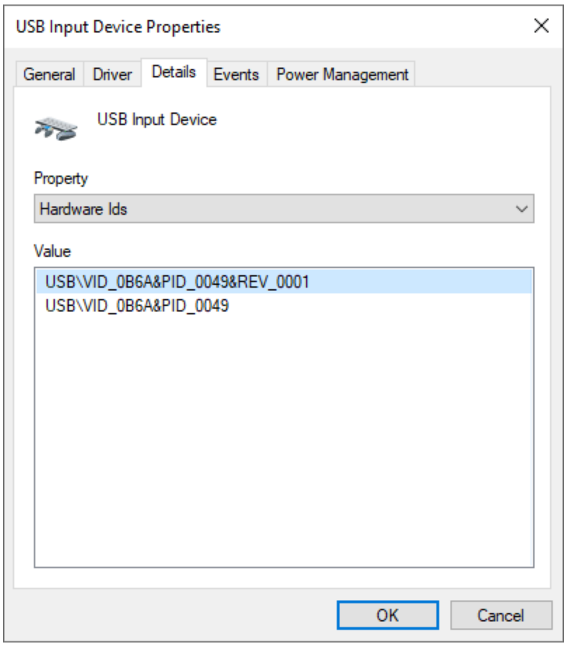
UTC
8:58 AM Today

May 2021

Mo	Tu	We	Th	Fr	Sa	Su
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Unfortunately, there is no “automatic” synchronisation of the clock with the system clock, so time setting is very much manual. Using the “continuous read” function causes the application to refresh the display in the upper-right every second.

The software is easy to work with and allows for easy set-up of parameters of the RTC chip which can be especially useful for running certain measurements on the RTC. It does not, however, feature any features which could be used to gauge the long-term accuracy of the clock. It also is not designed for use on platforms other than Windows 10.



USB Input Device

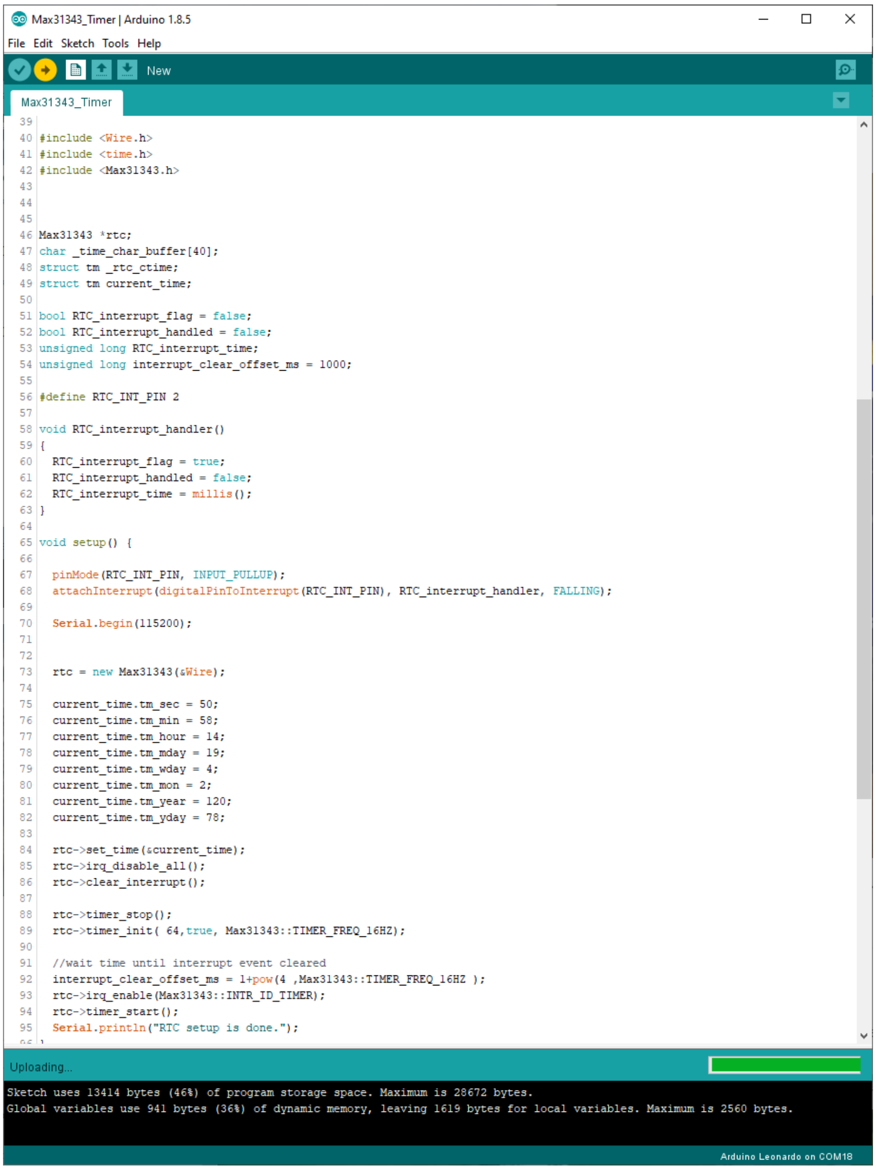
Property: Hardware Ids

Value:

USB\VID_0B6A&PID_0049&REV_0001
USB\VID_0B6A&PID_0049

The PICO is programmed in such a way that it connects to the PC as a USB HID class device, such that it requires no specific drivers, but no guidance is given on how to write your own applications to interface through this device. This limits the possibility of running more sophisticated experiments using the PICO, or writing your own applet for other platforms.

Fortunately, as the kit is designed as an Arduino shield, if you bring-your-own microcontroller to the party, you can get started more easily as Maxim provides an Arduino Library with two examples.

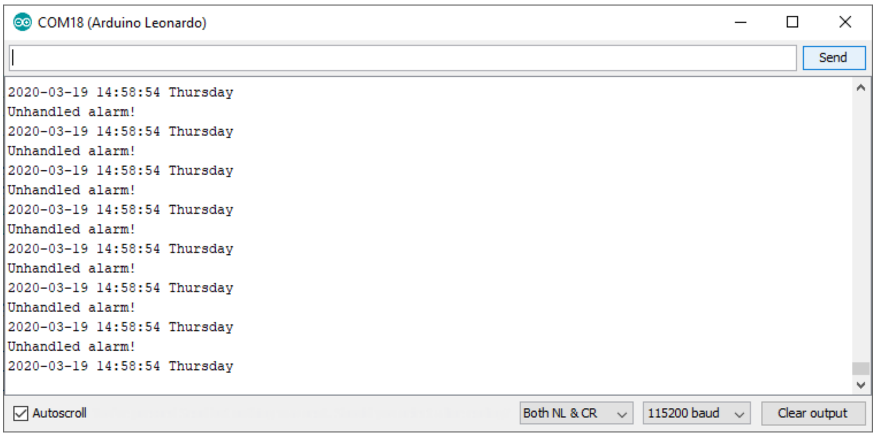


```

#include <Wire.h>
#include <time.h>
#include <Max31343.h>



Max31343 *rtc;
char _time_char_buffer[40];
struct tm _rtc_ctime;
struct tm current_time;

bool RTC_interrupt_flag = false;
bool RTC_interrupt_handled = false;
unsigned long RTC_interrupt_time;
unsigned long interrupt_clear_offset_ms = 1000;

#define RTC_INT_PIN 2

void RTC_interrupt_handler()
{
  RTC_interrupt_flag = true;
  RTC_interrupt_handled = false;
  RTC_interrupt_time = millis();
}

void setup() {

  pinMode(RTC_INT_PIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(RTC_INT_PIN), RTC_interrupt_handler, FALLING);

  Serial.begin(115200);


  rtc = new Max31343(&Wire);

  current_time.tm_sec = 50;
  current_time.tm_min = 58;
  current_time.tm_hour = 14;
  current_time.tm_mday = 19;
  current_time.tm_wday = 4;
  current_time.tm_mon = 2;
  current_time.tm_year = 120;
  current_time.tm_yday = 78;

  rtc->set_time(&current_time);
  rtc->irq_disable_all();
  rtc->clear_interrupt();

  rtc->timer_stop();
  rtc->timer_init( 64,true, Max31343::TIMER_FREQ_16HZ);

  //wait time until interrupt event cleared
  interrupt_clear_offset_ms = 1+pow(4 ,Max31343::TIMER_FREQ_16HZ );
  rtc->irq_enable(Max31343::INTR_ID_TIMER);
  rtc->timer_start();
  Serial.println("RTC setup is done.");
```

Uploading...

```
Sketch uses 13414 bytes (46%) of program storage space. Maximum is 28672 bytes.
Global variables use 941 bytes (36%) of dynamic memory, leaving 1619 bytes for local variables. Maximum is 2560 bytes.
```

Arduino Leonardo on COM18



```
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
Unhandled alarm!
2020-03-19 14:58:54 Thursday
```

☑ Autoscroll   Both NL & CR   115200 baud   Clear output

The documentation for the library is minimal with the examples being a little sparse on explanations, but the code is usable as a starting point, requiring some time investment to become familiar with the constant declarations and functions available. I did find that running it on an Arduino Leonardo board did result in some quirks with regards to Serial printing (as is the case with all boards using the ATmega32u4), so perhaps an Uno R3 (or newer) is recommended. When I pulled the interrupt pin jumper, the library correctly detected the issue.

Aside from this, the MAX31343SHLD itself has its own documentation which covers the evaluation board configuration jumpers and test-points, experiments for timekeeping current and CLKOUT frequency, a few screenshots of the software, bill of materials, schematics and board layout. The board layout files are also available as ODB files from the download page. The documentation is a bit light on the experiments and was initially confusing to me (i.e. pin x on the connector actually corresponds to pin D[x-1] because Arduino labels its pins starting with D0). However, other than that, the documentation of the shield is sufficient and of good quality.

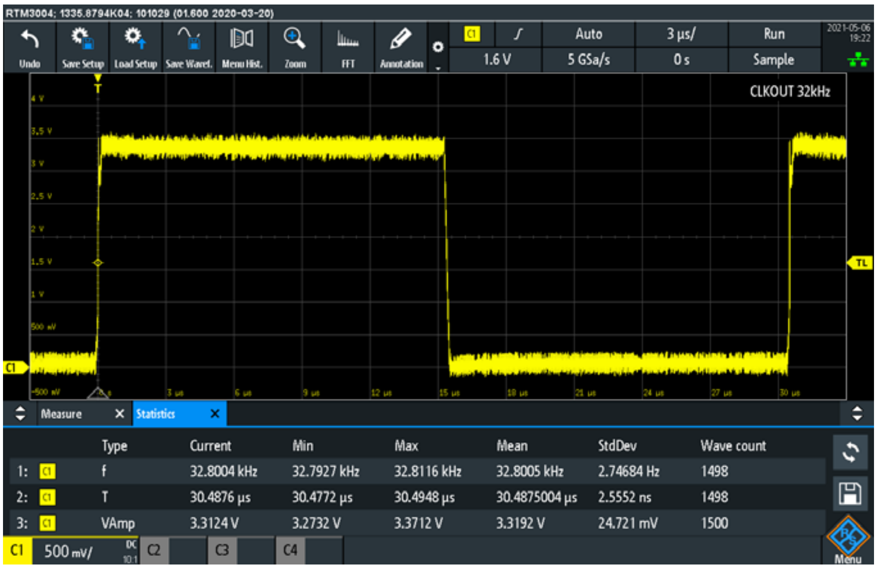
The main documentation that would be of interest to designers looking to integrate the RTC into their devices would be the datasheet for the MAX31343 itself, which is extremely exhaustive and comprehensive. I found the datasheet a good read, however, I was confused somewhat by some of the statements regarding the battery trickle charger which is apparently “UL recognised to ensure against reverse charging when used with a primary lithium battery.” This phrase usually applies to chips *without* trickle charging to indicate they do not have any “leakage” current into the port, as configuring the trickle charging on this chip could (potentially) result in reverse charging. Another point of confusion has to do with the trickle charger architecture which seems to allow the user to configure a selectable diode to be put in series with a Schottky diode always being in the path – this seems a little redundant and its use is not well elaborated (e.g. is it to add additional voltage drop when charging a 3.6V cell from a 5V supply?).

Timing Accuracy Experiments

One particularly important figure-of-merit of any real-time clock is its accuracy, especially in the face of changes in conditions such as temperature and operating voltage. The MAX31343 claims to have a +/- 5ppm accuracy which translates to a drift up to 0.432 seconds per day over a temperature range of -40 to 85 degrees Celsius which is considered to be high accuracy. I will attempt to judge its accuracy in practice based on a number of tests.

SQW & CLKOUT Frequency Offset Checks

The first test involves using the ⭐ Rohde & Schwarz RTM3004 Oscilloscope to examine the SQW (compensated) and the CLKOUT (uncompensated) outputs. The automatic temperature conversion interval was set to once-per-second to maximise accuracy. It should be noted that the RTM3004 has a timebase accuracy of +/- 3.5ppm, which is slightly more accurate than the 5ppm of the MAX31343 but is not accurate enough to necessarily provide firm conclusions about the accuracy of the MAX31343. For that, you would probably need a timebase accuracy of around 1ppm or less which puts you into oven-controlled crystal oscillator (OCXO) territory and that is just something I don’t have.



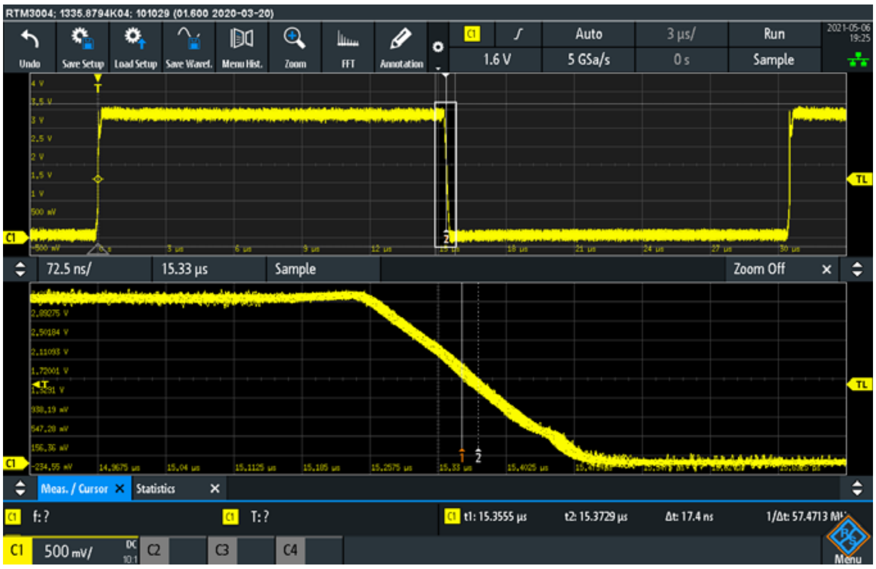
RTM3004; 1335.8794K04; 101029 (01.600 2020-03-20)

C1 1.6 V Auto 5 GSa/s 3 µs/ 0 s Run Sample 2021-05-06 19:22

CLKOUT 32kHz

		Type	Current	Min	Max	Mean	StdDev	Wave count
1:	C1	f	32.8004 kHz	32.7927 kHz	32.8116 kHz	32.8005 kHz	2.74684 Hz	1498
2:	C1	T	30.4876 µs	30.4772 µs	30.4948 µs	30.4875004 µs	2.5552 ns	1498
3:	C1	VAmp	3.3124 V	3.2732 V	3.3712 V	3.3192 V	24.721 mV	1500

C1 500 mV/ DC 10:1



RTM3004; 1335.8794K04; 101029 (01.600 2020-03-20)

C1 1.6 V Auto 5 GSa/s 3 µs/ 0 s Run Sample 2021-05-06 19:25

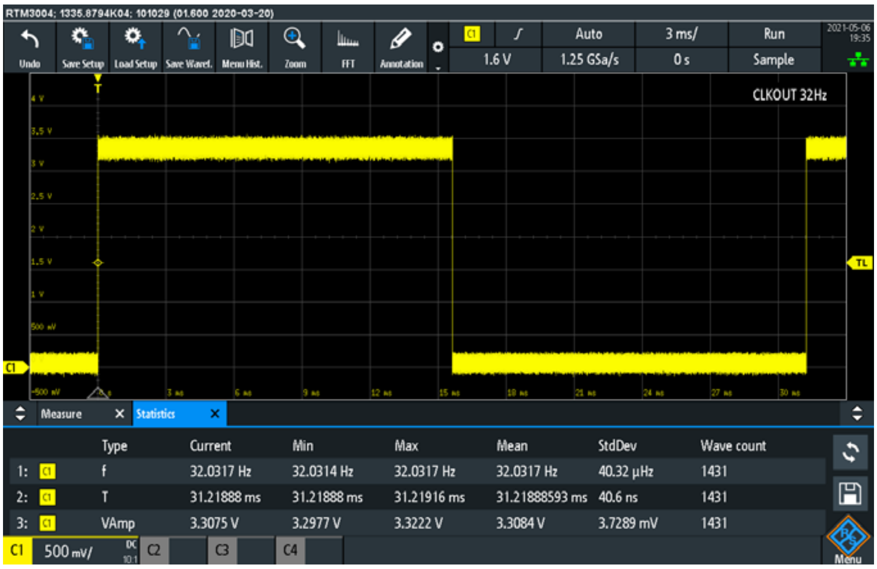
72.5 ns/ 15.33 µs Sample Zoom Off

C1 f: ? C1 T: ? C1 t1: 15.3555 µs t2: 15.3729 µs Δt: 17.4 ns 1/Δt: 57.4713 MHz

C1 500 mV/ DC 10:1

Examining CLKOUT when configured to 32kHz, the frequency appears to average about 32.8kHz which is a **deviation of +2.5%**. This seems to be more than the 1% that the datasheet claims, but I also note the unclean rising edge transition of the signal and noise overall. I wonder if the 3.3V rail from the PICO is noisy enough to compromise the true accuracy of the MAX31343.

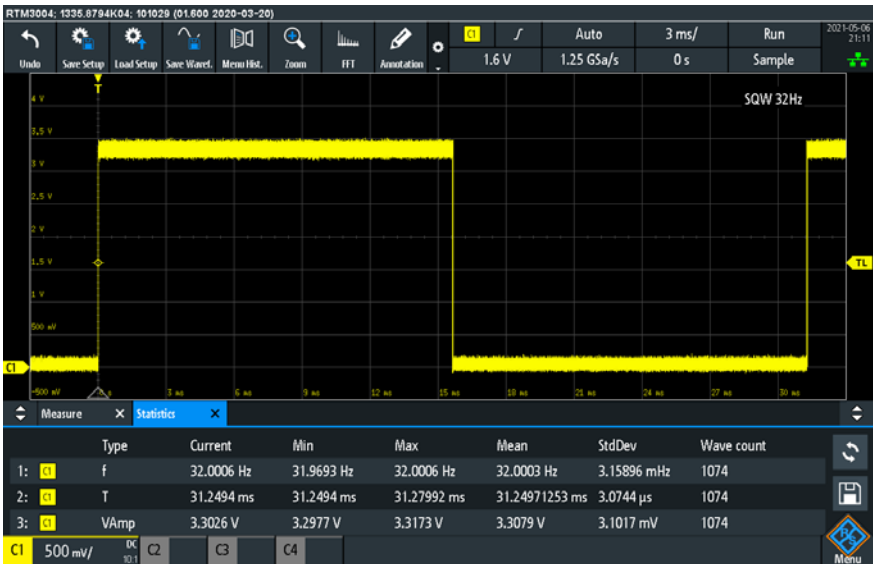
Looking very closely at the falling edge, a small jitter in the signal is visible. The difference is just 17.4ns, so not particularly relevant given the ideal period of 33.3µs. The deviation of the uncompensated output is surprising, and its true cause has not been determined.



C1 1.6 V | Auto 1.25 GSa/s | 3 ms/ 0 s | Run Sample

	Type	Current	Min	Max	Mean	StdDev	Wave count
1: C1	f	32.0317 Hz	32.0314 Hz	32.0317 Hz	32.0317 Hz	40.32 µHz	1431
2: C1	T	31.21888 ms	31.21888 ms	31.21916 ms	31.21888593 ms	40.6 ns	1431
3: C1	VAmp	3.3075 V	3.2977 V	3.3222 V	3.3084 V	3.7289 mV	1431

C1 500 mV/ DC 10:1 C2 C3 C4

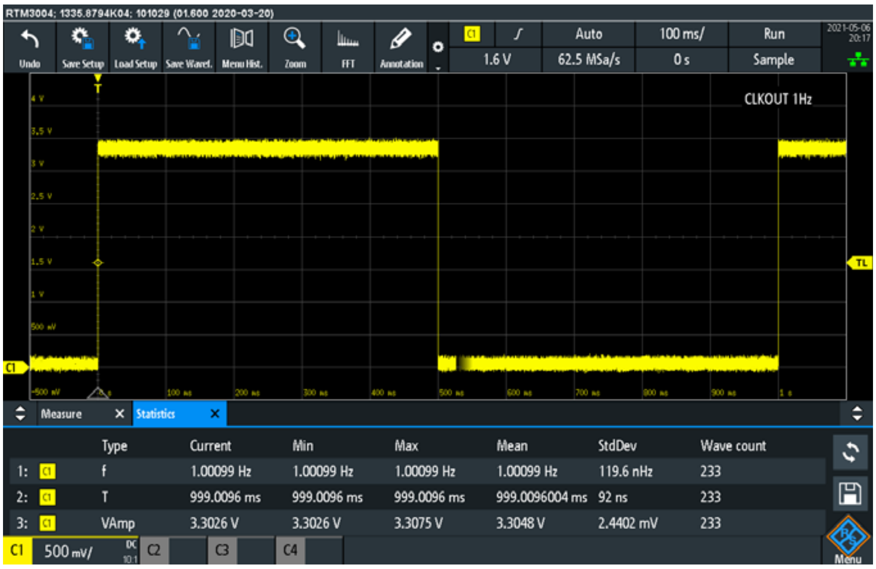


C1 1.6 V | Auto 1.25 GSa/s | 3 ms/ 0 s | Run Sample

	Type	Current	Min	Max	Mean	StdDev	Wave count
1: C1	f	32.0006 Hz	31.9693 Hz	32.0006 Hz	32.0003 Hz	3.15896 mHz	1074
2: C1	T	31.2494 ms	31.2494 ms	31.27992 ms	31.24971253 ms	3.0744 µs	1074
3: C1	VAmp	3.3026 V	3.2977 V	3.3173 V	3.3079 V	3.1017 mV	1074

C1 500 mV/ DC 10:1 C2 C3 C4

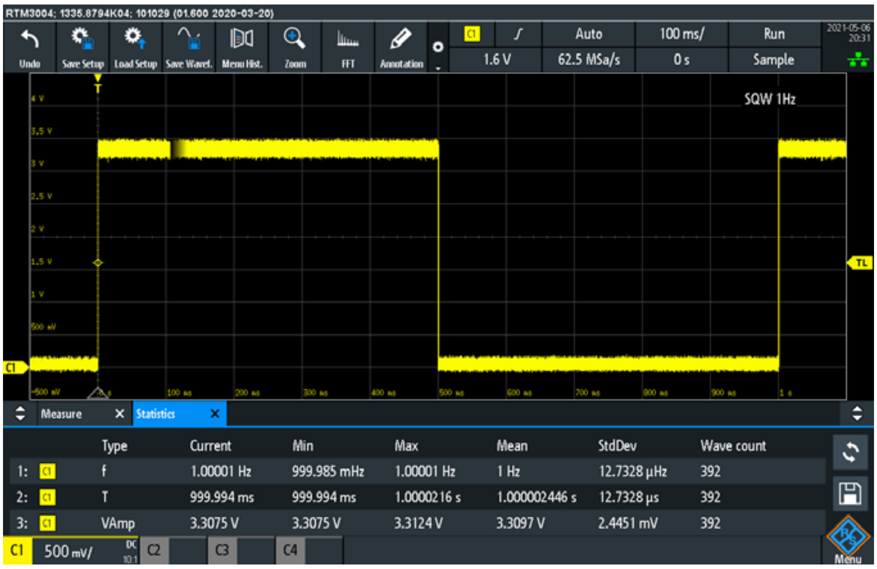
While I did examine every frequency step the programmable CLKOUT and SQW outputs allow for, the maximum common programmable frequency is 32Hz. Comparing the two, the uncompensated output measured 32.0317Hz (within 0.1%) while the compensated output measured 32.0003Hz (around +9.2ppm). This time, the uncompensated output is within the datasheet claim, but the compensated output is a bit off the 5ppm we expect, even if accounting for up to 3.5ppm error from the oscilloscope. Perhaps the temperature has not entirely settled, but at least we are in the ballpark.



C1 1.6 V | Auto 62.5 MSa/s | 100 ms/ 0 s | Run Sample

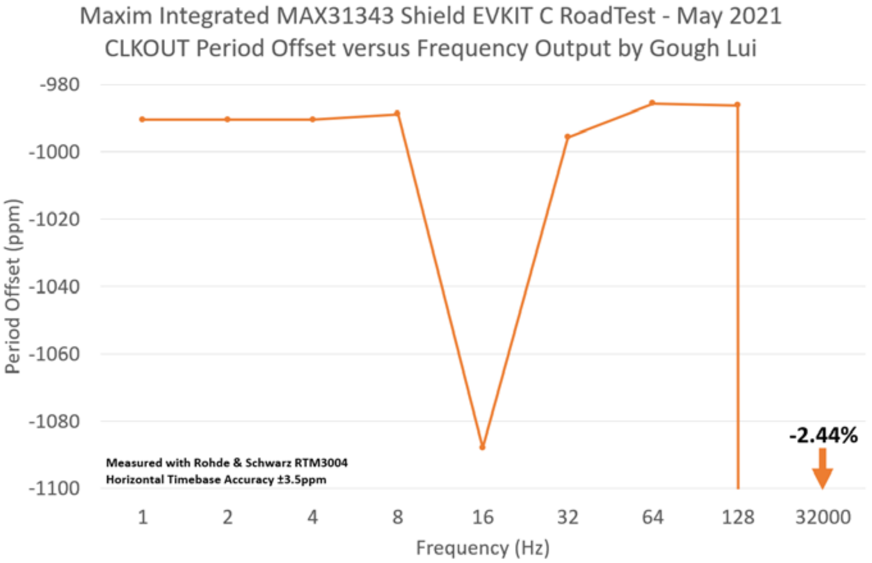
	Type	Current	Min	Max	Mean	StdDev	Wave count
1: C1	f	1.00099 Hz	1.00099 Hz	1.00099 Hz	1.00099 Hz	119.6 nHz	233
2: C1	T	999.0096 ms	999.0096 ms	999.0096 ms	999.0096004 ms	92 ns	233
3: C1	VAmp	3.3026 V	3.3026 V	3.3075 V	3.3048 V	2.4402 mV	233

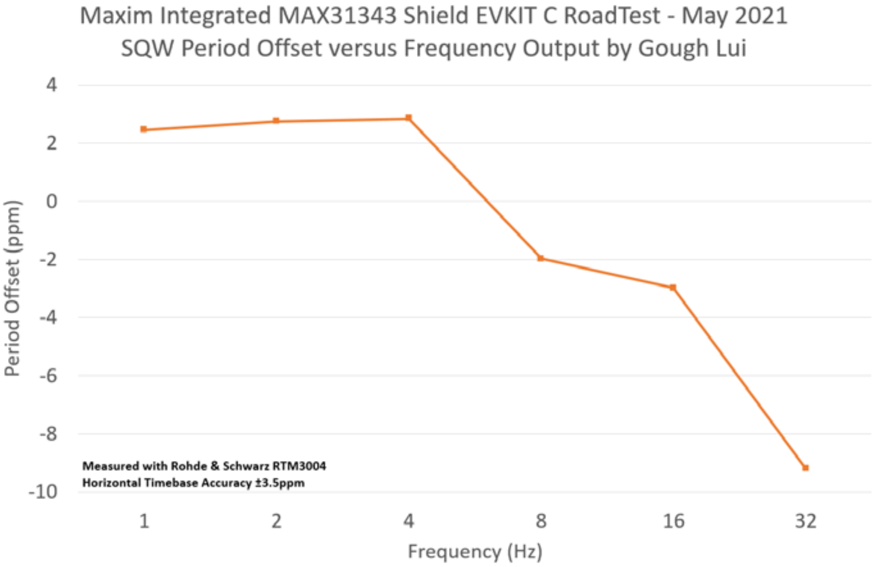
C1 500 mV/ DC 10:1 C2 C3 C4



		Type	Current	Min	Max	Mean	StdDev	Wave count
1:	C1	f	1.00001 Hz	999.985 mHz	1.00001 Hz	1 Hz	12.7328 µHz	392
2:	C1	T	999.994 ms	999.994 ms	1.0000216 s	1.000002446 s	12.7328 µs	392
3:	C1	VAmp	3.3075 V	3.3075 V	3.3124 V	3.3097 V	2.4451 mV	392

The lowest common programmable frequency is 1Hz, so I decided to compare the two as well. The uncompensated output measured 1.00099Hz (within 0.1%) and the compensated output measured pretty much right-on 1Hz (error of 2.4ppm). The observed error may just depend on the operational parameters chosen and the test conditions, but this is evidence that suggests the MAX31343 is potentially quite accurate.

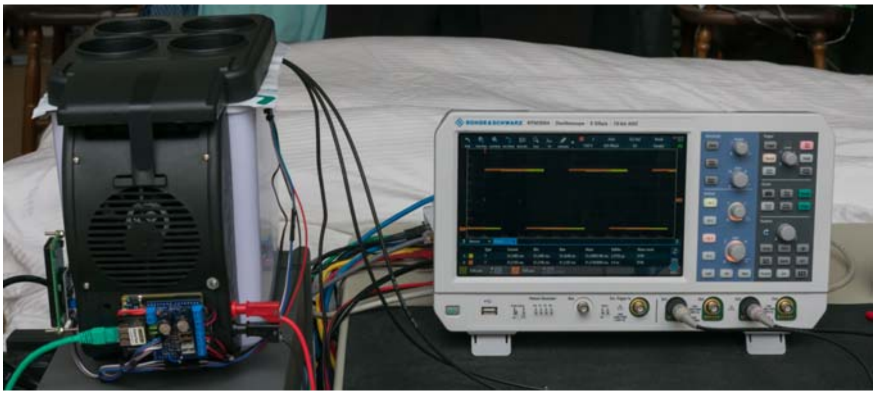




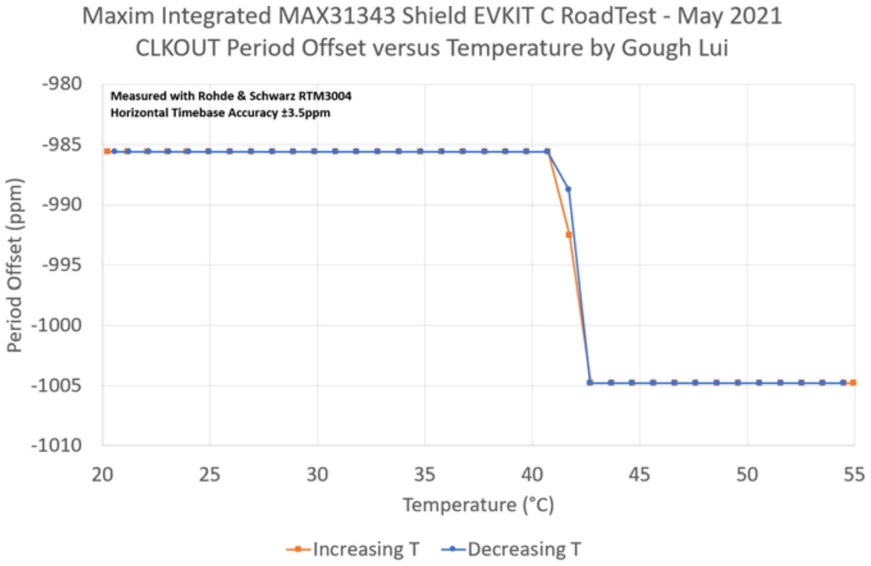
Summarising my “walk” through every configurable output combination of both compensated and uncompensated output, it seems that the uncompensated output remained within around 0.1% for every frequency except 32kHz, while the compensated output wandered between around +3ppm and -3ppm for all selections except 32Hz which registered -10ppm. Perhaps the reason for this has to do with thermal compensation, but it seems 5ppm accuracy is quite possible.

SQW & CLKOUT Frequency Versus Temperature

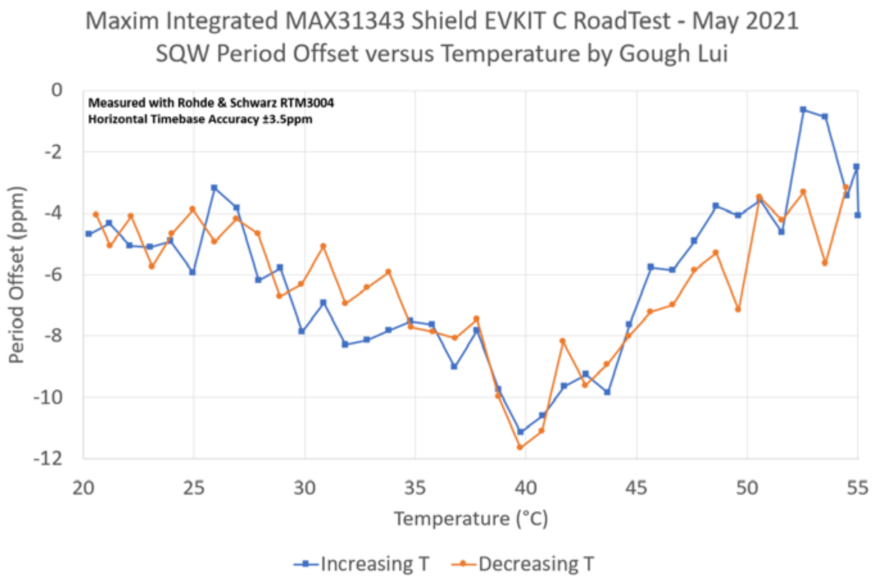
While the above check mainly grabbed one “snapshot” of the RTCs performance at the ambient temperature of the time, how it would perform as the temperature changes is something I was quite interested in. Because the RTM3004 is in the room at ambient temperature, the temperature excursions it would experience are very limited, compared to the MAX31343 which I placed inside a modified car fridge for temperature control.



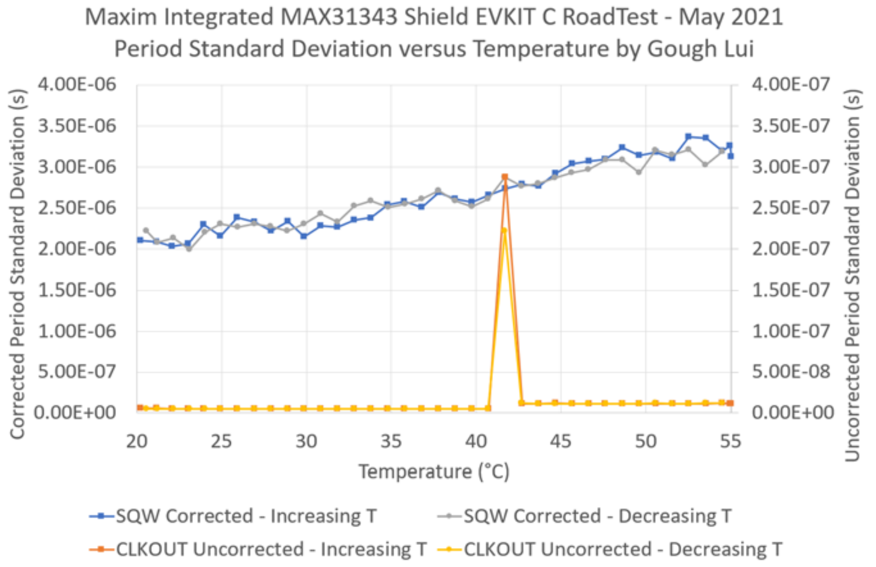
In this experiment, I configured the RTC to put out 32Hz on both CLKOUT and SQW to be measured on Channel 1 and Channel 3 on the RTM3004. I used a pyvisa script to set the temperature on the fridge to its initial temperature of 20 degrees Celsius, held the temperature for an hour, commencing the measurement which would reset the measurement statistics on the oscilloscope, measure the frequency and period over a time of 600 seconds (more than the 512s in the datasheet) and log it to disk. It would then increase the temperature by 1 degree Celsius, hold the temperature for 20 minutes, then repeat until it had reached a maximum temperature of 55 degrees Celsius (the limit of my home-made temperature chamber). It would then sweep the sequence **in reverse**, reducing one degree at a time and dwelling for 20 minutes. Running this test sequence took *a few days* to complete.



The uncompensated output was found to vary from about -986ppm to -1005ppm, with a **noticeable** change in the period offset at about 42 degrees Celsius in both upward and downward transitions. This suggests to me that this measured effect is **real** and physical in nature – perhaps there is a vibrational mode change in the MEMS structure which occurs at this temperature.



The compensated output showed a period offset ranging from about 0ppm to -12ppm with both increasing temperature and decreasing temperature curves following a similar trend where the maximum error occurred at around 40 degrees Celsius. It seems that the temperature-based compensation may take this step-change into account, but the correction factors may have its greatest deviation around 40 degrees Celsius. It should be noted that while the span is greater than what a +/- 5ppm device would be expected to have, when considering the oscilloscope's 3.5ppm accuracy, this is inconclusive as to whether the MAX31343 is within or out of specification.



It seems that an analysis of standard deviations shows that the corrected output has only very minor increases in standard deviation as the temperature increases, while the uncorrected seems to have two stable values with a spike in-between during the transition between the two stable values. The overall standard deviation of the uncorrected output is extremely low, indicating a very stable clock source, but the corrected output exhibits noticeably more jitter likely because of the digitally temperature compensated architecture of this RTC which will be adding or removing "pulses" depending on the temperature.

Longer-Term NTP (Stratum-2) Tests

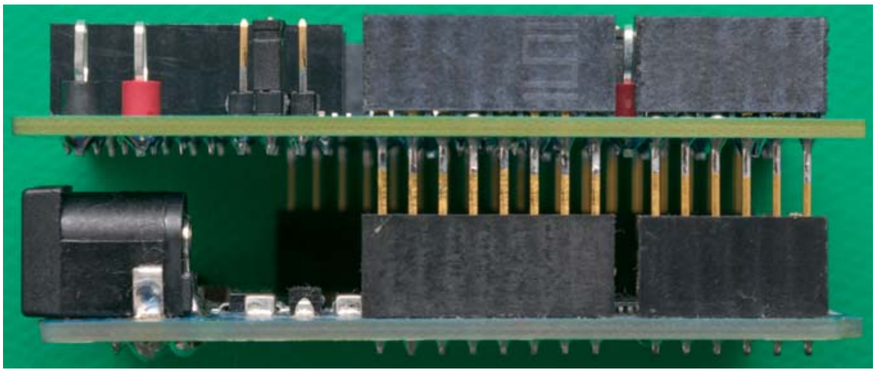
Unfortunately, while the previous two experiments did hint at some encouraging accuracy levels, they were not conclusive about it. Ideally, in order to test accuracy, it requires running the RTC for extended periods of time and comparing its output phase-wise to a highly stable time source – this could include an atomic clock or GPS disciplined oscillator. Unfortunately, I have neither of those at my disposal at this time, so I had to settle for *borrowing* someone else's.

It's probably common knowledge that the Network Time Protocol (NTP) allows for synchronisation of clocks across the internet. In the early days, this functionality was advertised as providing "atomic" accuracy to everyone, but of course this is limited in practice by the vagaries of internet transmission delays and jitters. A full NTP implementation checks time (chimes) against multiple servers in parallel, running the measured samples through a filter and disciplining the oscillator on a host to get as close as possible to zero error. This setup protects against incorrectly configured NTP servers which may be disseminating incorrect time values (a falseticker).

In the NTP field, servers can provide time to downstream clients, but each server has its own rank, termed a "stratum" which describes how **close** they are to an authoritative source of time. Stratum-1 servers are those connected **directly** to a time reference – an atomic clock, a GPS receiver with a PPS signal, or a shortwave radio receiver are common ways to achieve a Stratum-1 server. Many servers however, are not connected to a Stratum-1 time source, instead receiving their time from an upstream NTP server. As a result, each of these servers advertises a Stratum level one-higher than the source – e.g. a server that is chiming against a Stratum-1 source would itself be Stratum-2.

As NTP servers are a vital part of the internet infrastructure, larger organisations such as universities and telecommunications companies often offer their own NTP servers which customers can use. These servers are frequently at Stratum-2 or Stratum-3 level, which is suitable for ordinary use, but not as favourable where high precision is required. Stratum-1 servers are not as widely openly available as in the past due to abuse. Likewise, precision is also impacted on by the jitter and asymmetry in internet connection routing delays, so servers which are geographically and topologically closer to the user will provide more stable and accurate time.

While I did consider running my own Stratum-1 from home and had some GPS modules on-order, the global semiconductor shortage has put the brakes on that plan, as my GPS receivers still have not shipped at the point of writing this review. To best analyse the MAX31343's long-term accuracy, I decided to construct a test using NTP as a time reference as follows.



The shield will be mounted to an Arduino Leonardo board running a very simple sketch I call "ticker.ino". This particular sketch sets up the RTC to put out a compensated 1Hz square wave which triggers an interrupt on the Leonardo causing it to send a single ASCII "T" to the host over its USB to Serial. The choice of the Leonardo is **deliberate** as it uses a 32u4 which handles the USB internally, resulting in no USB-to-Serial timing jitter, which should allow for accuracies of about 1ms.

view plain

```
#include <Wire.h>
#include <Max31343.h>

Max31343 *rtc;

void MAX31343_interrupt_handler() {
  Serial.print("T");
}

void setup() {
  pinMode(3, INPUT);
  Serial.begin(115200);
  while(!Serial) {
  }
  rtc = new Max31343(&Wire);
  rtc->rtc_start();
 rtc->set_output_square_wave_frequency(Max31343::SQUARE_WAVE_OUT_FREQ_1HZ);
  attachInterrupt(digitalPinToInterrupt(3), MAX31343_interrupt_handler, FALLING);
}

void loop() {
}
```

The host will be a Raspberry Pi Model 3B+ connected by Ethernet to the network, dedicated to the experiment. A Python program which I call "tickercheck.py" will set up my cooling fridge to the desired temperature, dwell for an hour to ensure it has equilibrated with the setpoint, and then read the serial port, logging the system clock in floating-point UNIX-time to a CSV file.

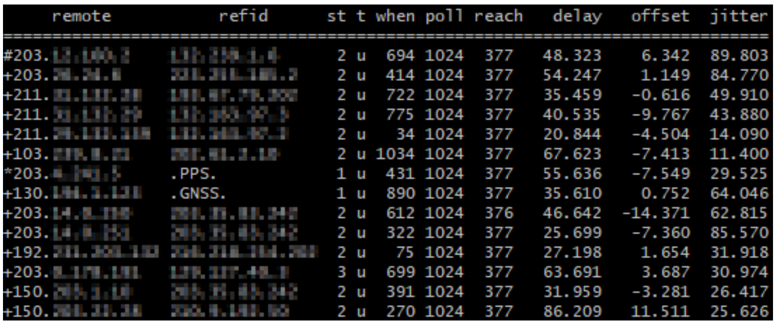
view plain

```
import time
import serial
import pyvisa

resource_manager = pyvisa.ResourceManager()
tickcount=0

print("Begin Testing")
print("Opening a logfile")
f = open("rtcticker.csv","a")

print("Setting Fridge")
ins_gfridge = resource_manager.open_resource("TCPIP0::192.168.XXX.YYY::5025::SOCKET")
ins_gfridge.read_termination = "\n"
ins_gfridge.query_termination = "\n"
ins_gfridge.timeout = 10000
ins_gfridge.query("TARGET 4000")
ins_gfridge.query("OFAN 32")
ins_gfridge.query("START")
print("Pre-heating/cooling for 1 hour ...")
time.sleep(3600)

```

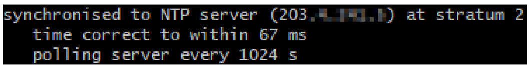
```

print("Opening Serial")
with serial.Serial("/dev/ttyACM0",115200,timeout=2) as ser:
  while(True) :
    x=ser.read()
    f.write(str(time.time())+"\r\n")
    f.flush()
    print(str(tickcount)+","+str(time.time()))
    tickcount=tickcount+1
```

Examining the resulting file should result in being able to observe the relative phase shift between the MAX31343 RTC and the system clock.

On the Raspberry Pi, the default **systemd-timesyncd** was replaced with **ntpd** to ensure reliable NTP synchronisation. This was necessary as the default daemon only perform SNTP which does not stably discipline the oscillator (resulting in very noisy output) and does not chime against multiple servers – a fact I only discovered after losing a few days of experimentation. Alternatively, I **could** have used **chrony** which is an even better implementation of NTP, however, I opted to remain with the tested and proven solution as I didn't have enough time to explore other solutions given the looming RoadTest deadline.



```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
#203.[redacted]  [redacted]       2 u  694 1024  377   48.323    6.342  89.803
+203.[redacted]  [redacted]       2 u  414 1024  377   54.247    1.149  84.770
+211.[redacted]  [redacted]       2 u  722 1024  377   35.459   -0.616  49.910
+211.[redacted]  [redacted]       2 u  775 1024  377   40.535   -9.767  43.880
+211.[redacted]  [redacted]       2 u   34 1024  377   20.844   -4.504  14.090
+103.[redacted]  [redacted]       2 u 1034 1024  377   67.623   -7.413  11.400
*203.[redacted]  .PPS.            1 u  431 1024  377   55.636   -7.549  29.525
+130.[redacted]  .GNSS.           1 u  890 1024  377   35.610    0.752  64.046
+203.[redacted]  [redacted]       2 u  612 1024  376   46.642  -14.371  62.815
+203.[redacted]  [redacted]       2 u  322 1024  377   25.699   -7.360  85.570
+192.[redacted]  [redacted]       2 u   75 1024  377   27.198    1.654  31.918
+203.[redacted]  [redacted]       3 u  699 1024  377   63.691    3.687  30.974
+150.[redacted]  [redacted]       2 u  391 1024  377   31.959   -3.281  26.417
+150.[redacted]  [redacted]       2 u  270 1024  377   86.209   11.511  25.626
```

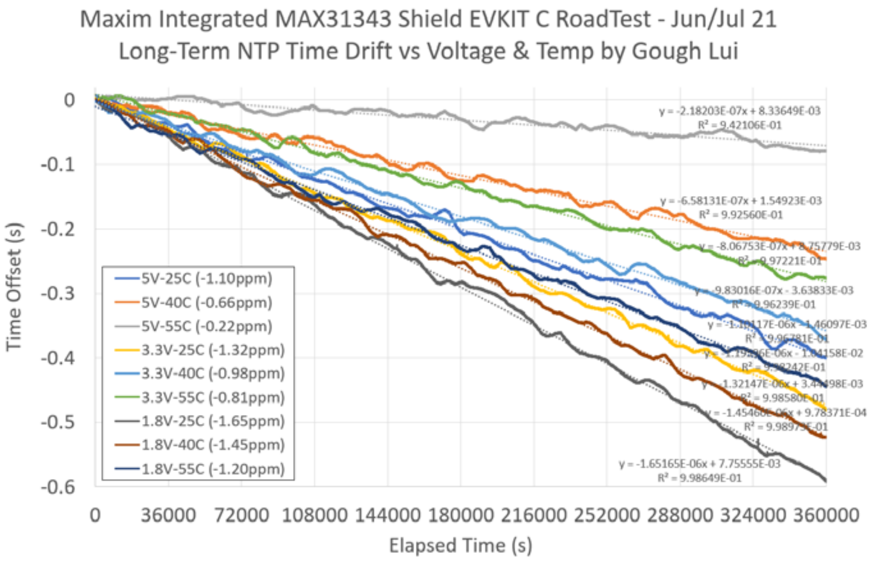


```
synchronised to NTP server (203.[redacted]) at stratum 2
   time correct to within 67 ms
   polling server every 1024 s
```

Configuration of ntpd was done with care, selecting a total of **fourteen** NTP servers from Australia and New Zealand to ensure redundancy and maintain accuracy. The vast majority are Stratum-2 servers operated by telecommunications companies, with two Stratum-1 servers, one from a telecommunications company and another from an academic institution. The resulting predicted absolute timing error is usually under 70ms which is about the best that can be expected given my LTE-based internet connection. The server addresses have been masked to avoid abuse.

The test would run for an observation period of 100 hours (360,000 seconds) at each voltage (1.8V, 3.3V and 5.0V) at three temperatures (25°C, 40°C and 55°C). The power supply at 1.8V was obtained using the Ext VCC connections from a ⭐ Keysight E36103A power supply, as the Leonardo does not supply this voltage. The resulting data would be **curve fitted** to a linear line, whose slope would be used to determine the offset of the clock, thus **reducing** the effect of the wandering local clock which is periodically corrected by NTP and improving accuracy.

Because of the protocol, **around 38 continuous days of testing** was required to obtain the data displayed in the deceptively simple-looking graph below and conclusively determine the level of accuracy this sample of the MAX31343 is able to achieve on a longer timescale.



The linear-fitted slope is what is important as it denotes the averaged rate of change between the NTP-synchronised clock and the MAX31343. The intercept represents the instantaneous offset from the average time which will depend on the NTP offset at the beginning on that particular run. The ntpstat utility considers the absolute timing accuracy at about 70ms, but in this data, the "wavering" is about 10ms, thus if we see a total drift of about 100ms, we should be fairly confident this error is mostly from the clock rather than the NTP synchronisation jitter.

On the whole, the clock's accuracy seems to be best at 5V and worst when operating at 1.8V. In all cases, the error remained between

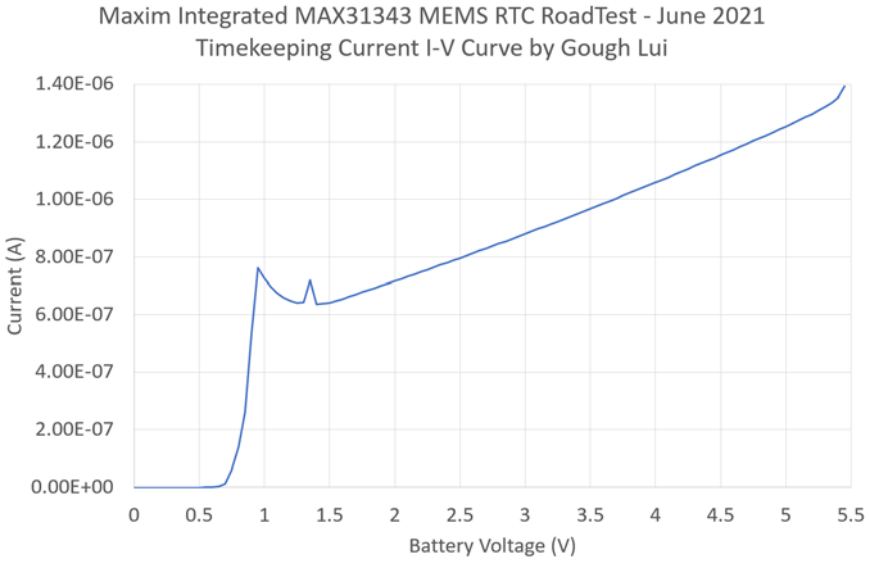
-0.22ppm and -1.65ppm, with the MAX31343 running slower. This is **comfortably within the ±5ppm specification** and is quite conclusive, with the curve-fitting R^2 value mostly above 0.99 (except 5V at 55C which only achieved 0.94 due to less overall drift). This means the expected **yearly** drift in time would be a loss of 6.9s to 52s.

It may be a bit perplexing why earlier tests were borderline. I suspect this could be because of the change in power source, from the switching power converters on the PICO which may have been noisy enough to affect the MAX31343's performance to a cleaner source of power. The previous tests being shorter run tests may also have affected the result because it may not have provided enough time for the digital temperature compensation to settle. Regardless, the long-term performance is well-within specifications and is probably the biggest priority for most RTC applications.

Power Consumption & Trickle Charging Experiments

Understanding the power consumption and trickle charging capabilities of the chip is important to properly supply back-up power to the RTC. Testing was (mostly) performed using the ⭐ Keithley 2450 SMU for accurate sourcing and measurements of such low current levels and the KickStart 2 software for running I-V curves.

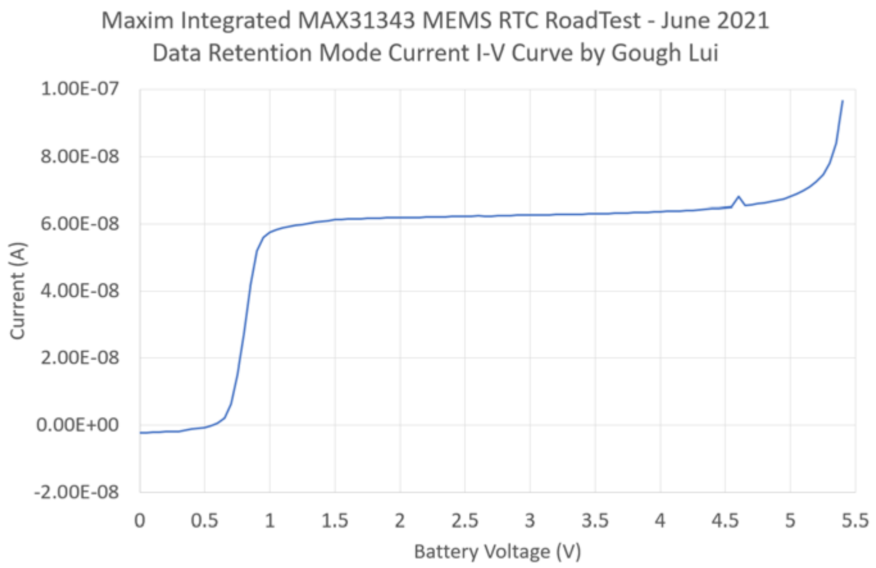
Timekeeping Current and Data Retention Mode



With the oscillator enabled, the MAX31343 appears to have a current consumption that is voltage-dependent. The operating current ranges from about 0.65µA through to about 1.4µA. At the lithium-battery standard voltage of 3V, the consumption measured 0.88µA which is within the datasheet specifications. It is thus expected that the RTC would consume about 7.7mAh of battery capacity every year. This means that a CR2032 which may have 225mAh capacity would theoretically be able to run for 29 years, although it is likely to give up before then as the battery itself fails due to internal charge leakage. Of note is that these currents also include the effect of temperature measurement at 32s intervals, matching the datasheet configuration.

When the voltage falls below operating, the current consumption rises slightly before falling to near zero when the input voltage is below about 0.65V.

In applications where the unit is shipped with a battery attached, reducing the use of energy in shipment is important to ensure the longest product life for the end user. The MAX31343 features a "data retention mode" which only keeps the battery-backed RAM data stored and turns almost everything else off to minimise energy consumption.

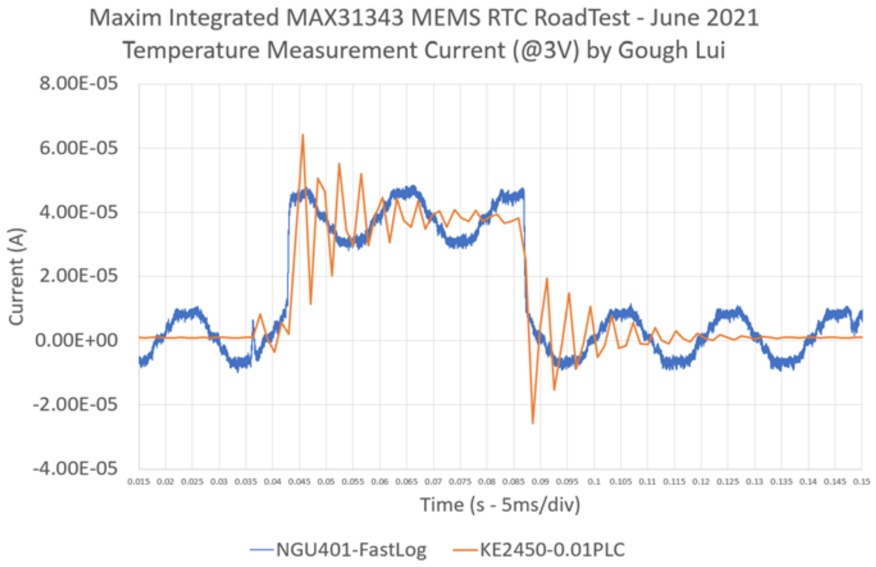


Measuring the I-V curve, it seems that retention mode current measures about 60nA for the most part – the increase towards higher voltages may be due to the effect of the capacitors on the VBAT_EXT input on the evaluation board. Such small currents are difficult to very accurately measure, however, while not as low as the 5nA typical, it is within the 100nA maximum specified by the datasheet.

Temperature Measurement Current

As the MAX31343's accuracy depends on correcting the oscillator based on sensing the temperature, the chip has to periodically measure the temperature which causes a short-duration current burst. To measure this, I used the Keithley 2450 SMU at 0.01PLC (shortest integration time), collecting the data into its internal buffer to avoid overheads from SCPI communication.

Unfortunately, while I was not successful in my bid for the Rohde & Schwarz NGU401 RoadTest, I was still able to secure the NGU401 on a short-term loan, so I decided to try to employ its FastLog to get a close look at the temperature measurement current behaviour.

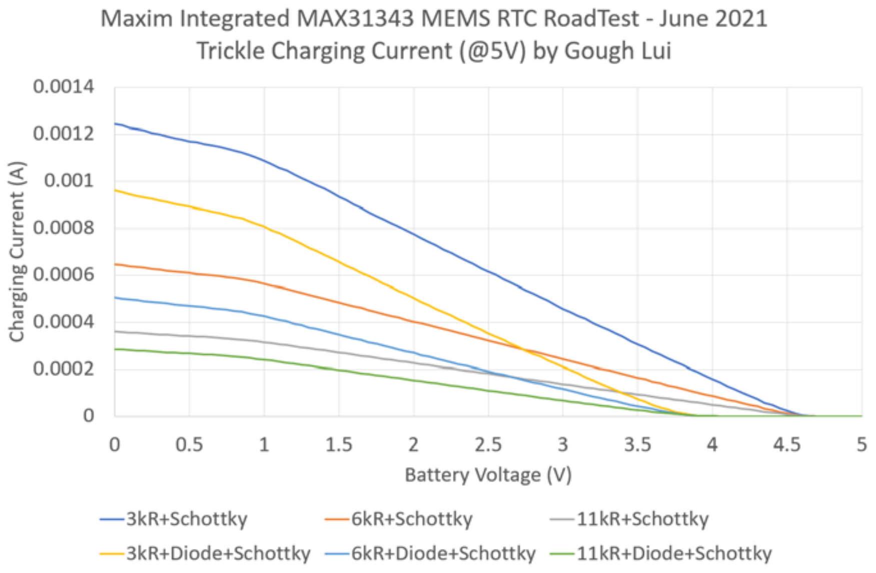


The increased sampling rate of the NGU401 gives us a clearer view of the temperature measurement current pulse, but it seems to be much more affected by the 50Hz line noise. The Keithley 2450 SMU delivers less samples, but it seems that its filtering or transient response results in the result "ringing".

Both pieces of equipment suggests that the temperature measurement pulses are of about 44ms long, requiring about 40μA at 3V. The datasheet states a typical of 40μA at 3.3V, so this is in-line with expectations.

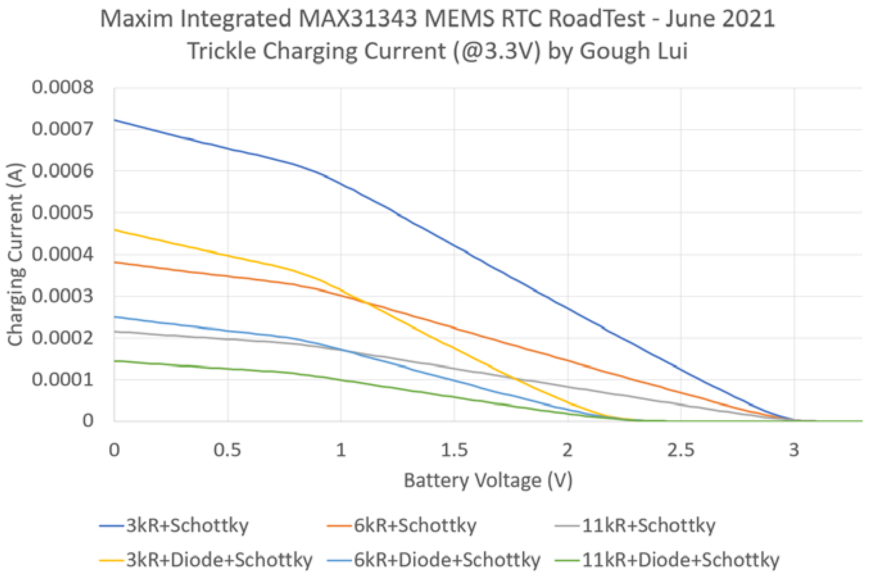
Trickle Charging – 5V

The integrated trickle charger in the MAX31343 allows for charging of back-up batteries or supercapacitors. It can be configured in six different combinations – three consisting of a resistor and Schottky diode in series, with three adding an additional diode in series.



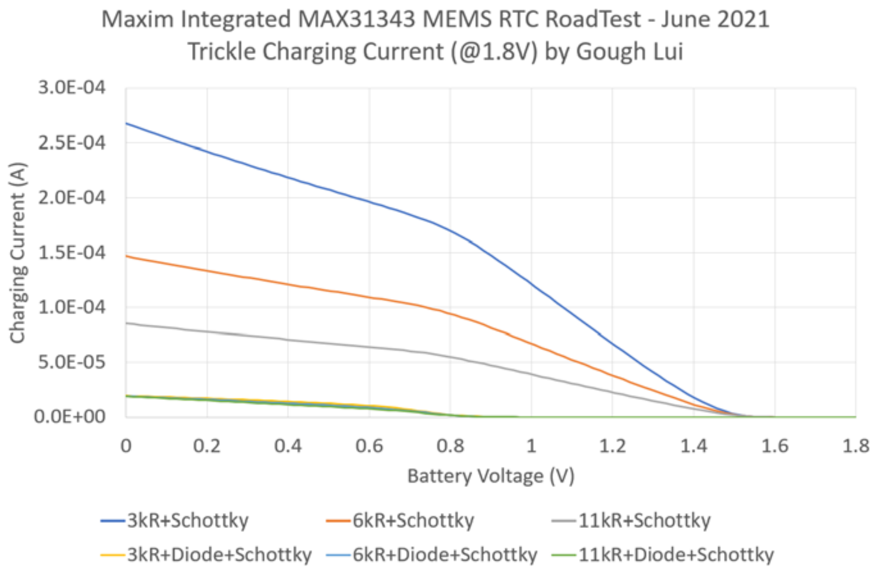
When a Vcc of 5V is supplied, the MAX31343 supplies about 4.7V without the additional diode or about 4.1V with the additional diode. The latter is possibly suitable for use with rechargeable Li-Ion and series-packs of Ni-MH cells. The current ranges from about 0.3mA to 1.28mA depending on the selected combination, so is truly a “trickle”. Because of the configuration, the charge current tapers off nearly linearly as the cell voltage rises.

Trickle Charging – 3.3V



Dropping down to a Vcc of 3.3V, the charging voltage reaches about 3.1V without the additional diode and 2.4V with the additional diode. Currents range from 0.15mA to 0.72mA, with the taper meaning that it is unlikely that you will charge to a voltage high-enough to utilise rechargeable batteries and supercapacitor use would be best.

Trickle Charging – 1.8V



While trickle charging can be enabled even with a Vcc of 1.8V, it is not useful as the cell can really only ever reach around 1.6V without the additional diode and 0.9V with the additional diode. This is below the operational voltage for the RTC, thus whatever is attached will never charge sufficiently to run the clock. Currents range from about 20µA to 260µA.

Conclusion

The MAX31343 is the latest MEMS-oscillator-based real-time clock offering from Maxim Integrated. It is one of very few MEMS-oscillator-based RTCs on the market, claiming advantages of improved physical robustness, high accuracy and small size. Compared with other RTCs on the market, it can be seen to be very competitive to both MEMS and crystal-based RTCs, with a footprint, flexibility and price that is hard to match except for possibly the MicroCrystal RV-3028-C7.

The evaluation kit was well packaged, being environmentally-friendly with no included print documentation. The board itself is of a quality construction with an excellent design that shows thoughtfulness throughout. This is seen in the various jumpers, test-points and level-shifters included in the design which allow for very flexible choices of Vcc, Vcc_IO and battery sources. The included pre-programmed MAX32625 PICO board also makes initial start-up a plug-and-play affair using the USB HID class driver and a small bloat-free evaluation tool. Perhaps the lack of a second MAX31343 in the second footprint, the lack of documentation for the USB HID protocol and the lack of an ability to synchronise the RTC with the system clock for straightforward accuracy measurements would be my criticism.

The downloadable documentation was of high quality and comprehensive. The documentation was slightly confusing at one point due to the pin-numbering conventions of the board and the Arduino ports. The Arduino library code provided is functional, however, seemed to be lacking examples and clear documentation of library functions, which may be unfortunate for those looking to evaluate the RTC using their own microcontroller as the provided software is compatible with Microsoft Windows 10-only.

The RTC itself showed mixed short-term results which were encouraging regarding its accuracy but did show some out-of-specification deviations and a clear temperature dependency. A very-long test comparing the clock to an NTP-disciplined Raspberry PI, chiming against multiple Stratium-1 and Stratum-2 sources found that the RTC was well within specification, achieving between -0.22ppm to -1.65ppm across operating voltages of 1.8, 3.3, 5.0V and temperatures of 25, 40, 55°C over a measurement period of 100 hours per run.

Measured currents were within datasheet specifications with a timekeeping current of about 0.88µA measured at 3V being strongly voltage-dependent. Data retention mode currents were about 60nA measured and relatively stable across voltages, which was above the 5nA typical but below the 100nA maximum. Temperature measurement current was measured to be 40µA for about 44ms at a supply voltage of 3V, which is consistent with datasheet expectations. Trickle charging seems most effective when used with a Vcc of 5V, delivering short-circuit currents of between 0.3mA and 1.28mA. The additional diode causes a voltage drop, thus allowing an open circuit voltage of either around 4.1V or 4.7V, making it relatively suitable for trickle-charging of Ni-MH packs or possibly even li-ion cells. When used with a Vcc of 3.3V, the voltages were 2.4V and 3.1V respectively and the short-circuit currents ranged from 0.15mA to 0.27mA, perhaps more suitable for use with supercapacitors. Trickle charging at a Vcc of 1.8V was possible, but the delivered voltages were insufficient to charge the storage to an operational voltage level.

Overall, it seems the MAX31343 is a very competitive RTC compared to the others on the market on capabilities, specifications, footprint and price. Testing has borne out that it is highly accurate over the long term, with some deviations over the short term, and current consumptions are as expected from the datasheet specifications. The documentation is comprehensive, and the design and construction of the evaluation kit is excellent.

53 Views
Tags: timing, maxim, chip, ic, review, roadtest, time, rtc, accuracy, mems, oscillator, clock, resonator, tested, real-time clock, maxim integrated, integrated circuit, backup battery, trickle charging

Average User Rating

★★★★★

(0 ratings)

Your Rating:

★★★★★

0 Comments

Content | Participate | Learn | Discuss | Build | Find Products | Store

About Us | Feedback & Support | FAQs | Terms of Use | Privacy Policy | Cookies | Sitemap

Follow element14

Powered by jive