## Introduction to RADIAN Protocol

The **RADIAN protocol** is a low-power wireless communication scheme developed around 1998 by a European metering user group (including utilities and manufacturers like EDF, GDF, Itron, Sontex, etc.)[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS). It was designed for Automatic Meter Reading (AMR) across water, gas, heat, and electricity meters. Itron’s EverBlu system adopted RADIAN for its local (433 MHz) network layer[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS), and other products (e.g. Itron AnyQuest, Sontex Supercom/Superstatic) implemented it as well. RADIAN operates in the 433 MHz ISM band (around 433.82 MHz) using frequency-shift keying (FSK) modulation at a data rate of ~2.4 kbps[[2]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc). This is essentially equivalent to Wireless M-Bus “Mode F” (sub-GHz, two-way walk-by mode) in terms of RF characteristics[[3]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=of%20cryptographic%20protection%20was%20confirmed,Le). The protocol enables **bidirectional** communication between a handheld or drive-by reader (master) and a meter radio module (slave) for on-demand reads. Notably, in its common 433 MHz form, RADIAN does *not* employ encryption or authentication – meter data is sent “in the clear” once the framing is understood[[4]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Importantly%2C%20the%20RADIAN%20protocol%20frames,early%20investigators%20who%20compared%20it). This openness (even described in marketing as “the most open radio protocol” for metering) makes it feasible for third parties to intercept or integrate with their own devices, but also means security relies on obscurity and limited access (newer implementations have added encryption – discussed later).

## Physical Layer & Timing

RADIAN uses FSK modulation with narrowband channels in the 433 MHz SRD band, typically at 2.4 kbps data rate and about 10 mW transmit power[[2]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc)[[5]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Donn%C3%A9es%20techniques%20Communication%20bidirectionnelle%20%3B,Bus%20permet%20de%20transmettre%20des). Each meter’s radio is battery-powered and stays in a low-power sleep state most of the time to preserve its ~10–15 year battery life. **Wake-up and polling:** To communicate, a reader must send a long preamble to wake the meter’s receiver. In practice, this is achieved by transmitting a **wake-up pattern** (alternating 0/1 bits) for a duration on the order of 2–3 seconds before the actual data frame[[6]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=,est%20sp%C3%A9cifique%20%C3%A0%20chaque%20compteur). This “WUP” tone (around 1.2 kHz bit toggling) primes the meter to listen for a valid packet. Meters only wake and listen during configured time windows – typically during daytime/business hours to further save power. For example, many Itron/Sontex modules are only responsive from about **06:00 to 18:00 or 20:00** each day[[7]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Your%20meter%20may%20be%20configured,avoid%20confusion%20and%20unnecessary%20troubleshooting)[[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques), remaining asleep overnight. (The exact wake window can be factory-set; one Sontex document notes 06:00–20:00 daily availability for RADIAN mode[[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques), and community findings in the UK showed 08:00–16:00 in some cases[[7]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Your%20meter%20may%20be%20configured,avoid%20confusion%20and%20unnecessary%20troubleshooting).) A reader attempting to poll outside those hours will simply get no response. Within the window, the reader sends the wake-up, then a request frame; if the meter hears its own address in the frame, it will transmit a response. Communication range is on the order of a few hundred meters with a good antenna (300 m or more line-of-sight)[[9]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=assistant,your%20property%20or%20just%20outside) – suitable for “walk-by” or “drive-by” meter reading. The RF parameters (FSK deviance, sync word, etc.) closely mirror wireless M-Bus mode F2 settings, so hardware like the TI CC1101 (which supports FSK and configurable data rates) is commonly used by integrators. Simpler ASK/OOK 433 MHz receivers cannot decode RADIAN[[10]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=RADIAN%20protocol%20to%20read%20their,source%20project) because of the different modulation and data rate.

## Frame Structure

Once the meter is awake, the actual RADIAN **frames** are exchanged. The packet structure is very similar to the standard EN13757-4 (wireless M-Bus) frame format. Each frame consists of:

* **Preamble/Sync:** A specific bit pattern to synchronize (often a sequence of 0x55 bytes after the long wake-up, similar to WMBus). This is followed by a sync word that marks the start of the frame data. (In community implementations, the CC1101 is configured with the expected sync word to delineate frames – details match WMBus mode F defaults[[11]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon%5D%20www.lamaisonsimon.fr%20%29%20%28maison2%3Acompteur_d_eau%3Acompteur_d_eau%20,engineered%20solution%20%E2%80%93).)
* **Length and Control:** The first byte (Octet 0) is typically a length L field, and the second byte (Octet 1) a control C field[[12]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Trame%20du%20hand%20recorder%3A). The length may specify the number of bytes following (up to and including CRC), and the control byte contains flags for frame type and direction. In wireless M-Bus, for example, different C values distinguish a request vs. a response frame (and whether a reply is expected) – RADIAN uses the same conventions. (Industry documents indicate typical C codes like 0x2X for a response with user data, etc.[[13]](https://support.elvaco.com/hc/en-us/articles/17393384692381-Administration#:~:text=Administration%20,Bus%20slaves%20the).)
* **Addresses:** Next come two address fields. RADIAN carries both the **destination address** and **source address** in each frame (each is 6 bytes)[[14]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%200%20%C3%A0%201%3A%20%3F%3F%3F%3F,mais%20avec%20quel%20polyn%C3%B4me)[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). This is effectively the M-Bus addressing scheme: a 6-byte address can include a meter ID and other info. In practice, the 6-byte address comprises a combination of the meter’s serial number and other identifiers. For Itron EverBlu/Cyble modules, the address is derived from the meter label: for example, a meter labeled with year “22” and serial “0828979” would use 0x22 (year) and 0x0828979 (serial) as part of its address (leading zeros omitted)[[16]](https://github.com/hallard/everblu-meters-esp32#:~:text=Image%3A%20Cyble%20Meter%20Label)[[17]](https://forum.hacf.fr/t/recuperer-donnees-du-compteur-hrl-c-g2/41709#:~:text=). Each meter manufacturer in the original RADIAN group likely had an assigned manufacturer code, but in some implementations the manufacturer ID is implicit or embedded in the 6-byte address. (Sontex documentation notes that the “radio address corresponds to the serial number” of the device[[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques)[[18]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=L%E2%80%99adresse%20radio%20est%20affich%C3%A9e%20dans,Donn%C3%A9es%20techniques), implying the 6-byte address is basically the unique serial/ID and the receiving system knows the manufacturer context.) In a frame, **Octets 2–7** are the address of the **destination** (who the frame is intended for), and **Octets 8–13** are the address of the **source**[[19]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=En%20fait%20c%27est%20plut%C3%B4t%3A%20Octet,%C3%A0%207%3A%20Adresse%20du%20destinataire). For instance, in a reader’s request frame, Octets 2–7 contain the meter’s ID, and Octets 8–13 contain the handheld unit’s ID (or some ID of the reading device). In the meter’s response, these are swapped. Including both addresses allows both endpoints to confirm the intended counterpart (and is useful in networks with repeaters or multiple devices in range). Each address is 48 bits, typically enough to be globally unique per meter.
* **Control Information & Payload:** After the addresses, there are one or more bytes that serve as a **control information (CI) field or command code**, followed by the data payload. In the simple read exchange, the CI might indicate a “read request” or the type of data to be sent. For example, community captures showed two constant bytes (0x00 0x28) in this position for the basic exchange[[20]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=au%20fabricant,mais%20avec%20quel%20polyn%C3%B4me). It’s likely that one of these is a CI indicating a general data request/response (0x28 is known in M-Bus as a response telegram identifier in some modes) and the other could be a status or segmentation field. In any case, these bytes tell the receiver what to expect in the payload. The **payload data** (if any) comes next. The payload length can vary: a request frame often has little or no payload beyond the CI (since it’s just a command), whereas a response frame from a meter contains the consumption data and possibly additional records. In a decoded EverBlu meter’s reply, **Octet 14 onward carried the data**: one capture showed the data running from Octet 14 up to Octet 121 (over 100 bytes of data) for a full consumption report[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). We discuss the content of this data in the next section.
* **CRC Check:** The frame concludes with a **CRC** for error-checking. RADIAN uses a 16-bit CRC (as is standard in wM-Bus) computed over the frame header and payload. The last two bytes of the frame are this CRC16 value[[21]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%2014%20et%2015%3A%20,mais%20avec%20quel%20polyn%C3%B4me)[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). Community developers had to determine the exact polynomial; it was eventually confirmed to match the CRC used in WMBus (poly 0x3D65, initial 0xFFFF, as per EN13757) – ensuring interoperability with standard receivers. If the CRC doesn’t validate, the frame is discarded.

**Frame example:** The Maison Simon Wiki provides a breakdown of a real **handheld request frame**. In that example, bytes 0–1 were undetermined (length/control), bytes 2–7 were the meter’s address, bytes 8–13 were the reader’s address, bytes 14–15 were 0x00 0x28 (likely indicating a read command), byte 16 was an unspecified byte (possibly a sequence number or reserved), and bytes 17–18 were the CRC[[12]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Trame%20du%20hand%20recorder%3A). The meter’s **first response frame** (an acknowledgment) had: bytes 0–1 header, 2–7 meter’s address, 8–13 reader’s address, 14–15 0x00 0x28 again, and 16–17 CRC[[22]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Premi%C3%A8re%20r%C3%A9ponse%20du%20compteur%20avec,mais%20avec%20quel%20polyn%C3%B4me). A subsequent **data frame** from the meter showed bytes 2–7 meter address, 8–13 reader address, then 14–121 data payload, and 122–123 CRC[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). This illustrates the consistent structure: header, dest/src addresses, CI/command, data, CRC. Essentially, it’s a classic meter bus frame layout wrapped in a long wake-up preamble at the physical layer.

## Command Types and Handshake

RADIAN is a *master-slave* protocol: the reading device (master) initiates all communication, and the meter (slave) only transmits upon receiving a valid request addressed to it. The typical sequence for a read operation involves a **three-step handshake**:

1. **Master → Meter: “Read Request” (Polling Command).** After the wake-up tone, the master sends a frame instructing a specific meter to send its data. This frame contains the meter’s address as the destination and the master’s ID as source, plus a command code. In Wireless M-Bus terms, this is analogous to a SND-NKE or REQ-UD2 packet depending on implementation. In RADIAN, the request is often a short frame (besides addresses) – for example, the code 0x0028 observed in the request likely tells the meter “prepare for data transmission”[[14]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%200%20%C3%A0%201%3A%20%3F%3F%3F%3F,mais%20avec%20quel%20polyn%C3%B4me). The meter will only respond if the address matches (there’s no broad “broadcast”; each meter wakes and checks if the poll is for them).
2. **Meter → Master: Acknowledgment (Wake-up Ack).** Upon receiving the request, the meter typically sends a quick acknowledgment frame. Community logs indicate the meter’s first reply contains no consumption data, only its address and the master’s address (mirroring what was sent) and possibly a confirmation code[[22]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Premi%C3%A8re%20r%C3%A9ponse%20du%20compteur%20avec,mais%20avec%20quel%20polyn%C3%B4me). This can be seen as the meter saying “Okay, I’m awake and I heard you.” It might correspond to an application acknowledgment or handshake required by the protocol. In some instances, this might be very brief or even omitted, but the evidence suggests EverBlu modules do transmit an empty response before the full data. The C field in such a frame likely indicates it’s an ACK (for example, in M-Bus a RSP-ACK or similar control). The handheld reader, upon getting this, knows the meter is about to send data.
3. **Meter → Master: Data Transmission.** Immediately after the ack (often just a few milliseconds after), the meter sends the **actual data frame** containing the meter’s readings and status. This is the longest frame in the sequence. It uses the meter’s address as source and the master’s as destination, and a C/CI indicating “response with user data.” In Wireless M-Bus conventions, this would be a RSP-UD (response, user data) frame – in fact, various C field values like 0x28 or 0x38 are defined for user data responses in different modes[[13]](https://support.elvaco.com/hc/en-us/articles/17393384692381-Administration#:~:text=Administration%20,Bus%20slaves%20the). The master will receive this frame and verify the CRC. After this, the exchange is complete. (If the master needed to acknowledge receipt, that could be another step, but in most simple RADIAN readouts the exchange ends with the meter’s data. The meter then returns to sleep mode until the next cycle or until its next scheduled wake window.)

**Known Command Types:** The primary command in RADIAN is the **read request** described above (often just to get current index and status). However, the protocol and devices support a range of command types, even if not all are publicly documented. Some known or likely command types include:

* *Read (Immediate) –* as used in walk-by reads, to retrieve current consumption and status.
* *Time sync or configuration –* In some systems, there may be commands to set the meter’s clock or configuration via RADIAN (since it’s two-way). For instance, fixed network installations could remotely configure the meter via 433 MHz before deployment of 868 MHz mode. These would use different CI codes.
* *Alarm reset or specific log retrieval –* e.g. a command to retrieve detailed historical logs or clear certain alarms. The EverBlu Pulse Enhanced spec mentions capability for 13 months of history, leak alarms, etc.[[23]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%C2%BB%20Up%20to%20181%20consumption,intelligence%20functions%3A%20%E2%80%A2%20meter%20oversized), which implies the protocol has commands to fetch those logs or get alarm status. In practice, utility handhelds might request a full profile read (which could explain why the default data frame is so large, containing interval data).

Most reverse-engineering has focused on the standard consumption read request, since that’s what the utility does periodically. The handshake described (request → ack → data) is what has been observed in the field[[12]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Trame%20du%20hand%20recorder%3A)[[22]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Premi%C3%A8re%20r%C3%A9ponse%20du%20compteur%20avec,mais%20avec%20quel%20polyn%C3%B4me). Simpler one-way transmissions (like some older “bubble-up” AMR systems) are not typical for RADIAN – it is generally **two-way**. If the meter doesn’t receive a valid request during its wake window, it remains silent (unlike, say, Wireless M-Bus mode S where the meter broadcasts regularly).

One important field in the protocol’s logic is the **“read counter”** on the meter. Every time the meter successfully sends its data, it increments an internal counter. This count is included in the payload (see next section) and serves as a measure of how many times the meter has been accessed remotely. The utility’s reading software often keeps track of how many reads *it* has done and cross-checks this counter. If there’s a discrepancy – for example, the meter’s counter is much higher than expected – the system may suspect tampering or out-of-sync reads[[24]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=%40Richard,just%20doesn%E2%80%99t%20read%20the%20data)[[25]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=My%20guess%20is%20that%20i,number%20of%20reads). In fact, there are reports (e.g. with Itron’s simpler **AnyQuest Cyble** modules) where the utility’s device refused to download data because the read counter didn’t match (suggesting someone else read the meter in between)[[24]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=%40Richard,just%20doesn%E2%80%99t%20read%20the%20data). This is effectively a *security/anti-tamper feature* of the protocol: it doesn’t stop the meter from responding, but it can alert the utility that “unscheduled” reads occurred. In the EverBlu Cyble Enhanced, the read counter is known to wrap at 255 back to 0[[26]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=the%20read%20counter%20can%20be,1)[[27]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,1%20RSSI%20%3A%20227) (one project observed a jump from 255 to 1 on the next read). Hobbyists take care not to poll too frequently to avoid raising this count rapidly – many community firmwares default to one read per day or week[[28]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Not%20sure%20if%20you%20have,you%20haven%E2%80%99t). (It’s worth noting this design is a bit short-sighted, as it penalizes customers who might legitimately want to monitor their own usage. But it’s in place likely to prevent constant querying that could kill the battery or to flag potential fraud.)

## Payload Layout & Data Fields

The most important part of the RADIAN protocol is the **payload** of the meter’s data frame, which encodes the meter readings and various status information. Through reverse engineering and comparison with meter specifications, the community has mapped out many of these fields. In an EverBlu Cyble Enhanced water meter’s standard response, the key fields include:

* **Meter Identification:** The meter’s unique ID or serial number is typically reiterated in the payload (for example, for data integrity or in case frames get mixed up). Often this could be the first part of the payload. However, since the address already contains the ID, this might not be repeated in plain text – it depends on whether the application layer is using a higher-level data structure. Some implementations might include a manufacturer ID or device type as part of the payload as well. In general, the address fields already identify the meter, so the payload may skip an explicit ID to save bytes.
* **Cumulative Consumption:** This is the current meter reading (e.g. total volume of water). The value is usually given in **liters** for water meters (or cubic meters could be scaled – but in EverBlu’s case it reports liters). In a decoded example, a meter returned a value of **635633 L** used[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00). The data is likely a 4-byte or 5-byte number. Many metering protocols use BCD (binary-coded decimal) for such values, but it could also be a binary integer. Given the 6-digit nature of that example, it might be a 3-byte BCD (e.g. “0635633” BCD) or a 4-byte binary. Regardless, this field represents the total consumption reading on the meter’s register. Units are not explicitly sent but are understood from the device type (for gas it might be cubic meters or similar).
* **Battery Status (Life Remaining):** EverBlu Enhanced radios report the remaining battery life in months[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00). In the example, **107 months** was in the payload (meaning roughly 8.9 years remaining)[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00). This suggests one byte (or two) is dedicated to battery info. It could be encoded as number of months or as a percentage. Given 107 months is oddly specific, it’s likely an *estimated remaining life*. The device probably starts at e.g. 180 months (15 years) and counts down. Other RADIAN devices might instead report battery voltage or a low-battery flag – but for Itron water modules, a month estimate is provided, which is quite user-friendly.
* **Read Counter:** As discussed, the meter keeps a count of read operations. This counter value is part of the payload. For example, the meter showed “Counter: 255” then “Counter: 1” after an overflow[[30][31]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00). The counter is likely a single byte (0–255) value. Each successful read increments it by 1 (modulo 256). The utility’s handheld might ignore data if this number is unexpected. This field basically acts like a sequence number for meter reads.
* **Time-of-Use Window:** Many RADIAN modules embed their configured wake/sleep schedule in the data. In EverBlu’s case, two bytes in the payload give the daily **start hour and end hour** when the radio is active[[32]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Counter%20%3A%20255%20RSSI%20%3A,06%3A00%20Time%20end%20%3A%2018%3A00)[[33]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=RSSI%20%3A%20227%20Time%20start,06%3A00%20Time%20end%20%3A%2018%3A00). For instance, the data showed **Time Start: 06:00, Time End: 18:00** for a UK meter[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00)[[31]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00). Indeed, these matched the known utility schedule (6 AM to 6 PM active). Having this in the payload is useful for diagnostics – the reader device can know if it needs to adjust read times. It’s likely encoded as two bytes (hour start, hour end in 24h format). Some devices might even include the days of week it’s active (some modules allow Monday–Friday only, etc.), though in our case it appears to be daily. The community firmware now even exposes these times to Home Assistant for insight.
* **Status Flags / Alarms:** RADIAN payloads also carry various flags that indicate meter or sensor status. According to product literature, EverBlu/AnyQuest modules can detect conditions like **leaks**, **reverse flow**, **meter tamper** (removal or magnetic interference), **battery end-of-life**, and flow anomalies (overspeed or underspeed)[[23]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%C2%BB%20Up%20to%20181%20consumption,intelligence%20functions%3A%20%E2%80%A2%20meter%20oversized). These would be encoded as bit flags within one or more bytes of the payload. For example, a specific bit might mean “leak alarm active” and another “tamper alarm.” In addition, **date stamps for critical alarms** may be stored (e.g. the last time a leak was detected)[[34]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%E2%80%A2%20meter%20oversized%20%E2%80%A2%20meter,critical%20alarms%20are%20date%20stamped), though these might not be fully transmitted in a simple read – they could require a special read of the meter’s log. Community decoders have noted at least one byte that represents an alarm/status bitmap. In practice, when everything is normal, these bits might all be zero. If a condition is triggered (say, continuous flow for 24h might set a leak flag), the corresponding bit would be 1. Some DIY decoders may not initially parse all these, but as documentation (or trial-and-error) reveals the mapping, one could interpret them. For instance, if a user notices a particular byte goes from 0x00 to 0x04 and later finds that 0x04 corresponds to “leak alarm,” they can map it accordingly.
* **Historical Consumption Records:** One striking aspect of the EverBlu Enhanced data frame is its length – over 100 bytes of data were observed[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). This is far more than needed for just the current index and a few flags. It strongly suggests that the meter is downloading a **history log** of consumption to the reader. Indeed, Itron’s spec mentions up to *181 intervals* and *13 months* of history available in the module[[23]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%C2%BB%20Up%20to%20181%20consumption,intelligence%20functions%3A%20%E2%80%A2%20meter%20oversized). It’s likely that the default response includes a block of recent interval data. For example, it might include the last 13 monthly readings, or perhaps daily readings for the last X days. Another clue: the community notes wondered if Octets 14–121 were “indexes of the last months”[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). If each monthly index (consumption at month-end) was, say, 4 bytes, then 13 months × 4 bytes = 52 bytes, which could be part of that segment. There might also be hourly/daily intervals: 181 intervals could mean, for instance, 181 days (~6 months) of daily readings. It’s not confirmed publicly, but given the data length and product capability, the payload likely contains a sequence of timestamped readings. Reverse-engineers who have compared multiple reads might see patterns (like a repeating structure every few bytes). In any case, this bulk data is probably organized in a known format (possibly following the standard M-Bus DIF/VIF data record format for multiple records). Each record might consist of a data descriptor (type of value and unit) followed by the value. For example, it could be a block of 13 records: “Monthly volume [m³] for month 1, value X; Monthly volume for month 2, value Y; …” etc., or a series of interval counts. Without official docs, the community focuses on the primary fields (current total, battery, etc.), but the rest is likely historical usage that a utility software would upload into their database.
* **Optional Fields:** Depending on device, other data might be present. Some meters could include a temperature (for heat meters), or a firmware version, or a module ID. Sontex heat meters, for instance, might send flow temperature, return temperature, and energy reading – but those are device-specific application data on top of the RADIAN transport. The protocol itself is agnostic to what the “data” is, it just carries a blob of bytes which the higher-level (application layer) defines. In our water meter context, the data is all about volume and status.

To summarize, the RADIAN payload contains a **self-contained report** of the meter’s state at the time of reading. Because encryption was not used in older implementations, anyone who knows the structure can extract meaningful values. Early community investigators confirmed that by decoding frames and matching them to known meter readings (for example, comparing the meter’s mechanical dial or utility bill to the decoded number in the frame)[[35]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=whatsoever%E2%80%9D%20on%20the%20RF%20transmissions,in%20decoding%20was%20therefore%20not). They found that everything – from the meter ID to usage count to flags – was right there in plaintext, just needed the proper byte parsing. This lack of cryptographic protection significantly lowered the barrier to entry for hobbyists.

## Example Decoded Frames

Several real-world examples illustrate the RADIAN protocol in action:

* **EverBlu Cyble Enhanced Water Meter Readout:** One user integrated an ESP32 + CC1101 transceiver to poll their Itron water meter and publish the data to Home Assistant. On a successful read, the system logged values such as **“Liters: 635633”**, **“Battery: 107 months”**, **“Counter: 255”**, **“Time Start: 06:00”**, **“Time End: 18:00”**[[30][31]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00). These correspond exactly to the fields discussed: 635,633 L used, about 9 years of battery remaining, the meter had been read 255 times, and it wakes daily between 6 AM and 6 PM. Shortly after, a second reading showed **Counter: 1** (since it rolled over after hitting 255)[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00)[[31]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00). All these values came from decoding the single response frame from the meter. The user also noted the RSSI (signal strength) of the meter’s reply, which was obtainable from the radio module (not from the RADIAN payload itself, but as meta-data). The **raw frame** in hex was also captured by some developers – by aligning it with known values, they identified which bytes represented the liters, which the battery, etc. For instance, one might see (hypothetically) a sequence in the payload like ... 0x33 0x56 0x63 ... that, when interpreted as BCD, gives 635633. Or a byte 0x6B (107 in decimal) representing battery months. This example confirmed the community’s understanding of the payload layout[[35]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=whatsoever%E2%80%9D%20on%20the%20RF%20transmissions,in%20decoding%20was%20therefore%20not).
* **Maison Simon Wiki Log (Unknown Meter):** The French DIY group captured the exchange between a utility’s **handheld reader** and a **water meter** (presumably EverBlu or compatible). They posted an Excel sheet of the decoded bytes[[36]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=J%27ai%20rapidement%20regard%C3%A9%20le%20fichier,voici%20ce%20que%20j%27ai%20compris). In the **handheld’s request**, the bytes included an address that encodes the manufacturer. They noticed that the emitter’s address portion “might contain an ID related to the manufacturer”[[37]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%200%20%C3%A0%201%3A%20%3F%3F%3F%3F,mais%20avec%20quel%20polyn%C3%B4me) (for example, an Itron device might have a different prefix than a Sontex device). The handheld’s frame ended in 00 28 … CRC as noted. Then the **meter’s first response** was a short frame (“trame vide” meaning no data) which simply had the meter’s address, the reader’s address, the same 00 28 bytes, and a CRC[[22]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Premi%C3%A8re%20r%C3%A9ponse%20du%20compteur%20avec,mais%20avec%20quel%20polyn%C3%B4me). The **second response** was long: meter’s address, reader’s address, and a large data block from octet 14 onward[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). They hypothesized this block contained monthly indexes or similar historical data. While they did not fully decode every byte publicly, the structure matched the expected format (and no encryption was apparent since recognizable patterns could be found).
* **Sontex Heat Meter (Superstatic 749 with RADIAN 0):** While we don’t have the exact byte dump in the text above, the Sontex documentation indicates what one would see if decoding such a meter. The radio address equals the serial number[[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques)[[18]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=L%E2%80%99adresse%20radio%20est%20affich%C3%A9e%20dans,Donn%C3%A9es%20techniques), so one would find that 6-byte address in the frame. The data payload would include the **energy reading** (for heat meter, in kWh or MJ), the volume, temperatures, etc., along with a similar structure of status flags. Sontex units, however, often come with encryption enabled. For instance, a Sontex 566 using “Protocol: Radian 0” was noted to have **AES-128 encryption** on its payload[[38]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,did%20not%20use%20encryption). If you captured such a frame, the bytes after the header would look like random data (ciphertext), and you would not be able to interpret consumption values without the key. So a reverse-engineered *unencrypted* example exists mainly for the water meter case. For an encrypted Sontex, the frame format is the same (addresses, etc.), but the data portion is effectively an encrypted block. The presence of encryption is usually indicated by the CI or a flag in the frame (e.g., a bit might signify “data encrypted”). One community member noted that newer Sontex frames they saw could not be decoded, confirming encryption was active[[39]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=bidirectional%20433,did%20not%20use%20encryption).

Overall, these examples show that with the proper hardware and knowing the meter’s ID, one can trigger a RADIAN exchange and obtain detailed meter information. The success of projects like the ESP32-based reader (which publishes to MQTT) demonstrates that every field from liters to tamper alarms can be parsed from the payload in real-time[[40]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Sensor%20MQTT%20Topic%20Description%20,value%20converted%20to%20a%20percentage)[[41]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,Third). The key was figuring out the packet structure, which the community accomplished by capturing actual meter reader sessions and comparing them to known standards[[42]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon,how%20meter%20readings%20and%20serial).

## Variations & Security in Different Versions

RADIAN protocol has its roots in a multi-vendor collaboration, so while the basics are consistent, there are **variations and evolutions** to be aware of:

* **Itron EverBlu vs. Itron AnyQuest:** Itron produced both the EverBlu Cyble Enhanced (often used in fixed network or walk-by) and the AnyQuest Cyble modules. Both use the RADIAN protocol at 433 MHz and are cross-compatible at the radio level[[43]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=network%20reading%20systems,the%20open%20RADIAN%20radio%20protocol). The main difference is functionality: EverBlu “Enhanced” modules can store more data and can work in both fixed network (868 MHz) and mobile (433 MHz) modes, whereas AnyQuest (often “Basic”) modules might only support mobile reads and fewer features. From a protocol perspective, an AnyQuest module will still respond to the same RADIAN frames. However, as noted by users, the simpler AnyQuest may enforce the read-counter tamper protection more strictly[[24]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=%40Richard,just%20doesn%E2%80%99t%20read%20the%20data)[[44]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Good%20to%20hear%20that,without%20any%20issue%2C%20every%20day). The **counter mismatch issue** where a meter becomes “unreadable” to the utility after too many third-party reads was reported first with AnyQuest modules, and later with EverBlu as well[[45]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Are%20the%20meters%20there%20connected,company%20had%20the%20same%20issue)[[25]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=My%20guess%20is%20that%20i,number%20of%20reads). This suggests the logic is present in all Itron RADIAN meters. It’s not a separate protocol version per se, but a *behavioral quirk* to keep in mind. Practically, integrators now adjust their reading frequency (e.g. monthly or weekly instead of daily) to try to stay under the radar. There isn’t an alternate command to read without incrementing the counter – every poll increments it. So, managing this is an out-of-band issue.
* **Sontex and “Radian 0”:** Sontex, a Swiss manufacturer, was part of the original RADIAN group and refers to their 433 MHz protocol as *Radian 0*. It is essentially the same protocol. For example, a Sontex Supercom/749 heat meter’s 433 MHz module uses RADIAN 0 (433.82 MHz FSK) and is reachable 365 days/yr, 06:00–20:00, exactly as an Itron would be[[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques)[[46]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Gli%20apparecchi%20con%20questa%20interfaccia,di%20serie%20del%20Superstatic%20749). They even explicitly state the radio address equals the serial number, aligning with the addressing scheme. The big difference: Sontex often **enables encryption**. In technical terms, they might be using the mode where the payload is encrypted with AES-128 (similar to Wireless M-Bus mode 5 or OMS security). The Sontex 566 heat cost allocator, for instance, was found to output encrypted data under the RADIAN protocol[[38]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,did%20not%20use%20encryption). This means that although a Sontex device will happily exchange frames with a RADIAN-compatible reader (same preamble, same framing), the content can’t be understood without the key. Typically, the key is set by the utility or at the factory, and isn’t publicly known. In contrast, the Itron water modules by default did **not** encrypt the 433 MHz data[[4]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Importantly%2C%20the%20RADIAN%20protocol%20frames,early%20investigators%20who%20compared%20it) – likely because they viewed the short range and proprietary nature as sufficient deterrent, and because implementing encryption in early 2010s might have been seen as unnecessary complexity. Today, however, as privacy and security concerns grow, newer versions could incorporate encryption. It’s wise to check: if one captures frames that seem valid (correct CRC) but contain no sensible values, that could indicate encryption is ON. Sontex documentation even allows ordering a meter with an OMS (Open Metering System) wireless M-Bus mode instead of Radian[[47]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=L%E2%80%99interface%20de%20communication%20Radio%20Wireless,sera%20ajout%C3%A9e%20avec%20la%20documentation)[[48]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Donn%C3%A9es%20technique%20Communication%20unidirectionnelle%2C%20Fr%C3%A9quence,Mode%20d%E2%80%98encryption%205%20ou%207) – highlighting that Radian was a pre-OMS approach. “Radian 0” doesn’t inherently mean unencrypted – it’s just the baseline protocol.
* **868 MHz EverBlu (Fixed Network mode):** Itron EverBlu systems have a higher-frequency mode at 868 MHz (in Europe) for fixed network AMI. This is not exactly the same as 433 MHz RADIAN, but it’s related. In practice, the 868 MHz mode behaves more like standard Wireless M-Bus mode C or T, where the meter sends a daily (or more frequent) telegram that can be picked up by nearby base stations[[49]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=%2A%20868%C2%A0MHz%20Fixed,unencrypted%20for%20simplicity%2C%20relying%20on). These transmissions *are* **encrypted** in virtually all cases[[49]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=%2A%20868%C2%A0MHz%20Fixed,unencrypted%20for%20simplicity%2C%20relying%20on). They typically use AES-128 with a network key, following OMS standards. The payload content at 868 MHz is usually a subset of data (like daily usage and meter ID) rather than the full log, since it’s often a one-way burst. It’s important not to confuse the two: a meter in EverBlu fixed network mode will send 868 MHz frames that an SDR might pick up, but unless you have the key, you won’t decode it (you’ll see a scramble of bytes). The same meter, when polled at 433 MHz with the right wake-up, will respond in RADIAN clear-text mode – often, these devices support both modes (the product sheet calls them “bi-frequency, bi-power” devices[[50]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=6,NAN%20%3D%20neighborhood%20area%20network)). The fixed network typically uses a different “CI” and frame format (often per EN13757-4 for mode C). Some hobbyists have tried to intercept 868 MHz frames and given up due to encryption, focusing instead on polling via 433 MHz which is feasible.
* **Other Vendors:** Apart from Itron and Sontex, other companies like Aquametro, Viterra (now part of Diehl), and Schlumberger were involved in RADIAN’s design[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS). It’s likely that some of their legacy products (e.g. old Schlumberger gas modules, or Aquametro heat meters) were RADIAN-compatible. Diehl (Viterra) for instance moved to their own protocols and OMS in recent years, so RADIAN might be found mostly in devices from the 2000s–2010s. Each vendor might have had slight differences in data encoding (for example, maybe different units or additional fields), but since the aim was compatibility, any RADIAN reader could theoretically get a basic reading from any RADIAN meter. The Open Meter project documentation confirms that EverBlu meters could be read either by the EverBlu fixed network or by “walk-by collection compatible with Radian protocol” for interoperability[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS). This dual-mode capability suggests that RADIAN was effectively a common denominator.

In summary, **RADIAN’s core frame format and handshake are consistent across implementations**, but the *content* and *security* can vary. Older water/gas modules: clear data, straightforward to decode. Newer or more security-conscious devices: data encrypted (requires keys, not publicly accessible). The existence of an open standard (OMS) eventually overtook RADIAN in popularity for new deployments, but millions of RADIAN-based modules are still in the field[[51]](http://www.lamaisonsimon.fr/wiki/lib/exe/fetch.php?media=maison2:compteur_d_eau:open-meter_wp2_d2.1_part3_v1.0.pdf#:~:text=About%201%20million%20radio%20modules,These%20meters%20have%20been). Integrators should thus be mindful: if you get gibberish data, you might be dealing with an encrypted variant or a different mode (e.g. an OMS mode instead of raw Radian).

## Community Integrations & Decoding Efforts

One of the reasons we know so much about RADIAN’s payload today is due to dedicated work by hobbyists and independent researchers. Early on (mid-2010s), individuals like the contributors to *La Maison Simon* wiki in France began capturing and deciphering EverBlu meter signals[[42]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon,how%20meter%20readings%20and%20serial). Using software-defined radios (SDRs) to eavesdrop on actual utility meter reading sessions, they were able to record the waveforms and eventually extract binary data. By analyzing these alongside known meter info (serial numbers, consumption from a recent bill, etc.), they cracked the encoding. They discovered, for instance, that RADIAN frames were very similar to wireless M-Bus frames, which gave hints on where the length, addresses, and CRC should be[[3]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=of%20cryptographic%20protection%20was%20confirmed,Le)[[11]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon%5D%20www.lamaisonsimon.fr%20%29%20%28maison2%3Acompteur_d_eau%3Acompteur_d_eau%20,engineered%20solution%20%E2%80%93). They also leveraged the fact that no encryption was present, so plain-text patterns (like a serial number or time schedule) could sometimes be spotted in the hex dumps.

As knowledge spread, open-source projects emerged. A notable one is the **everblu-meters** project (ESP8266/ESP32 with CC1101) originally by developers Fred and Neutrinus around 2016–2018[[52]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=the%20meter%20and%20decode%20the,time%20water%20consumption), later improved by others (psykokwak, Richard.Huish, etc.). This project essentially implements the full protocol: it sends the wake-up, issues the properly formatted “master request” to a given meter, listens for the response, parses the payload, and then outputs the data (e.g. via MQTT). The code is freely available and has been used in Home Assistant and Domoticz integrations. For example, Richard Huish’s fork added Home Assistant MQTT autodiscovery and exposed fields like RSSI and the meter’s wake/sleep times as sensors[[53]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Sharing%20my%20project%3A%20Fetch%20water%2Fgas,Home%20Assistant%20via%20MQTT%20AutoDiscovery)[[54]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Update%20%E2%80%93%2017th%20April%202025,Vorhees%20%20%2020%20%40AltF4me). The project confirms the technical details: it mentions the **RADIAN protocol at 433.82 MHz FSK** and that *“upon receiving the correct preamble, the meter transmits consumption data (ID, litres used, etc.)”*[[2][55]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc). It also notes these meters do **not use encryption** in most configurations[[56]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Encryption), which was a green light for hobbyists.

To build a DIY reader, common hardware includes a simple microcontroller (ESP8266, ESP32, Arduino, or Raspberry Pi) and a CC1101 transceiver module. The CC1101 is critical because it can be configured to the exact modulation and bitrate of RADIAN. Community guides provide CC1101 register settings for 433.82 MHz and 2.4 kbps, matching the expected deviation and bandwidth (often referencing TI’s Wireless M-Bus mode F settings)[[10]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=RADIAN%20protocol%20to%20read%20their,source%20project). Some have also used RTL-SDRs with custom demodulators (e.g. using GNU Radio) to decode frames in software, but a transceiver like CC1101 can do packet handling and is easier for real-time use.

One challenge encountered was frequency calibration – not every meter is on *exactly* 433.820000 MHz. The devices might be off by a few kHz due to component tolerance. Hobbyists have implemented frequency scan routines to find the precise frequency the meter is responding on[[57]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,private.h)[[58]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Your%20transceiver%20module%20may%20not,rerun%20the%20Frequency%20Discovery%20code). It often ends up in the 433.82–433.89 MHz range. Once locked in, the communication is stable.

The community has also shared insights like the need for transmitting the wake-up for a sufficient time, using Manchester encoding or not (Mode F frames are typically 2-FSK with Manchester coding in wireless M-Bus F2 mode). Indeed, the separation bits observed (3 separator bits between bytes when the reader talks, 4 when the meter talks) in the Maison Simon analysis[[59]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=nous%20n%27avons%20pas%20identifi%C3%A9%20de,Pic%2C%2004%2F2016%20Ok%2C%20%C3%A7a%20roule)suggest some form of symbol encoding or timing gap – possibly an artifact of how the frames were captured. In practice, using the CC1101 in synchronous mode bypasses having to manually deal with those details – the transceiver can output a clean byte stream once configured correctly for the known preamble and sync word. The open-source implementations handle this, making it easier for newcomers.

**Integrations:** With decoding achieved, RADIAN data has been fed into home automation systems. Home Assistant users, for example, set up sensors for water consumption that update daily from their meter’s broadcast[[60]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Itron%20EverBlu%20Cyble%20Enhanced%20module,%E2%80%9Csync%2Fwake%E2%80%9D%20preamble%20and%20then%20receives)[[61]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=modules%20,a%20few%20hundred%20meters%20with). MQTT topics carry the liter count, battery, etc.[[40]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Sensor%20MQTT%20Topic%20Description%20,value%20converted%20to%20a%20percentage). This essentially gives consumers their own live meter dashboard, something not originally provided by the utility. Domoticz (another home automation platform) also saw guides on tele-reading water meters via RADIAN and logging usage, thanks to community code[[62]](https://domotique.web2diz.net/tele-releve-de-compteur-deau-avec-ajout-dans-domoticz/#:~:text=TS_len_u8%3DMake_Radian_Master_req,YY%20sur%20la%201er%20ligne). These projects underscore that the **technical information is now well-known and accessible** – no official documentation was released, but through collective reverse-engineering, all key aspects of RADIAN were uncovered and published in forums, wikis, and GitHub. As one report put it, “with modest hardware and freely available code, one can capture their own meter’s consumption data”[[63]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=ESP32%2FESP8266%20and%20433Mhz%20CC1101,improved) – a powerful example of open tech in the IoT realm.

**Caveats:** While the protocol is now “open” in practice, users should remain mindful of legal and ethical considerations. In most jurisdictions (e.g. UK/EU), listening to or querying your own meter’s unencrypted radio signals for personal use is likely permissible (433 MHz is an unlicensed band and the meters are intended to be read remotely)[[64]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Radio%20Spectrum%20Licensing%20). However, actively transmitting to a meter you do not own or without permission could be illegal. Furthermore, as discussed, excessive polling can inadvertently flag you to the utility or even lead to the meter being swapped out[[45]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Are%20the%20meters%20there%20connected,company%20had%20the%20same%20issue)[[25]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=My%20guess%20is%20that%20i,number%20of%20reads). The community generally advises reading no more than the utility would (perhaps once a day or week)[[28]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Not%20sure%20if%20you%20have,you%20haven%E2%80%99t) to avoid any issue. There have been creative suggestions like deliberately rolling the read counter back around to the expected value before the utility’s annual check[[65]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Important%20note%20on%20counter%20implications%3A,match%20that%20same%20number%20again), but these are hacks and not recommended (they would require hundreds of rapid reads and would severely drain the battery). It’s better to use the knowledge responsibly and moderately.

In conclusion, the RADIAN protocol’s **frame format** consists of a structured header (length & control), dual 6-byte addresses, command/info bytes, a data payload carrying meter readings and statuses, and a CRC16 checksum. The **byte-level encoding** follows the patterns of wireless M-Bus, with identifiable fields for usage, battery, alarms, etc., usually unencrypted in 433 MHz implementations. **Command types** revolve around a read-request/response exchange, with the standard use-case being a handheld initiating a read and the meter responding with consumption data (and possibly logs). The community has provided **real-world payload examples** that match documentation: e.g. showing liters used, a read counter, alarm flags, and historical data in the bytes[[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00)[[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC). Timing nuances like required wake-up preambles and restricted listening hours are well-understood and handled in third-party code. If multiple protocol versions or modes exist, they are largely interoperable on 433 MHz (often labeled “Radian 0”) – differences come with encryption or when the device switches to other standards (868 MHz OMS mode). Thanks to hobbyist and integrator efforts, virtually every aspect of the RADIAN packet structure has been demystified, allowing integration of legacy “proprietary” meter radios into modern open-source systems without vendor support[[66]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Notable%20open,Now%20with)[[61]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=modules%20,a%20few%20hundred%20meters%20with).

**Sources:** RADIAN protocol details were synthesized from community reverse-engineering reports[[35]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=whatsoever%E2%80%9D%20on%20the%20RF%20transmissions,in%20decoding%20was%20therefore%20not)[[12]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Trame%20du%20hand%20recorder%3A), open-source project documentation[[67]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc)[[30]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00), and manufacturer literature[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS)[[43]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=network%20reading%20systems,the%20open%20RADIAN%20radio%20protocol), as cited throughout. These provide a comprehensive view of the frame format, encoding, and usage of RADIAN on 433 MHz water/gas meter devices.

[[1]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=including%20water%2C%20gas%20electricity%20and,eg%20GPRS) [[50]](https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0#:~:text=6,NAN%20%3D%20neighborhood%20area%20network) OPEN-Meter WP2 D2.1 Part4 v1.0 | PDF | Communications Protocols | Osi Model

<https://www.scribd.com/doc/87021171/OPEN-Meter-WP2-D2-1-Part4-v1-0>

[[2]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc) [[3]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=of%20cryptographic%20protection%20was%20confirmed,Le) [[4]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Importantly%2C%20the%20RADIAN%20protocol%20frames,early%20investigators%20who%20compared%20it) [[7]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Your%20meter%20may%20be%20configured,avoid%20confusion%20and%20unnecessary%20troubleshooting) [[9]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=assistant,your%20property%20or%20just%20outside) [[10]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=RADIAN%20protocol%20to%20read%20their,source%20project) [[11]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon%5D%20www.lamaisonsimon.fr%20%29%20%28maison2%3Acompteur_d_eau%3Acompteur_d_eau%20,engineered%20solution%20%E2%80%93) [[35]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=whatsoever%E2%80%9D%20on%20the%20RF%20transmissions,in%20decoding%20was%20therefore%20not) [[38]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,did%20not%20use%20encryption) [[39]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=bidirectional%20433,did%20not%20use%20encryption) [[40]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Sensor%20MQTT%20Topic%20Description%20,value%20converted%20to%20a%20percentage) [[41]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,Third) [[42]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Simon,how%20meter%20readings%20and%20serial) [[49]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=%2A%20868%C2%A0MHz%20Fixed,unencrypted%20for%20simplicity%2C%20relying%20on) [[52]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=the%20meter%20and%20decode%20the,time%20water%20consumption) [[55]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc) [[56]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Encryption) [[57]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,private.h) [[58]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Your%20transceiver%20module%20may%20not,rerun%20the%20Frequency%20Discovery%20code) [[60]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Itron%20EverBlu%20Cyble%20Enhanced%20module,%E2%80%9Csync%2Fwake%E2%80%9D%20preamble%20and%20then%20receives) [[61]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=modules%20,a%20few%20hundred%20meters%20with) [[63]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=ESP32%2FESP8266%20and%20433Mhz%20CC1101,improved) [[64]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Radio%20Spectrum%20Licensing%20) [[66]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=Notable%20open,Now%20with) [[67]](https://github.com/genestealer/everblu-meters-esp8266-improved#:~:text=,ID%2C%20litres%20used%2C%20etc) GitHub - genestealer/everblu-meters-esp8266-improved: Fetch water usage data from Cyble meters for ESP8266 / ESP32

<https://github.com/genestealer/everblu-meters-esp8266-improved>

[[5]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Donn%C3%A9es%20techniques%20Communication%20bidirectionnelle%20%3B,Bus%20permet%20de%20transmettre%20des) [[8]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=syst%C3%A8me%20de%20rel%C3%A8ve%20%C3%A0%20dis%02tance,Donn%C3%A9es%20techniques) [[18]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=L%E2%80%99adresse%20radio%20est%20affich%C3%A9e%20dans,Donn%C3%A9es%20techniques) [[46]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Gli%20apparecchi%20con%20questa%20interfaccia,di%20serie%20del%20Superstatic%20749) [[47]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=L%E2%80%99interface%20de%20communication%20Radio%20Wireless,sera%20ajout%C3%A9e%20avec%20la%20documentation) [[48]](https://www.ter-en.com/uploads/tsk-25-2.pdf#:~:text=Donn%C3%A9es%20technique%20Communication%20unidirectionnelle%2C%20Fr%C3%A9quence,Mode%20d%E2%80%98encryption%205%20ou%207) ter-en.com

<https://www.ter-en.com/uploads/tsk-25-2.pdf>

[[6]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=,est%20sp%C3%A9cifique%20%C3%A0%20chaque%20compteur) [[12]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Trame%20du%20hand%20recorder%3A) [[14]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%200%20%C3%A0%201%3A%20%3F%3F%3F%3F,mais%20avec%20quel%20polyn%C3%B4me) [[15]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Deuxi%C3%A8me%20r%C3%A9ponse%20du%20compteur%20avec,122%20et%20123%3A%20CRC) [[19]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=En%20fait%20c%27est%20plut%C3%B4t%3A%20Octet,%C3%A0%207%3A%20Adresse%20du%20destinataire) [[20]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=au%20fabricant,mais%20avec%20quel%20polyn%C3%B4me) [[21]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%2014%20et%2015%3A%20,mais%20avec%20quel%20polyn%C3%B4me) [[22]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Premi%C3%A8re%20r%C3%A9ponse%20du%20compteur%20avec,mais%20avec%20quel%20polyn%C3%B4me) [[36]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=J%27ai%20rapidement%20regard%C3%A9%20le%20fichier,voici%20ce%20que%20j%27ai%20compris) [[37]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=Octet%200%20%C3%A0%201%3A%20%3F%3F%3F%3F,mais%20avec%20quel%20polyn%C3%B4me) [[59]](https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI#:~:text=nous%20n%27avons%20pas%20identifi%C3%A9%20de,Pic%2C%2004%2F2016%20Ok%2C%20%C3%A7a%20roule) House2 - Water - Counter - Water - Counter (The Maison Simon WIKI) | PDF | Rede de computadores | Telecomunicações

<https://pt.scribd.com/document/660504066/house2-water-counter-water-counter-The-Maison-Simon-WIKI>

[[13]](https://support.elvaco.com/hc/en-us/articles/17393384692381-Administration#:~:text=Administration%20,Bus%20slaves%20the) Administration - Elvaco

<https://support.elvaco.com/hc/en-us/articles/17393384692381-Administration>

[[16]](https://github.com/hallard/everblu-meters-esp32#:~:text=Image%3A%20Cyble%20Meter%20Label) GitHub - hallard/everblu-meters-esp32: Fetch water usage data from Cyble meters for esp8266

<https://github.com/hallard/everblu-meters-esp32>

[[17]](https://forum.hacf.fr/t/recuperer-donnees-du-compteur-hrl-c-g2/41709#:~:text=) Récupérer données du compteur HRL-c-G2 - Autres - Home Assistant Communauté Francophone

<https://forum.hacf.fr/t/recuperer-donnees-du-compteur-hrl-c-g2/41709>

[[23]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%C2%BB%20Up%20to%20181%20consumption,intelligence%20functions%3A%20%E2%80%A2%20meter%20oversized) [[34]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=%E2%80%A2%20meter%20oversized%20%E2%80%A2%20meter,critical%20alarms%20are%20date%20stamped) [[43]](https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf#:~:text=network%20reading%20systems,the%20open%20RADIAN%20radio%20protocol) hyndswater.co.nz

<https://www.hyndswater.co.nz/wp-content/uploads/W8.16-Itron-AnyQuest-Everblu-Pulse-Enhanced.pdf>

[[24]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=%40Richard,just%20doesn%E2%80%99t%20read%20the%20data) [[25]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=My%20guess%20is%20that%20i,number%20of%20reads) [[26]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=the%20read%20counter%20can%20be,1) [[27]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,1%20RSSI%20%3A%20227) [[28]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Not%20sure%20if%20you%20have,you%20haven%E2%80%99t) [[29]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Liters%20%3A%20635633%20Battery%20,06%3A00%20Time%20end%20%3A%2018%3A00) [[30]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00) [[31]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Current%20date%20,06%3A00%20Time%20end%20%3A%2018%3A00) [[32]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Counter%20%3A%20255%20RSSI%20%3A,06%3A00%20Time%20end%20%3A%2018%3A00) [[33]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=RSSI%20%3A%20227%20Time%20start,06%3A00%20Time%20end%20%3A%2018%3A00) [[44]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Good%20to%20hear%20that,without%20any%20issue%2C%20every%20day) [[45]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Are%20the%20meters%20there%20connected,company%20had%20the%20same%20issue) [[53]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Sharing%20my%20project%3A%20Fetch%20water%2Fgas,Home%20Assistant%20via%20MQTT%20AutoDiscovery) [[54]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Update%20%E2%80%93%2017th%20April%202025,Vorhees%20%20%2020%20%40AltF4me) [[65]](https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180#:~:text=Important%20note%20on%20counter%20implications%3A,match%20that%20same%20number%20again) Reading Itron EverBlu Cyble RF Enhanced Water meter with ESP32/ESP8266 and 433Mhz CC1101. Home Assistant MQTT AutoDiscovery (Now with RSSI and more) - Share your Projects! - Home Assistant Community

<https://community.home-assistant.io/t/reading-itron-everblu-cyble-rf-enhanced-water-meter-with-esp32-esp8266-and-433mhz-cc1101-home-assistant-mqtt-autodiscovery-now-with-rssi-and-more/833180>

[[51]](http://www.lamaisonsimon.fr/wiki/lib/exe/fetch.php?media=maison2:compteur_d_eau:open-meter_wp2_d2.1_part3_v1.0.pdf#:~:text=About%201%20million%20radio%20modules,These%20meters%20have%20been) [PDF] la maison Simon - OPEN meter

<http://www.lamaisonsimon.fr/wiki/lib/exe/fetch.php?media=maison2:compteur_d_eau:open-meter_wp2_d2.1_part3_v1.0.pdf>

[[62]](https://domotique.web2diz.net/tele-releve-de-compteur-deau-avec-ajout-dans-domoticz/#:~:text=TS_len_u8%3DMake_Radian_Master_req,YY%20sur%20la%201er%20ligne) Télé relevé de compteur d’eau avec ajout dans domoticz | Domotic and stupid geek stuff

<https://domotique.web2diz.net/tele-releve-de-compteur-deau-avec-ajout-dans-domoticz/>