PushStack

RANDOM PROJECTS STACK

Main menu

Somfy Smoove Origin RTS Protocol

This document describes the Somfy RTS protocol as used by the “Somfy Smoove Origin RTS”. Most information in this document is based on passive observation of the data send by the Smoove Origin RTS remote, and thus can be inaccurate or incorrect!

Changelog:

- 2014-06-25: Updated with information provided by Dirk Farin in the comments.
- 2014-09-24: Extended Control code list with info provided by Gerrit in comments. And incorporated Gerardwr’s comment about non constant key MSB.

Parameters

- **Frequency:** 433.42 Mhz
- **Modulation:** ASK/OOK
- **Encoding:** Manchester code, rising edge = 1, falling edge = 0
- **Payload Data length:** 56 bit
- **Symbol Width:** 1208 us

Frame description

Before the first frame of a button press there is some kind of wakeup pulse of 9415 us long followed by 89565 us of silence.

Frame structure:

```
                                            604 us
```

```
| 2416us | 2416us | 2416us | 2416us | 45

+---------+         +---------+         +---
+         +---------+         +---------+

|                hw. sync.              |   s
|                                       |   s
```

Where:

- **Hardware sync.:** Repeating sequence of 4 symbol width’s high(~2416 us) and 4 symbol width’s low(~2416 us). When a button is pressed the first frame has only 2 of these, repeating frames have 7
- **Software sync.:** 4550 us high, followed by 1/2 symbol width low(~604 us)
- **Inter-frame gap:** 30415 us of silence after last symbol. (Note: This means that if last bit = 0, then the line will be silent 1/2 symbol width longer)
- **Data:** Data is Manchester encoded, 1=rising edge, 0=falling edge. One bit is encoded per symbol(~1208 us)

Encoded data example:

```
|   1208 us     |
|604 us|

        +-------+       +-------+-------+       +-------+
+-------+       +-------+               +-------+       +-------+

|       1       |       1       |       0       |       0       |
```

The whole timing isn’t very critical. It is not required to send the wake up pulse and it is enough to just send one frame. Dirk Farin reports in the comments below that there is about a 15-20 percent tolerated error margin. Although what is accepted can probably depend on the receiver you are trying to talk to.

Payload Data

```
   byte
    0       1         2        3        4        5        6
|-------|--------|-------|-------|-------|-------|-------
|  key  |ctrl|cks|  Rolling Code |   Address(A0|A1|A3)
|-------|--------|-------|-------|-------|-------|-------
```

Where:

- **Key:** “Encryption Key”, Most significant 4-bit are always 0xA, Least Significant bits is a linear counter. In the Smoove Origin this counter is increased together with the rolling code. But leaving this on a constant value also works. Gerardwr notes that for some other types of remotes the MSB is not constant.
- **Ctrl:** 4-bit Control codes, this indicates the button that is pressed
- **Cks:** 4-bit Checksum.
- **Rolling Code:** 16-bit rolling code (big-endian) increased with every button press.
- **Address:** 24-bit identifier of sending device (little-endian)

The payload data is send starting with the MSB from byte 0.

Payload obfuscation

The payload data is obfuscated by doing an XOR between the byte to obfuscate and the previous obfuscated byte. In pseudo C the algorithm can be written as:

```
uint8_t frame[7];
for (i=1; i < 7; i++) {
        frame[i] = frame[i] ^ frame[i-1];
}
```

Checksum algorithm

The checksum is calculated by doing a XOR of all nibbles of the frame. To generate a checksum for a frame set the ‘cks’ field to 0 before calculating the checksum. To verify a frame check if the value returned by the checksum algorithm is equal to 0, if not the frame is corrupt.

In pseudo C the algorithm can be written as:

```
uint8_t frame[7];
```

```
for (i=0; i < 7; i++) {
        cksum = cksum ^ frame[i] ^ (frame[i] >> 4);
}
cksum = cksum & 0xf;
```

Control codes

The Smoove Origin RTS uses the following control codes:

Value	Button(s)	Description
0x1	My	Stop or move to favourite position
0x2	Up	Move up
0x3	My + Up	Set upper motor limit in initial programming mode
0x4	Down	Move down
0x5	My + Down	Set lower motor limit in initial programming mode
0x6	Up + Down	Change motor limit and initial programming mode
0x8	Prog	Used for (de-)registering remotes, see below
0x9	Sun + Flag	Enable sun and wind detector (SUN and FLAG symbol on the Telis Soliris RC)
0xA	Flag	Disable sun detector (FLAG symbol on the Telis Soliris RC)

More info on the motor programming/limits see the Somfy Motors FAQ

Addressing

The address field of a frame contains the address of the source(eg. the remote control). The device to be controlled is programmed with the addresses of the remote controls it should listen to. Looking at the addresses of multiple remote controls from the same batch suggests that the addresses are little-endian and are sequentially assigned to the devices.

Rolling code

The Rolling code is increased every time a button is pressed. The receiver only accepts frames if the rolling code is above the last received code and is not to far ahead of the last received code. This window is in the order of a 100 big.

Frame repeat

When a button is hold pressed the sender will repeat the same code. The frames are separated by the Inter-frame gap. The second and following frames have a hardware sync that consists out of 7 high/low repetitions instead of 2. The repeating frames are identical to the original, the key and rolling code is not increased.

(De-)Registering remotes

The Somfy receivers allow new remotes to be registered and old remotes to be deregistered from a receiver. The exact procedure is described in the Somfy manuals.

In short the registration process goes like this:

- Long press the ‘Prog’ button on a remote that is already registered to the device, until the blinds move shortly up and down. (TODO: how many frames is a long press?)
- Press the ‘Prog’ button on the new remote that you want to register, until the blinds move shortly up and down again. Sending the same frame 4 times, like I do with a normal button press, worked for me.

The deregistration process should be similar although I haven’t tried that yet.

As Dirk note’s in the comments below, when registering a new remote its rolling code can start at an arbitrary value. But for deregistering the rolling code must be within the receivers expected range.

Share this:

Twitter Facebook Google

★ Like

2 bloggers like this.

60 THOUGHTS ON “SOMFY SMOOVE ORIGIN RTS PROTOCOL”

Dirk Farin

— JUNE 10, 2014 AT 11:47 AM

A small fix to your description of the obfuscation algorithm: if implemented as given in your pseudo-code, the frame bytes would be overwritten by the unobfuscated bytes, while in reality each byte is computed by a pair of original input bytes.

Reply

---

pushstack

— JUNE 10, 2014 AT 1:45 PM

Yes, you are correct when it comes to de-obfuscating a frame. In that case you'll probably want a second buffer. For obfuscating a buffer the above pseudo code can be used.

Reply

Dirk Farin

— JUNE 22, 2014 AT 10:23 PM

Very nice, I got it working on a raspberry with a cheap transmission module. Actually, I originally started to decipher the protocol by myself and came to the same results as you did. I could have saved quite some time had I seen your very detailed description earlier.

Here are some additional things that I observed:
– The lower 4 bits of the key seem to always be bound to the rolling code in the original remotes, however it does not have to. Simply setting them to a constant zero also works.
– The timing is not very critial, it still worked when I was sending pulses 20% longer, or 15% shorter.
– When registering the DIY remote with its own ID, the initial rolling code can be arbitrary, but when deregistering the remote again, the rolling code has to match. So you better take care not to lose it.
– As far as patent US8189620 is concerned, I don't think that this is relevant for us because that describes sending additional data in the gap between two commands of an "old protocol". What we are talking about on this page is exactly this "old protocol" which the patent considers as prior art.

Reply

pushstack

— JUNE 25, 2014 AT 9:45 AM

Thanks for the information! You answered some questions I still had on my ToDo list but didn't have time for:).
I updated the post.

Reply

Olivier Burgard

— JULY 12, 2014 AT 12:07 PM

Hello !

Which transmission module did you use ? I can't find one with the required 433.42Mhz frequency…

Thanks,
Olivier

Reply

pushstack

— JULY 12, 2014 AT 4:58 PM

Originally I used a Aurel RX-4M50RR30SF which is actually at 433.92 Mhz. This frequency mismatch causes the range to be only a couple of meters, but this varied per blind. Currently I'm switching to a SI4010, which is tunable and has a MCU build in. In an initial test with this module was successful at a range of 10m through a wall.

Reply

John

— JULY 15, 2014 AT 7:10 AM

This is awesome, great job reverse engineering this. I looked into doing some home automation with my Somfy powered shades a year ago but didn't find information on the protocol so I gave up. I decided to start again today and am waiting for a Raspberry Pi, and found this page in the meantime

Do you have more information about how you connected the SI4010? Are you using a Raspberry Pi?

Reply

pushstack

— SEPTEMBER 24, 2014 AT 8:10 AM

For info on the SI4010 see the blog posts. I use a Raspberry Pi with the SI4010 connected to it using a FT232 USB->UART adapter. Although I could also have connected it directly to the RPI serial port.

Reply

gerardwr

— SEPTEMBER 24, 2014 AT 8:50 AM

I am currently using an RPI with a very cheap 433.92Mhz transmitter directly attached to the GPIO connector, and software based on WiringPi and a migrated Arduino sketch on the Arduino forum.

Works OK for me (at least) within 5 meters and through 2 brick walls.

Reply

John

— SEPTEMBER 24, 2014 AT 2:54 PM

Thanks, I don't have a programmer for an SI4010. That would be necessary right? I found an evaluation kit on digikey with a programmer for about $150, but given that the hacky (and easier since no programming) method of hardwiring a remote cost 70, I went with that for now.

Reply

pushstack

— SEPTEMBER 24, 2014 AT 5:14 PM

I use a standard FT232R chip + SI4010prog for programming the SI4010, and also use the FT232R for communicating with the SI4010 firmware. Total price of the component including SMA connector and removable antenna is less than $15. This setup has a clear line of sight range of more than 300M.

But I'll will write a blog post on this in the future.

Reply

John

— SEPTEMBER 25, 2014 AT 1:05 AM

@pushstack: a writeup would be awesome, thanks. I'd love to not use a secondary remote connected to my Raspberry Pi since it's so big.

Reply

John

— SEPTEMBER 24, 2014 AT 2:56 PM

gerardwr: I tried using a cheap 433.92 transmitter using the GPIO connector. However even when setting realtime priority for the process, I couldn't switch the GPIO signal fast enough for this protocol (i.e. I'd be off by 200-500us each time). How did you do this? Also do you have your code up somewhere? Thanks

Reply

Creamers

— SEPTEMBER 25, 2014 AT 5:13 PM

John, check out : pilight . org In the next upcoming release (soon) it will probably support somfy.

Reply

Ben H

— APRIL 30, 2015 AT 10:07 PM

Did anyone add somfy to pilight yet? Searching on their site, looks like not.

Reply

gerardwr

— SEPTEMBER 24, 2014 AT 3:50 PM

@john : Hi.

I am not sure if PushStack would appreciate a direct link here, so here it goes. Google for „somfy protocole” to get to a French topic on the Arduino forum. The last page contains a link to a somfy.ino file. This sketch basically works, I needed some minor adaptations for the input/port port.

Later I migrated the transmit part of the Arduino sketch to an RPi C program, using the WiringPi library for the GPIO. This performs very well, I had no problems with timing, and I use it for several RF protocols.

Leave an email address for a direct contact.

Reply

John

— SEPTEMBER 25, 2014 AT 1:05 AM

@gerardwr, I couldn’t find that information. Can you please email me at 8x35NFj54cli@meltmail.com?
Thanks

Reply

Pingback: Reversing Somfy RTS | PushStack

Creamers

— AUGUST 24, 2014 AT 2:03 PM

Whhaaw, great find. Like John I searched also alot on the internet, gave up as I also coudnt find anything about it. Please give a look at the opensource pilight.org project. It’s probably something you also like!
You write that you connect your sender as another remote to the somfy receiver, but how to you manage the rolling code? (you can contact me if you like)

Reply

pushstack

— SEPTEMBER 24, 2014 AT 7:56 AM

I manage the rolling code using a state file. The state file contains the remote it's address and the rolling code. I Use a command line program that accepts as arguments which button press to generate and the state file. My program updates the state file by increasing the rolling code and key, after sending the frames.

pilight definitely looks interesting for integrating the RTS protocol.

Reply

gerardwr

— SEPTEMBER 24, 2014 AT 8:57 AM

Same here. My command looks like "somfy 12345 4", where 12345 is the address and 4 the command.

The C program reads the current rolling code from file 12345.conf, use to for sending, and then increments the code in the .conf file.

Reply

Creamers

— OCTOBER 22, 2014 AT 5:39 PM

question: as I disconnect the shutter from the power, record the pulsetrain, enable the power, playback the pulsetrain (433,8mhz) my shutter doesnt move. I confirm the pulsestream as valid and correct (looking at the next press of the somfy to comfirm also)

What could be the issue? Receiving is no problem and matches exactly the logic, but somehow sending isn't accepted by the (pretty) new shutter. (you think this 433.4 vs 433.8 could be THE issue or should it also work?)

Any suggestions?

Reply

pushstack

— OCTOBER 25, 2014 AT 8:54 AM

I found that the 433.4 vs 433.8 MHz mismatch can be an issue with some screens. One of the screen's I tested with could only be controlled with my 433.8 MHz sender when I placed the antenna in a certain sweet spot(almost right below the left, as seen from inside, hand side of the screen.). But moving it like 15-30 cm caused it to fail. While other screens in the same building I was able to control from 10 meters distance.

So try to move around a bit or try a different antenna/antenna orientation.

Reply

gerardwr

— SEPTEMBER 2, 2014 AT 8:13 AM

Excellent info, it has enabled to decode the RF signal from my Somfy remotes using Arduino. Thanks!

Please note that the "Somfy Telis RTS 1" (vanilla color with 3 grey buttons) remotes I own do NOT have 0xA in the MSB of the 1st byte, but is seemingly random. I also have a Somfy Telis 1 PURE (with the MY center button), that HAS the 0xA in the MSB.

Regards.

Reply

Gerrit

— SEPTEMBER 21, 2014 AT 12:48 PM

Hello Pushstack,

The control codes 0x3, 0x5 and 0x6 have the following functions:
0x3 -> My && Up (pressed at the same time).
0x5 -> My && Down (pressed at the same time).
0x6 -> Up && Down (pressed at the same time).

There are also the control codes 0xA and 0x9, they are used for the sun and wind detector like the Somfy Soliris:
0x9 -> Enable sun and wind detector (SUN and FLAG symbol on the Telis Soliris RC)
0xA -> Disable sun detector (FLAG symbol on the Telis Soliris RC)

Good luck,

Gerrit

Reply

pushstack

— SEPTEMBER 24, 2014 AT 8:06 AM

Thank you very much for the information. I will update the post with it.

If I understand it correctly the My + Up and My + Down are used to configure the upper and lower engine limits. Do you also know what Up + Down means?

Reply

gerardwr

— SEPTEMBER 24, 2014 AT 8:40 AM

I found the following pdf very useful, it describes the programming of the Somfy motors with the remote, including the used button combo's: http://www.automatedshadeinc.com/files/motors/all-somfy-rts%20motors-programming-quick-guide-02-09.pdf.

Thanks for sharing.

Reply

Samuele

— OCTOBER 22, 2014 AT 4:21 PM

How do you find the sequence of rolling codes? I need to press the button (ok, I can make a circuit to press the button :-)) on the remote 64k times?
Thanks.

Reply

pushstack

— OCTOBER 25, 2014 AT 8:44 AM

The sequence is very easy. Just +1 for every button press, eg. 1, 2, 3, 4, …..

Reply

nick

— NOVEMBER 21, 2014 AT 12:30 PM

Hi,

There’s a rather big community[1] around the CUL 433/868 devices[2]. They implemented a (meanwhile) read only version of the somfy protocol[3].

Your Blog is great, lead me into the basics of the protocol.

1. http://culfw.de/culfw.html#Features
2. http://busware.de/tiki-index.php?page=CUL
3. http://sourceforge.net/p/culfw/code/HEAD/tree/trunk/culfw/clib/somfy_rts.c

Reply

jondoe

— JANUARY 5, 2015 AT 10:20 AM

Hi there!

is the registration-process of a new remote device mandatory?

Yet I don’t see the point where it should be necessary, since we’re not sending a “sender-device-id”, correct?

@pushstack: thx for reverse engineering!

Reply

gerardwr

— JANUARY 5, 2015 AT 3:04 PM

No, the address of a new remote is sent when adding a new remote, have look at the “payload” : “Address: 24-bit identifier of sending device (little-endian)”.

When the new remote is added for every command the Address is sent too.

Reply

jondoe

— JANUARY 5, 2015 AT 4:21 PM

Oh sorry, I should learn how to read . Well I guess I have a hardware issue here because the payload i've sent should be correct. Iam using most of the code of the arduino forum thread mentoined above…

Reply

gerardwr

— JANUARY 5, 2015 AT 5:45 PM

Did you use the proper procedure?
1 – set the Somfy shade with an original remote in "learn" mode, the shade "jogs".
2 – have the Arduino send the datagram for "program a new remote", the shade "jogs"
3 – then the Arduin can send up/down datagrams, the shade (hopefully) moves up or down

Reply

jondoe

— JANUARY 5, 2015 AT 5:58 PM

No, thats why i asked if the registration is mandatory. I just tried to resend a frame of an existing device with a new rolling-code (which didn't work, and i'd like to figure out why).

I will try to do the registration process.

Reply

jondoe

— JANUARY 5, 2015 AT 10:01 PM

Mhpf, it didn't work either, 3rd Action doesn't happen. But I also noticed that while using my current device, the MSB of the "key"-Byte is rolling with the roll-code (as u mentioned). Maybe that must be sync with the "arbitrary"-value of the rolling_code while registering?

Reply

gerardwr

— JANUARY 5, 2015 AT 10:29 PM

No, as I understand it, this "extra" is ignored by the Somfy receiver.

Did the shade "jog" at step 1 AND step 2? When YES, then I assume the shade accepted your "new" remote.

Verify that:
– shade "jogs" at step 2
– increase the "rolling code" by 1, at every transmission
– (sometimes) the receiver only responds to the reception of multiple datagrams. Sometimes 1 datagram is OK, sometimes sending the same datagram 2,3,4 times seems to be required.

Good luck.

Reply

jondoe

— JANUARY 5, 2015 AT 11:02 PM

No, the shade “jogged” only during step 1.

I sent the frame in step 2 about a 100 times….

Reply

gerardwr

— JANUARY 5, 2015 AT 11:26 PM

My conclusion is that the frame is not recognized by the receiver.

If the frame is not correct sending it a million times will not help

Not sure how I can help you anymore.

Reply

jondoe

— JANUARY 5, 2015 AT 11:34 PM

Maybe you could take a look at my code/wiring diagram.

Reply

gerardwr

— JANUARY 5, 2015 AT 11:39 PM

Sure, tell me where I can see it.

jondoe

— JANUARY 6, 2015 AT 1:55 PM

Here is my wiring:
http://i57.tinypic.com/33maa1y.jpg
Here is the Code:
http://pastebin.com/sJgWRPkh

I provided a link to my used RF-modules in the code-header. Receiving works brilliant, but transmitting not.
I send via serial a cmd like "P100,0x123456#" which actually sends a Programm frame. But as I said, the shade doesn't "jog" when it is in programming mode.

Reply

Reich

— JANUARY 5, 2015 AT 5:20 PM

Hey,

was anyone able to transfer the described protocol to the Telis 4 remote? Due to a mass of measurements I'm still not able to find a solution. I found a fixed key including the count in front of the code of all channels and pairs of addresses for the controles. The described

Rolling Code and Address(A0|A1|A3) area is a mystery to me including the indication for the selected channel.

Has anyone a final code in C# .net or equal?

Reply

gerardwr

— JANUARY 5, 2015 AT 5:50 PM

Have a look here and catch up on your French

http://forum.arduino.cc/index.php?topic=208346.msg1840140#msg1840140

Reply

Reich

— JANUARY 5, 2015 AT 6:42 PM

thx a lot. Receiving works. Am I correct that this code id also able to transmit? This level of programming is far away from my skills.

Reply

gerardwr

— JANUARY 5, 2015 AT 9:19 PM

You're welcome!

I assume that you mean the code id that your Somfy remote transmits?

Using the same "address" as a Somfy remote will probably get you into trouble, because at each transfer the "rolling code" is increased and checked by the receiver. You will probably get into trouble, the receiver not recognizing your original remote anymore, and needs initialization.

Using a new code for transmission seems a better way.

Reply

Reich

— JANUARY 6, 2015 AT 10:26 AM

sry. The "d" should be a "s". I'm not sure how to use the code to transmit data. I don't know how to send commands.

To the other point you had. If I disconnected my blinds from power and use my remote, the "rolling code" increases. But after I connected the blinds again, everything works fine. Are you sure the revicer is smart enough handle the "rolling code"? I thought it is more or less a kind of filter to avoid doubble commands!

Reply

gerardwr

— JANUARY 6, 2015 AT 11:09 AM

Yes I'm sure that the receiver checks that the rolling code is increased by 1 (it believe to "tolerates" a slightly larger increase).

In my setup I had some HW troubles where the "rolling code" was increased, but the datagram was never sent. When the HW problem was corrected the receiver did not recognize my datagrams (wit a tool large rolling code increase) anymore, and I had to re-program my "remote" anew.

Reply

creamers

— JANUARY 6, 2015 AT 11:32 AM

The receiver accepts a window of codes. So using the remote when the shutter is powered down should not matter, unless you press is alot of times pushing it outside this 'window'
I have the same problem sending. Receiving works and analysing the raw codes gives me no other clues. It should work, but I use 433,9mhz with very new somfy shutters so probably it won't accept / receive it correctly. (somfy uses 433.4mhz)

Reply

gerardwr

— JANUARY 6, 2015 AT 12:12 PM

My Somfy shades are older, maybe 5 years, and I have had virtually few troubles sending working datagrams. Working now every day since a couple on months.

You're the "plight" guy, right? Tell me if I can help you.

Reply

creamers

— JANUARY 11, 2015 AT 12:35 PM

Hi Gerardwr,
Just a pilight user ,CurlyMo is 'the pilight guy' but very

busy with other things at the moment, but willing to test your setup with my somfy shutters if possible? Send me a private message through forum pilight org if possible…

Reply

dadaurka

— JANUARY 11, 2015 AT 12:23 PM

Hey gerardwr! My post somehow disappeared or wasn't approved by pushstack, maybe you could write me an email to dada@jondoe.de. Thanks in advance!

Reply

ruizprogelio

— JANUARY 23, 2015 AT 1:07 AM

Hi, i am very interested in getting this work, to this point I am able to receive and send successful datagrams with an arduino, I know this only because I have 2 arduinos (sender, receiver) and can decode what I am sending through the arduino sender with the arduino receiver, I can't make my motors recognize those commands though, my mail is rogelio@601.mx. I used very cheap 433.92 modules, hope anyone can help me.

Reply

pushstack

— JANUARY 24, 2015 AT 11:22 AM

I do think the 433.92 vs. 433.42 MHz mismatch does cause problems. Initially I also used a 433.92 MHz transmitter, but with this transmitter I had very limited range and some of the blinds I tested with didn't react at all, or only if I placed the antenna in a very specific sweet spot.

I can only suggest to use a transmitter that can send on 433.42 MHz. Like my Si4010 based solution or a CUL(see Nick's comment) or other CC11xx based solution. It might also be possible to modify your 433.92 MHz transmitter by replacing some capacitors/crystal. But I didn't had much success with that.

Reply

T

— FEBRUARY 17, 2015 AT 11:44 AM

Could you post your code if it's operational

Reply

pushstack

— FEBRUARY 18, 2015 AT 10:19 AM

What I currently use is a Si4010 with my own _ser4010_ firmware and the accompanying _ser4010-rts_ tool. This works very stable and has a good range. All required sources are available on GitHub: https://github.com/dimhoff/ser4010

Reply

Johannes

— FEBRUARY 28, 2015 AT 11:22 PM

Great job and excellent investigation! Finally I could decode my Somfy signals with the information found here!

One topic was unclear to me. When I look at patent US7860481 B2 it describes a format of 56bits in 71.68ms. With the fixed bit length for the 56bits, this would result in 1.28 ms per bit (or 1280us). For my tests taking this as bit length (and therefore 640us for a half bit) improves my reception substantially. Have you found another source for the 1208us in the doc here or have you found this by inspecting the signal?

Again, great stuff and thanks for providing all this information!

Reply

pushstack

— MARCH 1, 2015 AT 10:00 AM

The 1208 us come from the signal traces I made from "Somfy Smoove Origin RTS". I also noticed the difference with the patent and the surprising resemblance between the two numbers(eg. the 8 and 0 being swapped). I haven't redone my measurement, So it could be that I just fucked up somewhere.

What I did for my own decoding software is to logged the actual sample counts between edges and used these values. So I probably didn't really use the 1208us value there.

Thanks for you feedback, If I have some time i'll redo my calculations.

Reply

sok33t

— MAY 22, 2015 AT 5:19 PM

Hello Pushstack,

The control code 0x07have the following function:
0x07 -> My && Up && Down(pressed at the same time).

Reply

Sparx23

— SEPTEMBER 4, 2015 AT 8:28 AM

Just wanted to let you know that I got a RFM23BP working as a Somfy transmitter. I didn't even start to try the direct mode as this seems to be pretty complicated but used the normal OOK FIFO mode and set the bitrate so that one bit is (aproximately) 604us The longer periods are now just a multiple of that (E.g. HW Sync 4 x '1', 4 x '0', and so on). I basically adapted the code from: http://www.mike-stirling.com/2013/02/implementing-the-elv-fht-protocol-with-an-rfm23/
I only transmit signals, didn't try it as a receiver.

Reply

Leave a Reply

Enter your comment here...

Create a free website or blog at WordPress.com. | The Reddle Theme.