Low Power Ultra Long Range Transceiver

1 General Description

The K5553BB015 is a high-performance low power RF transceiver intended for RF wireless applications in the sub-1 GHz band.

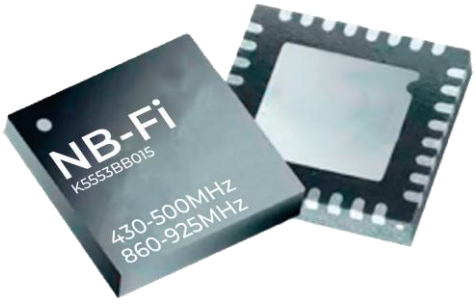
The K5553BB015 transceiver operates in the 430 - 500 MHz and 860 - 925 MHz frequency ranges.

The K5553BB015 transceiver supports high performance NB-Fi protocol with data rates of 50 - 25 600 bit/s with exceptional receiving sensitivity down to -148 dBm.

Operation for K5553BB015 transceiver is specified over the industrial temperature range, from -40°C to 85°C.

3 Applications

- Smart Metering
- Smart lighting systems
- Wireless alarm and security systems
- Industrial monitoring and control
- Wireless sensor networks
- Smart Agriculture
- IoT low data rate wireless applications



2 Key Product Features

High sensitivity/High selectivity Receiver:

- DBPSK modulation
- Built-in Forward Error Correction
- NB-Fi MAC Layer compatible
- Data rates: 50, 100, 400, 3200, 25600 bps
- Sensitivity:
  - -148 dBm @ 50 bps, 868 MHz
  - -145 dBm @ 100 bps, 868 MHz, Frequency Hopping mode
  - -139 dBm @ 400 bps, 868 MHz
  - -130 dBm @ 3200 bps, 868 MHz
  - -121 dBm @ 25600 bps, 868 MHz
- Adjacent channel suppression: 80 dB
- Immunity to reference oscillator error up to ±1 ppm for 50 bps receiving

Transmitter:

- DBPSK modulation with Frequency Hopping support
- Data rates:
  - Up to 100 kbps using digital pin-controlled phase modulator
  - Up to 25 600 bps for short DBPSK messages (288 & 320 bits) using built-in IQ modulator
- High Efficiency Power Amplifier
- Maximum Output Power: 15 dBm @ 3.3 V, 868 MHz
- Programmable Power Output Level with < 3 dB step
- Frequency bands:
  - 860 MHz - 925 MHz
  - 430 MHz - 500 MHz
- Programmable RF Carrier Frequency with < 10 Hz step
- Maximum bandwidth: 200 kHz
- 4-Wire SPI interface
- Supply voltage range: 1.8 V - 3.6 V
- Low power consumption:
  - Rx Mode, for 3.3 V supply:
    - 17 mA @ Low-current LNA mode
    - 23 mA @ Low-NF LNA mode
  - Tx Mode, for 3.3 V supply:
    - 20 mA @ 8 dBm
    - 50 mA @ 15 dBm
- Operating temperature range: from -40°C to +85°C
- QFN32 5x5 mm Package

Table of Contents

4 Block Diagram & Chip Architecture

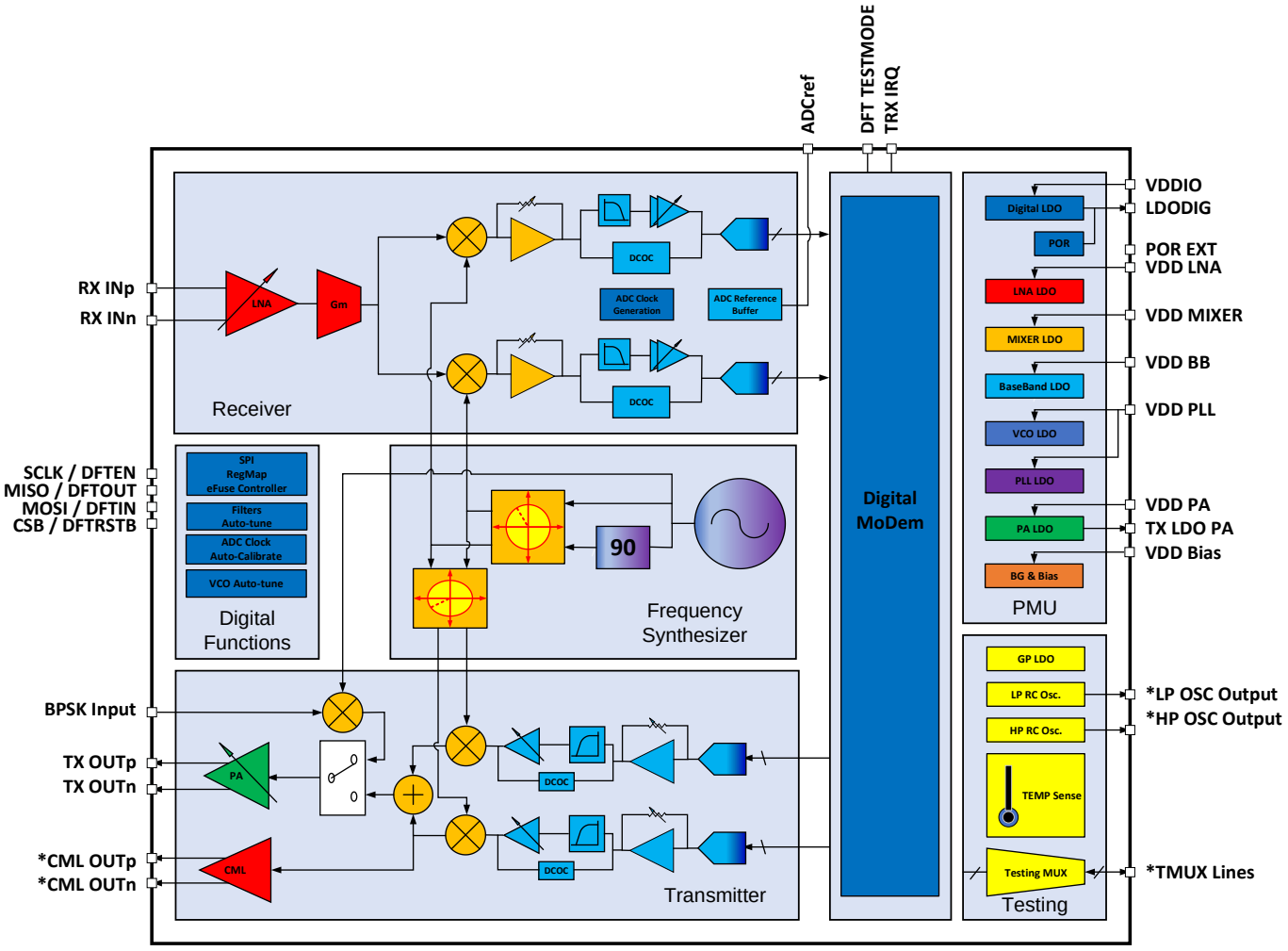


Figure 1. K5553BB015 Functional Block Diagram

The K5553BB015 receiver architecture is direct conversion where the received RF signal is converted directly to baseband. The received RF signal is amplified at the input by a differential low noise amplifier (LNA) followed by a trans-conductance (Gm) stage and quadrature down conversion mixer (I and Q) to baseband signal. The baseband signal is amplified and low pass filtered by 4th order active low pass filter (Rx-BB). The output signal from Rx-BB is digitized by analog to digital converter (ADC) and passed to the baseband modem for further processing.

The K5553BB015 transmitter utilizes direct conversion from baseband digital to analog converter (DAC), followed by an active low pass filter with programmable attenuation to RF frequency and finally output power is generated using an integrated class-E power amplifier (PA). Output power level can be configured from -8 dBm to +15 dBm at antenna with steps of 3 dB as maximum. Input of transmitter DAC is generated by the baseband modem.

The integrated PA is used for constant envelope modulation schemes as BPSK/DPSK. The integrated PA input has two modes:

1. RF input is generated from the integrated frequency synthesizer and its phase is modulated by a serial digital data stream (0° and 180°).
2. DAC baseband output is up converted directly to RF frequency in a quadrature architecture (I and Q), then summed up at the PA pre-driver input (Tx mixer output).

In case that a linear PA is required (due to a variable amplitude modulation scheme) or a higher output power level is required, there is a linear CML driver following the Tx mixer to drive an external PA while the integrated (internal) PA will be off in this mode.

The K5553BB015 has a single input clock for proper operation, TCXO at 26 MHz is expected.

The K5553BB015 can operate using a voltage supply ranging from 1.8 V to 3.6 V, based on integrated power management regulators.

A standard 4-pin SPI interface is used to communicate with the external MCU.

One-time programmable memory is integrated to store the ID and some required production trimming controls.

5 Pin Diagram

The following diagram shows the pin arrangement of the K5553BB015 QFN package, top view. A description of the pins is given in Table 1.

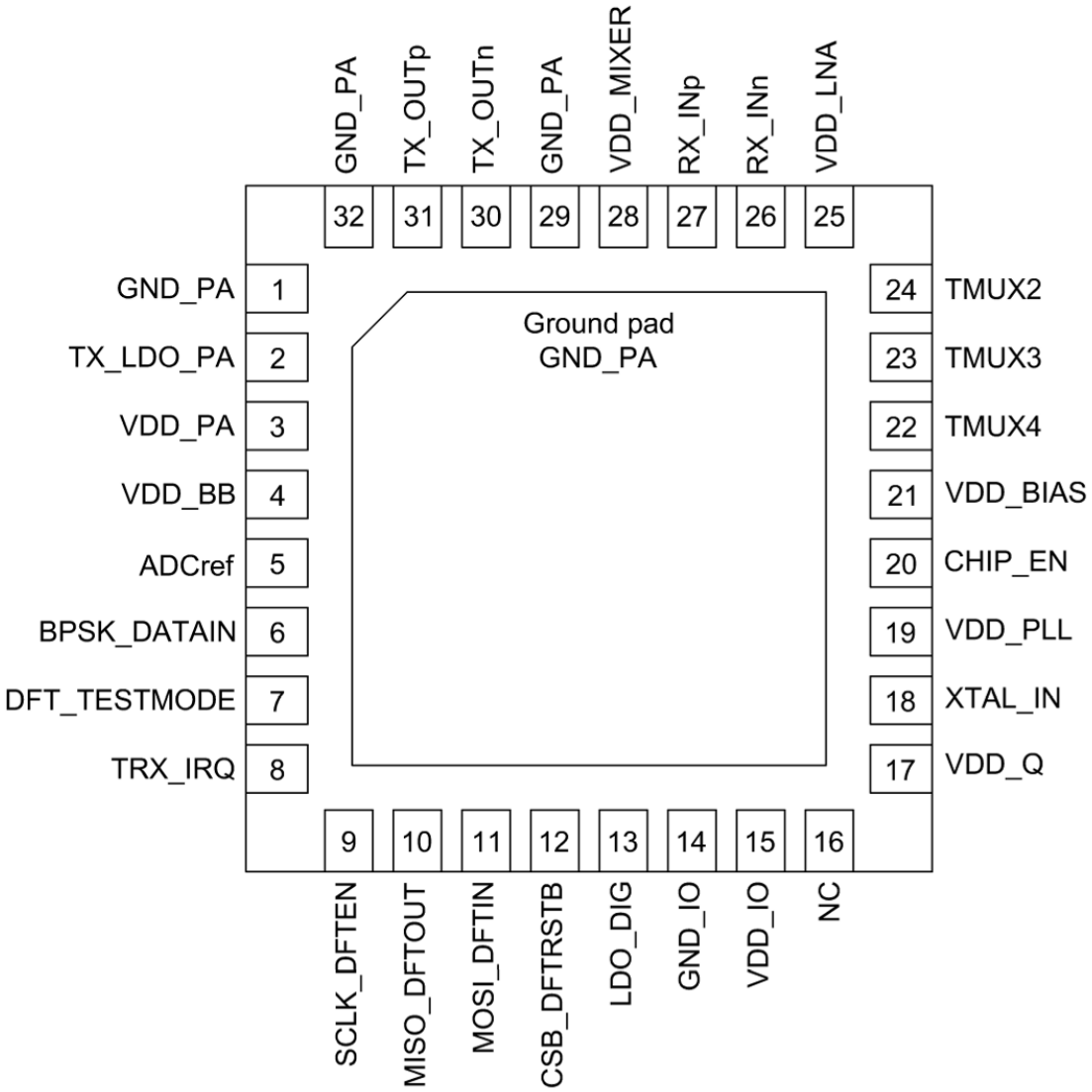


Figure 2. Pin Diagram

6 Pin List & Description

Table 1. Pin Description

Pin	Pin Name	Description
1	GND_PA	PA ESD Ground
2	TX_LDO_PA	PA LDO Output
3	VDD_PA	PA ESD Supply
4	VDD_BB	Base-Band ESD Supply
5	ADCref	SAR ADC Reference Voltage
6	BPSK_DATAIN	BPSK Input Data
7	DFT_TESTMODE	DFT Test Mode Enable
8	TRX_IRQ	Interrupt Signal
9	SCLK_DFTEN	SPI Clock
10	MISO_DFTOUT	SPI Data Out
11	MOSI_DFTIN	SPI Data In
12	CSB_DFTRSTB	Slave Select
13	LDO_DIG	Digital LDO (1.1 V) Output
14	GND_IO	I/O ESD Ground
15	VDD_IO	I/O ESD Supply
16	NC	Not Connected
17	VDD_Q	eFuse Supply
18	XTAL_IN	Crystal Oscillator Input
19	VDD_PLL	PLL ESD Supply
20	CHIP_EN	Chip Enable
21	VDD_BIAS	Bias ESD Supply
22	TMUX4	TMUX line #4
23	TMUX3	TMUX line #3
24	TMUX2	TMUX line #2
25	VDD_LNA	LNA & Mixers ESD Supply
26	RX_INn	LNA negative Input
27	RX_INp	LNA positive Input
28	VDD_MIXER	Mixers Supply
29	GND_PA	PA ESD Ground
30	TX_OUTn	PA negative Output
31	TX_OUTp	PA positive Output
32	GND_PA	PA ESD Ground
Ground pad	GND_PA	PA ESD Ground

7 Application Information

7.1 Typical Application Diagram

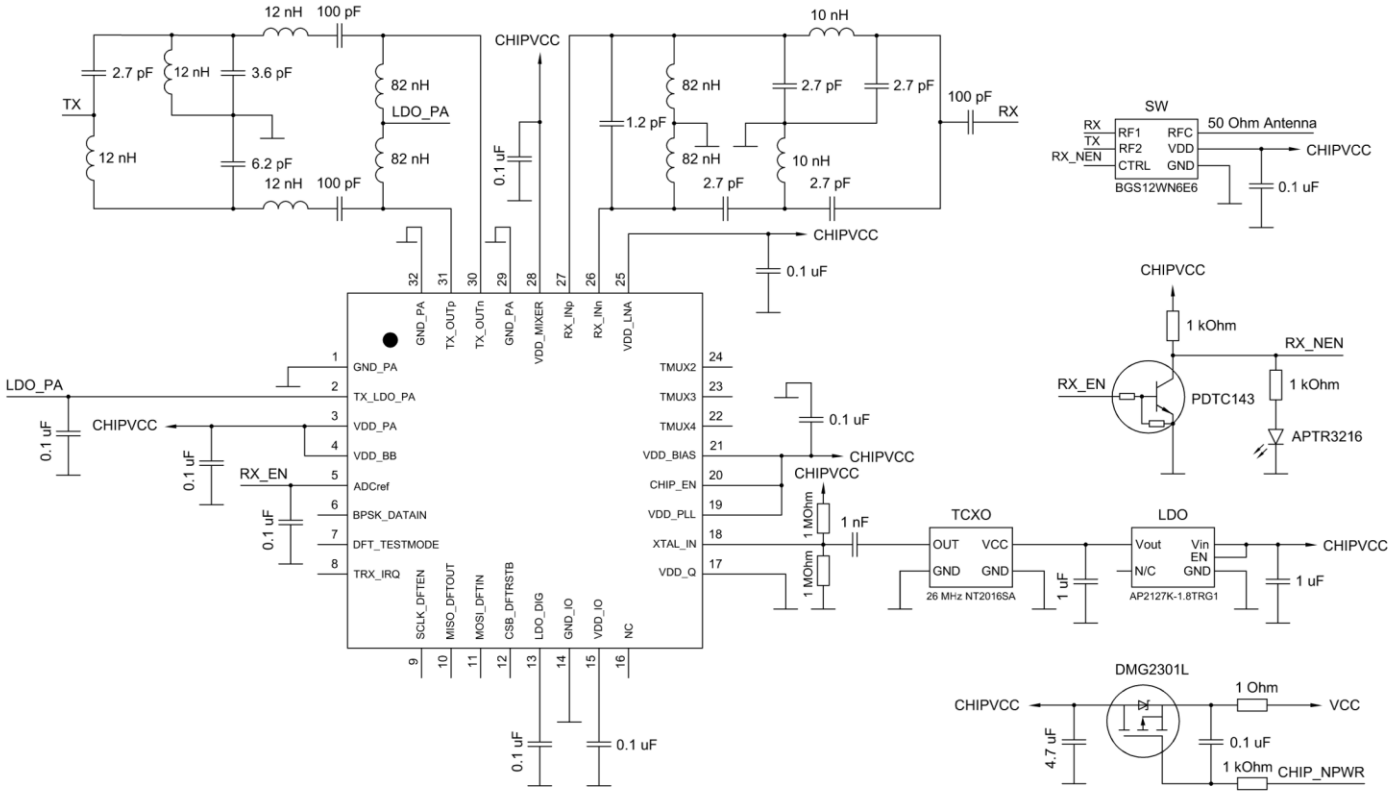


Figure 3. Typical Application Diagram

For detailed application configuration and BOM see the Data Store section at the www.nb-fi.org website.

7.2 Recommended Land Pattern

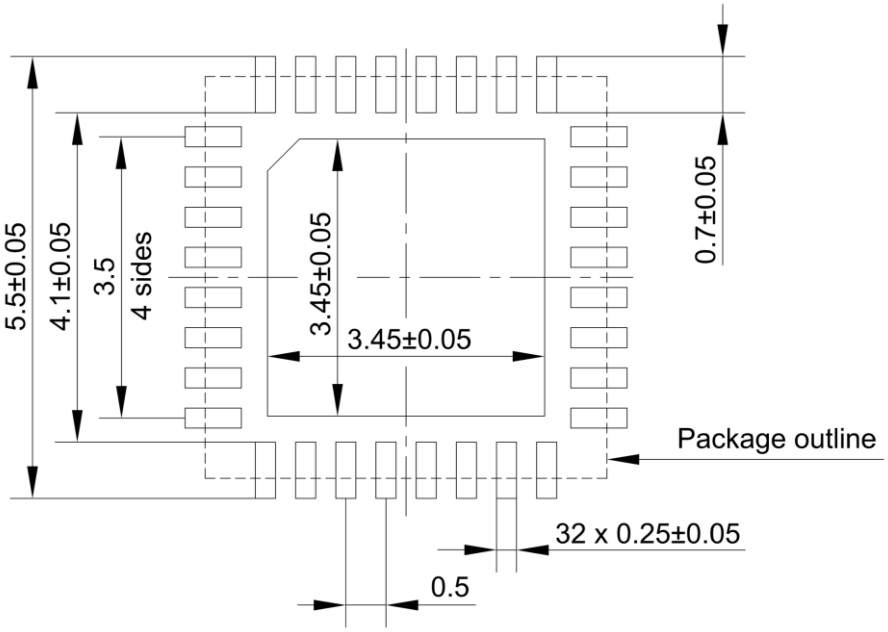


Figure 4. Recommended Land Pattern

8 Chip Modes

The K5553BB015 transceiver has different modes of operation with different expected transition times and different blocks ON/OFF as indicated in Table 2.

Table 2. Chip Operation Modes and Expected Transition Times

Initial Mode	Initial Mode Conditions	Idle	Normal Rx	Normal Tx	Sleep	Deep Sleep
Idle	All bias circuits are ON All Rx Blocks are OFF All Tx Blocks are OFF LO is ON	N/A	20 µs	20 µs	20 µs	1 ms
Normal Rx	All bias circuits are ON All Rx Blocks are ON All Tx Blocks are OFF LO is ON	10 µs	N/A	10 µs	20 µs	1 ms
Normal Tx	All bias circuits are ON All Rx Blocks are OFF All Tx Blocks are ON LO is ON	10 µs	10 µs	N/A	20 µs	1 ms
Sleep	All bias circuits are ON All Rx Blocks are OFF All Tx Blocks are OFF LO is OFF	3 ms	3 ms	3 ms	N/A	1 ms
Deep Sleep	Core bias circuits[1] are OFF All Rx Blocks are OFF All Tx Blocks are OFF LO is OFF	3 ms	3 ms	3 ms	3 ms	N/A
Shutdown	All circuits are OFF (Analog & Digital)	Transition from Shutdown to any other mode is expected in 3ms				

The Shutdown mode is not controlled by SPI, it is controlled by a Chip Enable pin.

All other modes (other than Shutdown) are controlled by SPI commands where all transitions are allowed as indicated in Table 3.

Table 3. Mode controls in different modes of operation

Mode / Control	TxRx_poff REG 0x402F[7]	TxRx_sleep REG 0x402F[6]	TxRx_idle REG 0x402F[5]	TxON_RxOFF REG 0x402F[4]
Deep Sleep	1	X	X	X
Sleep	0	1	X	X
Idle	0	0	1	X
Normal Tx	0	0	0	1
Normal Rx	0	0	0	0

[1] Core bias circuits include reference voltage (bandgap), LDOs (Other than digital LDO) and bias voltage & current for any block other than Rx, Tx & Synthesizer (LO) blocks.

9 Specifications

All typical values are at ambient room temperature (25°C) and at supply of 3.3 V.

9.1 ESD Ratings

Table 4. K5553BB015 ESD Ratings

Parameter	Min.	Typ.	Max.	Unit
VESD – HBM			±2000	V
VESD – CDM			±500	V

9.2 Operating Range

Operating range defines the supply voltage and temperature ranges where this device is functional.

Table 5. K5553BB015 Operating Range

Parameter	Min.	Typ.	Max.	Unit
Operating supply voltage	1.8	3.3	3.6	V
Operating ambient temperature range	-40	25	+85	°C

9.3 Performance Specifications

Table 6. K5553BB015 Performance Specifications

Specification	Min	Typ	Max	Units	Comments/Test Condition
Band of Operation					
Lower Band	430		500	MHz	
Upper Band	860		925	MHz	
Current Consumption					
Shutdown Current			5	µA	Disconnecting TCXO and forcing Enable low – All Chip blocks are OFF
Deep Sleep Current		2.5	4.5	mA	Digital Backend only is ON and the rest of Chip is OFF
Sleep Current		3.2	5.0	mA	Digital Backend and Core Bias cells are ON – rest is OFF
Idle Current		9	13	mA	Same as Sleep mode + Frequency Synthesizer is ON
Receiver Current Upper Band		17.5	25.0	mA	Complete Rx chain is ON
Tx BPSK Upper Band		50	60	mA	Complete TX Chain is ON @Max Output Power
Receiver Specifications					
S11		-11		dB	
Gain	44		92	dB	
Gain Step		3.0		dB	
NF		3.6		dB	
Input P1dB		-40		dBm	@ Rx Gain = 44 dB
IIP3		-50		dBm	@ Rx Gain = 44 dB
DC Offset (VGA Output)	±3		±20	mV	
Transmitter (BPSK mode) Specifications					
Max output Power (@ PA Gain setting = 21)	12	15	17	dBm	
Min Output Power (@ PA Gain Setting = 0)	-25	-10	-8	dBm	
Output Power Step	1.2	3.0		dB	

Note – Maximum and Minimum are across temperature.

10 Description of Messages that can be Received and Transmitted

10.1 Rx Messages

The total length of a message that can be received by the K5553BB015 transceiver (hereinafter – Rx Packet) is 36 bytes. An example code for generating an Rx Packet in the C language is given in section 10.4.

The format of Rx Packets is the same for various data rates. The bit ordering in the bytes of an Rx Packet is from the higher one to the lower one.

The structure of an Rx Packet is given in Table 7. This Rx message structure is compatible to the DOWNLINK packets specified by the NB-Fi protocol.

Table 7. Structure of an Rx Packet

Preamble	Error Correction Code Data (Input data for the Forward error correction code)		Error Correction Code
	Payload	Packet CRC	
4 bytes	13 bytes	3 bytes	16 bytes

10.1.1 Preamble Field

The preamble is required to detect a packet in the radio spectrum. The size of the Preamble field is 4 bytes.

Typically, the algorithm of generating the preamble for an Rx Packet is based on iterative generation of pseudorandom numbers with a 4-bytes seed (usually endpoint ID) as a starting position of the generator. The correlation factor of the obtained preamble should be checked in the process of each iteration. If the selected correlation factor criterion is satisfied, the preamble generation is deemed completed. An example code for generating the preamble of an Rx Packet in the C language is given in section 10.5.

10.1.2 Payload Field

The Payload field is used to transfer a user-defined data, that may include encrypted data, message integrity code, other message overheads if required. The size of the Payload field is 13 bytes.

A detailed structure of the Payload field for NB-Fi messages is provided in the NB-Fi specification that is out of scope of this datasheet.

10.1.3 Packet CRC Field

The Packet CRC field contains a CRC32 checksum of the Payload field (from the 5th to the 17th byte of the Rx Packet). The three lower bytes of this value are used. The byte ordering is from the higher one to the lower one. The code for CRC32 checksum calculation in the C language is given in section 10.6.

10.1.4 Error Correction Code Field

The Error Correction Code field is a code word of the forward error correction code and should be calculated from the data of the Error Correction Code Data field. The Error Correction Code Data field is in total composed of the Payload and Packet CRC fields. The size of the Error Correction Code field is 16 bytes.

The Zigzag code with a code rate of ½ is used for forward error correction[2]. The interleaver for ZIGZAG Data Encoding as well as the code for the ZIGZAG forward error correction in the C language are given in section 10.7.

[2] L. Ping, Xiaoling Huang, N. Phamdo (1999). Zigzag Codes and Concatenated Zigzag Codes. URL: https://pdfs.semanticscholar.org/3091/6eb78443174658fbf8d4464a5bf966846399.pdf.

10.2 Tx Messages

The K5553BB015 transceiver supports transmitting of the following messages (hereinafter – Tx Packets):

1. Any DBPSK/BPSK modulated messages at data rates of up to 100 kbps. RF signal is generated from the integrated frequency synthesizer and its phase is modulated by a serial digital data stream (0° and 180°) from BPSK_DATAIN pin.
2. DBPSK messages with size of 288 or 320 bits at data rates of up to 25 600 bps using built-in IQ modulator. Thus, the K5553BB015 transceiver allowing to transmit the NB-Fi messages, specified by the NB-Fi protocol, with enabled built-in IQ modulator. Using of the built-in IQ modulator allows to reduce MCU power consumption by transitioning it to the inactive state right after pushing the message to the transceiver's RAM input stack.

Use the 0x4006[0] register (see 12.1) to enable the Tx BPSK or Tx IQ mode.

10.3 NB-Fi Specification Overview

This NB-Fi specification imposes requirements on the wireless NB-Fi exchange protocol for the Internet of Things in the narrow-band spectrum, including requirements for the Physical Layer, MAC Layer and Transport Layer of the Open System Interconnection Model (OSI Model) of the NB-Fi protocol.

The MAC Layer of the of the NB-Fi protocol provides for transmission of a datagram (an information packet) at the Physical Layer and describes the format of UPLINK and DOWNLINK packet fields, addressing methods, data protection methods, data integrity control methods, and error recovery methods.

A description of the Physical Layer of NB-Fi protocol is given in Table 8. A description of the MAC and Transport Layers of NB-Fi protocol is given in Table 9.

As it follows from Tables 8 - 9, the K5553BB015 transceiver is compatible with the NB-Fi protocol at Physical & MAC Layers.

Table 8. NB-Fi Physical Layer overview

Parameter name	Parameter value
Minimum reception bandwidth of the NB-Fi base station	51.2 KHz
Minimum transmission bandwidth of the NB-Fi base station	102.4 KHz
Modulation	DBPSK or BPSK
Data rates	50, 400, 3200, 25600 bps
NB-Fi message length	288 bits

Table 9. NB-Fi MAC and Transport Layers overview

Parameter name	Parameter value
Network numbering capacity	4294967296 (2^{32}) NB-Fi endpoints
Effective data rates	12.5, 100, 800, 6400 bps
Forward error correction code type (for NB-Fi DOWNLINK packets)	ZIGZAG code[3]
Forward error correction code rate (for NB-Fi DOWNLINK packets)	½
Payload encryption	A symmetric block encryption algorithm (AES or other)
Encryption key size	256 bits
Length of the payload of one NB-Fi packet at Transport Layer	8 bytes
Maximum payload length for a group packet at Transport Layer	240 bytes

An NB-Fi specification is available in the Data Store section at the www.nb-fi.org website.

[3] With the same Interleaver, as given in section 10.7.

10.4 Example Code for Generating and Sending an Rx Packet

```
nbfi_status_t NBFi_MAC_TX_ProtocolD(nbfi_transport_packet_t* pkt)
{
 uint8_t ul_buf[64];
 uint8_t len = 0;
 static _Bool parity = 0;
 uint8_t lastcrc8;
 uint32_t mic_or_crc32;

 nbfi_phy_channel_t phy;

 uint32_t tx_freq;

 static uint32_t preamble;

 memset(ul_buf,0,sizeof(ul_buf));

 preamble = get_preamble(Modem_ID, (uint32_t *)0, (uint32_t *)0);
 for(int i=0; i<sizeof(preamble); i++)
       ul_buf[len++] = ((uint8_t *)&preamble)[sizeof(preamble) - 1 - i];

 ul_buf[len++] = nbfi_iter.ul;
 ul_buf[len++] = pkt->phy_data.header;
 memcpy(&ul_buf[len], pkt->phy_data.payload, pkt->phy_data_length);

 NBFi_Crypto_Encode(&ul_buf[len - 1], Modem_ID, nbfi_iter.ul, 9);
 len += 8;
 mic_or_crc32 = NBFi_Crypto_UL_MIC(&ul_buf[len - 9], 9);
 nbfi_iter.ul = NBFI_Crypto_inc_iter(nbfi_iter.ul);
 ul_buf[len++] = (uint8_t)(mic_or_crc32 >> 16);
 ul_buf[len++] = (uint8_t)(mic_or_crc32 >> 8);
 ul_buf[len++] = (uint8_t)(mic_or_crc32);

 mic_or_crc32 = CRC32(ul_buf + 4, 13);

 ul_buf[len++] = (uint8_t)(mic_or_crc32 >> 16);
 ul_buf[len++] = (uint8_t)(mic_or_crc32 >> 8);
 ul_buf[len++] = (uint8_t)(mic_or_crc32);

 if(nbfi.tx_freq)
 {
  tx_freq = nbfi.tx_freq ;
 }
 else
 {
  tx_freq = NBFi_MAC_get_DL_freq();
 }

 ZCODE_Append(&ul_buf[4], &ul_buf[len], 1);

 NBFi_RF_Init(nbfi.tx_phy_channel, (nbfi_rf_antenna_t)nbfi.tx_antenna,
 nbfi.tx_pwr, tx_freq);

 NBFi_RF_Transmit(ul_buf, len + ZCODE_LEN, phy, NONBLOCKING);
 return OK;
}
```

10.5 Example Code for Generating the Preamble of an Rx Packet

```
//preamble.h

#ifndef _PREAMBLE_H_
#define _PREAMBLE_H_

#ifdef __cplusplus
extern "C"
{
#endif

#include <stdint.h>

#define RAND_MULL    0x1234
#define RAND_ADD     0x10

#define MAX_ITER     100
#define MAX_COEFF    6

uint32_t get_preamble(uint32_t seed, uint32_t *iter, uint32_t *coeff);

#ifdef __cplusplus
}
#endif
#endif
//preamble.c

#include "preamble.h"
#include <stdint.h>
#include <stdlib.h>

static uint32_t _alu(uint32_t data, uint32_t tmp)
{
        uint32_t test = data ^ tmp;
        int32_t count = 0;
        while (test)
                count += test & 0x01, test >>= 1;
        count = abs(count - (32 - count)) >> 1;

        return count;
}

static uint32_t _factor(uint32_t data)
{
        uint32_t i, count, tmp, max = 0;
        for (i = 0, tmp = data; i < 32; i++)
        {
                count = _alu(data, tmp);
                tmp <<= 1;
                if (count > max && i)
                        max = count;
        }
        for (i = 0, tmp = data; i < 32; i++)
        {
                count = _alu(data, tmp);
                tmp >>= 1;
```

```
                if (count > max && i)
                        max = count;
        }
        return max;
}

static uint32_t _random(uint32_t seed)
{
        static uint32_t _seed;
        if (!seed)
        {
                _seed = _seed * RAND_MULL + RAND_ADD;
                _seed = _seed << 7 | _seed >> 23;
        }
        else
                _seed = seed;

        return _seed;
}

uint32_t get_preamble(uint32_t seed, uint32_t *iter, uint32_t *coeff)
{
        uint32_t _rand, _coeff, _iter = 0;
        _random(seed);
        while (_iter < MAX_ITER)
        {
                _rand = _random(0);
                _coeff = _factor(_rand);
                _iter++;
                if (_coeff < MAX_COEFF)
                        break;
        }
        if (iter)
                *iter = _iter;
        if (coeff)
                *coeff = _coeff;

        return _rand;
}
```

10.6 Code for CRC32 Checksum Calculation

```
uint32_t CRC32(const uint8_t *buf, uint8_t len)
{
    return digital_update_crc32(0xffffffff, buf, len) ^ 0xffffffff;
}

#define WIDTH (8*4)
#define TOPBIT (1 << (WIDTH-1))
#define POLYNOMIAL (0x104C11DB7)

uint32_t crc_table(uint8_t n)
{
    uint32_t c;
    int k;
    c=((uint32_t)n) << (WIDTH - 8);
    for(k=8;k>0;k--)
    {
             if(c & (uint32_t)TOPBIT)
        {
            c = (c<<1) ^ POLYNOMIAL;
        }
        else
        {
            c=c<<1;
        }
    }
    return c;
}

uint32_t digital_update_crc32( uint32_t crc, const uint8_t *data, uint8_t len )
{
    while (len > 0)
    {
     crc = crc_table(*data ^ ((crc >> 24) & 0xff)) ^ (crc << 8);
     data++;
     len--;
    }
    return crc;
}
```

10.7 Interleaver and Code for ZIGZAG Data Encoding

```
const uint8_t n[4][128] =
{
{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,2
9,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55
,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,
82,83,84,85,86,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,102,103,104,105,1
06,107,108,109,110,111,112,113,114,115,116,117,118,119,120,121,122,123,124,125,
126,127},

{104,52,43,96,31,7,71,78,58,37,93,25,125,85,42,111,6,95,72,117,27,51,63,84,91,3
5,120,26,97,45,110,70,1,28,86,114,53,67,12,127,40,101,73,94,115,61,20,126,3,46,
92,116,9,56,87,77,109,44,65,54,100,118,2,34,21,41,76,14,69,124,90,18,103,48,113
,36,0,81,13,62,24,38,105,68,15,75,88,50,122,29,83,102,8,16,108,23,32,49,99,112,
19,55,89,11,107,82,47,98,22,30,60,80,66,121,10,57,17,39,79,4,64,123,33,59,106,7
4,5,119},

{26,10,105,48,38,84,76,57,23,125,115,3,106,33,77,99,71,113,22,1,44,87,8,31,111,
96,2,42,70,81,13,93,122,37,114,88,63,107,50,40,82,116,68,6,127,16,51,73,61,83,4
6,0,126,104,78,67,41,119,28,11,56,47,4,21,52,66,15,98,24,7,30,91,112,35,55,124,
64,5,95,32,49,9,85,65,43,18,92,36,12,86,118,60,25,72,53,80,123,45,58,102,110,12
0,89,34,17,75,94,27,100,62,20,39,108,90,69,117,97,59,79,109,101,19,121,54,29,14
,74,103},

{0,93,104,36,87,125,23,97,44,107,11,3,70,35,60,77,29,84,6,91,126,15,76,56,4,89,
115,99,43,22,122,16,105,55,2,113,78,51,63,14,120,102,8,19,68,111,86,47,64,32,12
1,72,59,108,96,80,25,67,118,12,58,127,20,90,9,37,103,53,62,69,85,10,110,34,100,
119,39,73,1,83,48,112,30,54,65,45,5,123,101,26,88,18,46,95,40,109,7,27,57,66,11
6,38,75,92,21,52,61,28,106,114,94,33,17,79,42,71,124,50,82,13,31,41,117,74,98,8
1,24,49}
};

#define ZCODE_LEN 16

void ZCODE_Append(uint8_t * src_buf, uint8_t * dst_buf, _Bool parity)
{
 uint8_t b0;
 uint8_t b1;
 uint8_t bprev;
 uint8_t res;
 uint8_t code[4][8]= {
                     {0,0,0,0,0,0,0,0},
                     {0,0,0,0,0,0,0,0},
                     {0,0,0,0,0,0,0,0},
                     {0,0,0,0,0,0,0,0}
                     };
for(int j=0; j<4; j++)
 {
  for(uint8_t i=0; i<64; i++)
  {
   b0 = (src_buf[ n[j][ i ]  /8] << ( n[j][ i ]  %8)) & 0x80;
   b1 = (src_buf[ n[j][64+i] /8] << ( n[j][64+i] %8)) & 0x80;
   bprev   = (code[j][   (i-1)    /8] << (   (i-1)    %8)) & 0x80;
   res     =  b0 ^ b1 ^ bprev;
   code[j][i/8] |= res >> (i%8);
  }
 }
```

```
 for(int i=0; i<8;i++)
 {
  dst_buf[i] = (code[0][i] & (parity ? 0xAA : 0x55)) | (code[1][i] & (parity ?
 0x55 : 0xAA));
  dst_buf[i+8]    = (code[2][i] & (parity ? 0xAA : 0x55)) | (code[3][i] & (parity
 ? 0x55 : 0xAA));
 }
}
```

11 Main Chip Settings

11.1 Initial settings

Some register values should be written by SPI commands as stated in Table 10.

Table 10. Example of Initial K5553BB015 Settings

Register Address [Hex]	Value [Binary]	Value [Decimal]	Notes
4000	10101110	174	Fractional mode PLL
	10101010	170	Integer mode PLL
4002	10101110	174	for 860 MHz - 925 MHz band
	00101110	46	for 430 MHz - 500 MHz band
4003	01010111	87	for 860 MHz - 925 MHz band
	01011111	95	for 430 MHz - 500 MHz band
4004	11011110	222	ADC ref & I-ADC delay unit
4005	11011010	218	Bypass LPF tuner & VCO cap tune
4006	01111010	122	I/Q Tx mode
	01111011	123	BPSK Tx mode
4008	00010111	23	ADC Compvalid & Mixer CM
400A	01110011	115	GP ADC setting
400C	01111000	120	GP ADC delay unit
400D	10001000	136	LPF tune (Tx & Rx)
400E	01000110	70	Clock Generator setting
400F	01100110	102	LNA tune
4010	00011000	24	Clock Generator setting
4011	01100100	100	Clock Generator setting
4013	00000001	1	ADC-Q setting
4016	01101011	107	Tx DAC clock
4019	00010000	16	Low Power ref trim
401F	11111011	251	PLL lock detector mode (narrow mode)
4023[7:2]	100001	33	N_{int} (Integer value of the PLL divider) for the 868800000 Hz frequency (see 11.3)
4023[1:0]	10	2	Fractional mode PLL for the 868800000 Hz frequency (see 11.3)
4026[7:0]	10101001	169	N_{frac} (Fractional value of the PLL divider) for the 868800000 Hz frequency (see 11.3)
4027[7:0]	10010101	149	
4028[6:1]	00011010	26	
4028[0]	1	1	
402A	01111111	127	BB Tuner enable & Tx VGA current
4030[5]	1	1	Rising edge trigger of PLL auto tuning (VCO tuner)

Note – It is always mandatory to check the 0th bit of address 0x4039 "regfile_init_done" for the SPI master after Chip power up and before any SPI communication. SPI communication is only allowed at startup if this flag is HIGH.

11.2 Power Mode Settings

The Chip normally starts in deep sleep mode where Tx, Rx, PLL and all bias & LDOs are OFF except for the digital LDO to enable SPI communication. Then to switch to any other mode, it is required to set register 0x402F according to Table 11.

Table 11. K5553BB015 Power Modes Control

Power Mode	REG 0x402F Value [Bin]	REG 0x402F Value [Dec]	Notes
Deep Sleep	10000000	128	Only SPI communication is allowed
Sleep	01000000	64	All bias & LDOs are ON
Idle	00100000	32	All bias, LDOs & PLL are ON
Tx	00010000	16	Tx mode is enabled
Rx	00000000	0	Rx mode is enabled

Note – It is always mandatory to check the value of 0th bit in register 0x403B "pmode_switch_busy" for the SPI master after writing the power mode control register (0x402F) before any further communication to make sure the mode switch is complete. This bit acts as busy flag, as long as this bit is HIGH, then mode switching is still in progress & no other SPI write command is allowed. Once the mode switching is complete, the busy flag reset to LOW automatically & any SPI write command is allowed.

11.3 PLL Locking & Frequency Control

To change the carrier frequency from the PLL, it is required to change the divider value & expect the output frequency ($\boldsymbol{F_{out}}$) as in equation below:

$$\boldsymbol{F_{out}} = \left(\left(\boldsymbol{N_{int}} + \boldsymbol{N_{frac}}\right) * \boldsymbol{F_{ref}}\right) / (\mathbf{2} - \boldsymbol{txrx\_bandsel})$$

where:

- $\boldsymbol{N_{int}}$ is the integer value of the PLL divider which is the decimal equivalent of register 0x4023[7:2] which can be changed from 0 to 63.
- $\boldsymbol{N_{frac}}$ is the fractional value of the PLL divider which is the decimal equivalent of 22 bits divided by 2^{22}. The 22 bits are consuming the registers 0x4026[7:0], 0x4027[7:0] & 0x4028[6:1] while 0x4026 is LSB & 0x4028 is the MSB.
- $\boldsymbol{F_{ref}}$ is the reference frequency from the TCXO (26 MHz).
- $\boldsymbol{txrx\_bandsel}$ is the value of 7th bit in address 0x4002, which is "0" for Lower Band and "1" for Upper Band.

To close the PLL loop & make sure the VCO is tuned at the right frequency, it is required to follow the below steps:

1. Make sure the Chip is in a mode where PLL is enabled (Idle, Tx or Rx).
2. Write the $\boldsymbol{N_{int}}$ & $\boldsymbol{N_{frac}}$ values in registers 0x4023[7:2], 0x4026, 0x4027 & 0x4028[6:1] to get the required $\boldsymbol{F_{out}}$ based on the equation above.
3. Tune the VCO by changing 5th bit of register 0x4030 from 0 to 1 as the VCO tuning is +ve edge triggered. Check that tuning is complete by checking 2nd bit in register 0x4034, if it is set to HIGH, then tuning is complete. Also check tuning output value from register 0x4034[7:3] which represents the VCO frequency curve selected by the auto tuner.

4. Every time the N_{int} is changed, step 3 is recommended to be repeated for the best performance of the PLL.

Check the output frequency of the PLL by checking transmitter PA output in BPSK mode while BPSK data can be static at HIGH or LOW.

Also check locking of PLL by checking 1st bit of address 0x403B "adc_clk_ready". If PLL is locked and control voltage is within the expected range, then this bit will be HIGH.

Note – Every time the PLL is turned OFF, it is required to repeat the tuning steps again.

11.4 Transmitter Output Power Control

The transmitted output power can be controlled by PA gain control, this control is 5 bits & located in register 0x402D[4:0] "txpa_gctrl". It is expected to control PA output power by sweeping the decimal equivalent value this control from 0 (lowest output power < -13 dBm) to 21 (Highest output power >15 dBm) with a step not larger than 3 dB.

11.5 Transmitter DAC output amplitude

In Tx-IQ mode, DAC output amplitude can be controlled by 0x402D[7:5] "txdac_gctrl". The maximum amplitude is at zero & minimum amplitude is at 7.

There is a trade-off between linearity & output power which can be optimized by txdac_gctrl.

11.6 Receiver Gain Control

There are four controls over the receiver gain distributed along the chain. These controls are:

- LNA gain control: it is recommended to have it fixed at the highest gain for the best NF.
- Rx Mixer gain control: it is recommended to have this at 5 in Upper Band and 7 in Lower Band.
- Rx VGA gain control: This is the main gain control in the receiver. There are two cascaded stages of VGA with control of gain from 0 dB to 24 dB (step ~ 3 dB) in each stage. It is recommended to increase gain in 2nd VGA (VGA2) before increasing of 1st VGA (VGA1) for better linearity, the reverse for better NF & distribute the gain between VGA1 & VGA2 for a better compromise between NF & Linearity.

The registers of the different Rx gain controls are shown in Table 12.

Table 12. Receiver gain controls

Gain Control	Register address [Hex] & control bits	Control Value [Dec]	Notes
LNA	0x4002[2:1]	3	Set at maximum
Rx Mixer	0x4003[5:2]	5	for 860 MHz - 925 MHz band
		7	for 430 MHz - 500 MHz band
VGA1	0x4029[7:4]	0	Maximum Gain (24 dB)
		8	Minimum Gain (0 dB)
VGA2	0x4029[3:0]	0	Maximum Gain (24 dB)
		8	Minimum Gain (0 dB)

11.7 eFuse Details

11.7.1 eFuse Registers

The internal state machine makes use of some SPI registers to enable the different eFuse operations. The list of these registers are:

1. Address register (eFuse_addr at address 0x4020): used for the eFuse bit/byte address in different eFuse operations (if required);
2. Control register (eFuse_ctrl at address 0x4021): used to control the state/operation of eFuse state machine;
3. Read register (eFuse_Read_Data at address 0x403C): used to keep the read data from certain eFuse address in eFuse read operation.

There are also two flags to be checked after eFuse operations as below:

1. "eFuse_st_cmdDone": in 1st bit of address 0x4039 which indicates that required eFuse operation is complete and eFuse state machine is ready for a new operation;
2. "regfile_init_done": in 0th bit of address 0x4039 which indicates that Chip initialization after power up is complete including automatic reload of all eFuse registers if "eFuse_fully_burned" is programmed to "1" (3rd bit of address 0x4005).

11.7.2 eFuse Operations

eFuse controller supports six different operations as below:

1. Read: copy the eFuse data in "eFuse_addr" into "eFuse_Read_Data";
2. Reload Byte: copy eFuse data in "eFuse_addr" into the equivalent mirror register;
3. Reload All: copy all eFuse data into all mirror registers;
4. Program Bit: program eFuse bit defined by "eFuse_addr" by "1";
5. Program Byte: program "eFuse_addr" byte by the data in the mirror register of this address;
6. Program All: program all eFuse addresses by data in the mirror registers.

To start any eFuse operation, follow the below steps for each operation.

11.7.3 Read

The procedure of a Read operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
2. Make sure that VDDQ pin is tied to ground;
3. Use SPI write to set the "eFuse_addr" register value to the required address to be read;
4. Use SPI write to set the "eFuse_ctrl" register value to 0x06;
5. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
6. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
7. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

11.7.4 Reload Byte

The procedure of a Reload Byte operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
2. Make sure that VDDQ pin is tied to ground;
3. Use SPI write to set the "eFuse_addr" register value to the required address to be reloaded;

4. Use SPI write to set the "eFuse_ctrl" register value to 0x42;
5. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
6. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
7. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

11.7.5 Reload All

The procedure of a Reload All operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
2. Make sure that VDDQ pin is tied to ground;
3. Use SPI write to set the "eFuse_ctrl" register value to 0x0A;
4. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
5. Use SPI write to set the "eFuse_ctrl" register value to 0x02;
6. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

11.7.6 Program Bit

The procedure of a Program Bit operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
2. Use SPI write to set the "eFuse_addr" register value to the required bit address to be programmed;
3. Connect VDDQ pin to 1.8 V;
4. Use SPI write to set the "eFuse_ctrl" register value to 0x11;
5. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
6. Reconnect VDDQ pin to ground;
7. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
8. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

11.7.7 Program Byte

The procedure of a Program Byte operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
2. Use SPI write to set the "eFuse_addr" register value to the required address to be programmed;
3. Make sure the mirror register has the data to be programmed into eFuse;
4. Connect VDDQ pin to 1.8 V;
5. Use SPI write to set the "eFuse_ctrl" register value to 0x81;
6. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
7. Reconnect VDDQ pin to ground;
8. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
9. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

11.7.8 Program All

The procedure of a Program All operation is as below:

1. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
2. Make sure all mirror registers have the data to be programmed into eFuse;
3. Connect VDDQ pin to 1.8 V;
4. Use SPI write to set the "eFuse_ctrl" register value to 0x21;
5. Use SPI read to get the "eFuse_st_cmdDone" register value to make sure when the operation is complete ("eFuse_st_cmdDone" is "1");
6. Reconnect VDDQ pin to ground;
7. Use SPI write to set the "eFuse_ctrl" register value to 0x01;
8. Use SPI write to set the "eFuse_ctrl" register value to 0x00.

Note – In Program Bit, Program Byte and Program All operations, VDDQ should NOT be kept at 1.8 V for more than 40 ms.

11.8 Tx Channel and Hopping Modulation Settings

11.8.1 Setting of the Tx Channel

Transmitter modulates RF signal on 32 frequency channels in 200 kHz band, as given in Table 13. Write to the 0x0064 register (tx_hop_table[7:0]) the required Frequency Channel number (in binary form).

Table 13. Tx Frequency Channels

Tx Frequency Channel number	Tx Frequency Channel, Hz	Tx Frequency Channel number	Tx Frequency Channel, Hz
0	-97000	16	15000
1	-89000	17	11000
2	-83000	18	19000
3	-90000	19	29000
4	-79000	20	40000
5	-73000	21	37000
6	-59000	22	47000
7	-65000	23	53000
8	-53000	24	65000
9	-47000	25	59000
10	-37000	26	73000
11	-40000	27	79000
12	-29000	28	90000
13	-19000	29	83000
14	-11000	30	89000
15	-15000	31	97000

11.8.2 Tx Hopping Modulation

Transmitter generates a hopping message when Tx Frequency Hopping is enabled (tx_init[7] = 1). The message is transmitted being divided into 8 consecutive sections, each section on its own frequency channel.

Write to eight registers 0x0064 - 0x006B (tx_hop_table[7:0], tx_hop_table[15:8], etc.) the required Frequency Channel numbers chosen from the Table 13.

Table 14. Example of defining the Tx Frequency Hopping table

Tx Frequency Channel (Hz)	-90000	-65000	-40000	-15000	15000	40000	65000	90000
Section	0	1	2	3	4	5	6	7
Register address	0x0064	0x0065	0x0066	0x0067	0x0068	0x0069	0x006A	0x006B
Register value [Dec]	3	7	11	15	16	20	24	28

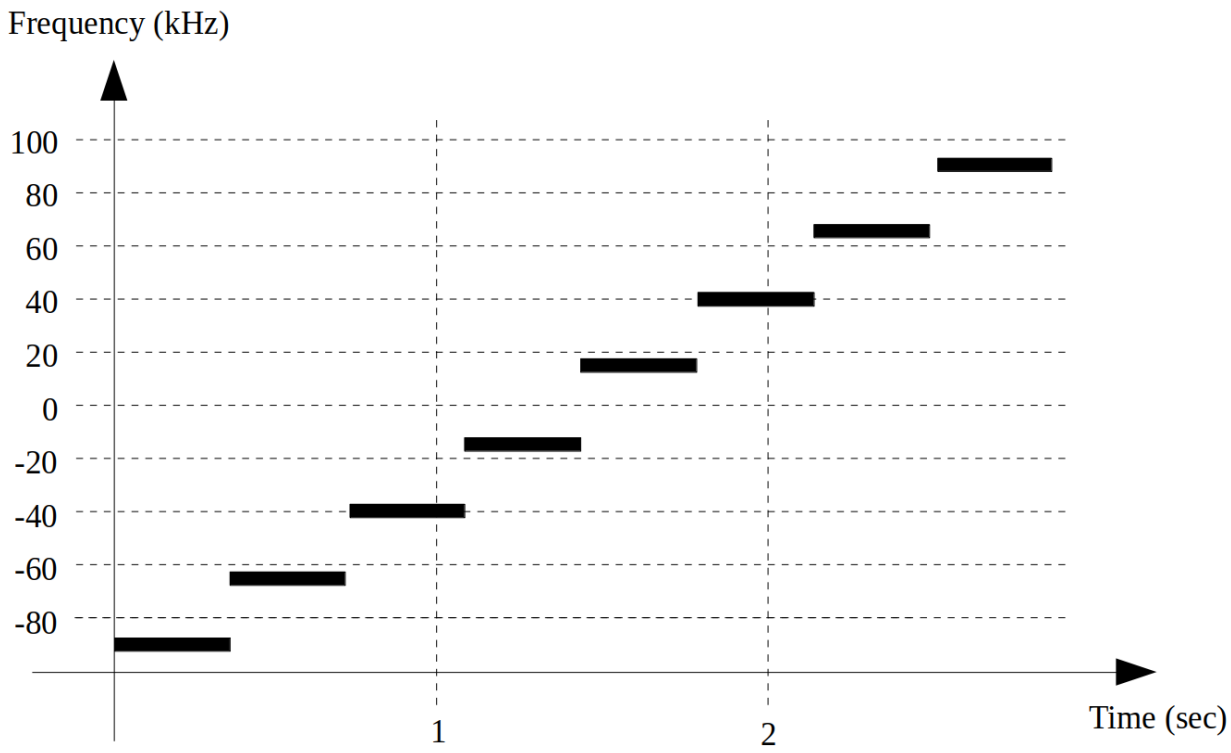


Figure 5. Example chart of Tx Frequency Hopping message

11.9 Rx Channel and Hopping Demodulation Settings

11.9.1 Setting of Rx Channels

Receiver demodulates RF signal at 8 frequency channels in 200 kHz band.

Write to the register 0x0032 (rx_hop_table[7:0]) the Value corresponding to the required Rx Frequency Channel, as it given in Table 15.

Table 15. Rx Frequency Channels

Rx Frequency Channel number	Rx Frequency Channel, Hz	Value for the data rate of 50, 400, 3200 bps	Value for the data rate of 25600 bps
7	-90000	111	1110000
6	-65000	110	1100000
5	-40000	101	1010000
4	-15000	100	1000000
3	15000	11	110000
2	40000	10	100000
1	65000	1	10000
0	90000	0	0

11.9.2 Rx Hopping Message Demodulation

Receiver demodulates RF hopping signal automatically if Rx Packet was received at data rate of 100 bps.

The Rx Packet is divided into 8 consecutive sections, each section on its own frequency channel.

Value to be written for each of register 0x0032 - 0x0035 (rx_hop_table) should be calculated based on the required Rx Frequency Channels for each section, as given in Table 16, where Ch_1 – Ch_7 are the Values for the required Rx Frequency Channels for sections 0 - 7 respectively, as given in the Table 17.

Table 16. Calculation formula for rx_hop_table registers values

Section	0	1	2	3	4	5	6	7
Register address	0x0032		0x0033		0x0034		0x0035	
Register value	Ch_0 <<4 + Ch_1		Ch_2 <<4 + Ch_3		Ch_4 <<4 + Ch_5		Ch_6 <<4 + Ch_7	

Table 17. Rx Frequency Channels

Rx Frequency Channel number	Rx Frequency Channel, Hz	Value ($Ch_{7,6,5,4,3,2,1,0}$)
7	-90000	111
6	-65000	110
5	-40000	101
4	-15000	100
3	15000	11
2	40000	10
1	65000	1
0	90000	0

Table 18. Example of defining the Rx Frequency Hopping table

Rx Frequency Channel (Hz)	-90000	-65000	-40000	-15000	15000	40000	65000	90000
Section	0	1	2	3	4	5	6	7
Register address	0x0032		0x0033		0x0034		0x0035	
Register value [Hex]	0x67		0x45		0x23		0x01	
Register value [Bin]	1100111		1000101		100011		1	

Code example of defining the Rx Frequency Hopping table:

```
uint8_t DEM_HOP_TABLE = 0x32;
uint8_t hop[8] = {7,6,5,4,3,2,1,0};
spi_write8(DEM_HOP_TABLE, (hop[1] << 4) | hop[0]);
spi_write8(DEM_HOP_TABLE+1, (hop[3] << 4) | hop[2]);
spi_write8(DEM_HOP_TABLE+2, (hop[5] << 4) | hop[4]);
spi_write8(DEM_HOP_TABLE+3, (hop[7] << 4) | hop[6]);
```

12 Description of the Registers

12.1 eFuse Mirror Registers

These are the registers that can be reloaded automatically at Chip startup from the pre-programmed eFuse every power up.

Table 19. K5553BB015 eFuse Mirror Registers

Addr [Hex]	Default [Bin]	Bit	Name	Description
4000	011	7:5	pll_ivco_ctrl	Trimming bias current in VCO
	01	4:3	pll_icp_ctrl	Trimming PLL-CP current
	1	2	pll_sdm_en	Enable FracN mode (SDM) of PLL
	1	1	pll_sdm_third	Enable SDM in 3rd order mode
	0	0	pll_sdm_fourth	Enable SDM in 4th order mode
4001	01	7:6	pll_vmux_sel	Control VCO control voltage if "pll_muxoverride_n" = 0
	000101	5:0	pll_adccal_1v2	Control ADC calibration clock frequency
4002	0	7	txrx_bandsel	Control the RF band of the TRx: 1 for 860 MHz - 925 MHz band 0 for 430 MHz - 500 MHz band
	0011	6:3	lna_ictrl	Trimming of bias current in LNA: 0011 for 860 MHz - 925 MHz band 0110 for 430 MHz - 500 MHz band
	11	2:1	lna_gctrl	Trimming of LNA gain: 00 for Min Gain 11 for Max Gain
	0	0	rx_mode	Not Used
4003	01	7:6	mixer_ictrl	Trimming of bias current in Rx Mixer Gm stage
	0101	5:2	mixer_tia_gctrl	Control of Rx mixer TIA gain: 0101 for 860 MHz - 925 MHz band 0111 for 430 MHz - 500 MHz band
	1	1	rx_occ_en	Enable Rx offset cancellation
	1	0	tx_occ_en	Enable Tx offset cancellation
4004	01	7:6	sadcrefbufftrim	Trimming of ADC reference voltage
	000111	5:0	sadci_trimmingman	Trimming of I-Channel ADC delay unit
4005	0	7	sadci_adccomprstplssel	Trimming of I-Channel ADC reset pulse: 0 for 10.5 ns 1 for 12 ns
	1	6	sadci_trimmingmansel	Select between auto & manual trimming of I-Channel ADC delay unit: 0 for auto calibration 1 for manual mode
	01	5:4	tmpsns_reftrim	Trimming of TMPSNS reference voltage
	0	3	vga_f3db_tune_override	Enable bypassing Rx LPF tuning

Addr [Hex]	Default [Bin]	Bit	Name	Description
	0	2	eFuse_fully_burned	Should be programmed as 1 to enable the Chip auto initialization
	10	1:0	vco_cap_tune	Trimming of VCO output frequency
4006	000111	7:2	sadcq_trimmingman	Trimming of Q-Channel ADC delay unit
	1	1	sadcq_trimmingmansel	Select between auto & manual trimming of Q-Channel ADC delay unit: 0 for auto calibration 1 for manual mode
	0	0	txmode	1 for Tx BPSK mode 0 for Tx IQ mode
4007	0000	7:4	tx_pcalib_q_c	Coarse trimming of Tx LO signal of Q-Channel
	0000	3:0	tx_pcalib_q_f	Fine trimming of Tx LO signal of Q-Channel
4008	0	7	vga_filt_4th_to_2nd	Not Used
	0	6	rxadc_adcpsen	Enable power saving mode in Rx ADC (Not Recommended)
	0	5	rxadc_adccompensel	Controls Rx ADC comparator enable in power saving mode
	0	4	rxadc_compvalidmode	Enable auto synchronous cycling of Rx ADC
	01	3:2	tia_ictrl	Trimming of Rx Mixer TIA bias current
	0	1	mixer_cm_ctrl	Trimming CM level of Rx I-Channel mixer
	0	0	gpadc_compvalidmode	Enable auto synchronous cycling of TMPSNS ADC
4009	01010101	7:0	bctrl [7:0]	Trimming Rx VGA bias current
400A	01	7:6	bctrl [9:8]	Trimming Rx VGA bias current
	0	5	gpadc_adccomprstplssel	Trimming of TMPSNS ADC reset pulse: 0 for 10.5 ns 1 for 12 ns
	1	4	gpadc_trimmingmansel	Select between auto & manual trimming of TMPSNS ADC delay unit: 0 for auto calibration 1 for manual mode
	0	3	tx_extpa_mode	Enable Tx with External PA mode
	0	2	txpa_ldo_byp	Enable bypass mode of PA LDO
	11	1:0	tx_cml_ibias_ctrl	Trimming bias current of Tx CML buffer in external PA mode
400B	0000	7:4	rx_pcalib_i_c	Coarse trimming of Rx LO signal of I-Channel
	0000	3:0	rx_pcalib_i_f	Fine trimming of Rx LO signal of I-Channel
400C	000111	7:2	gpadc_trimmingman	Trimming of TMPSNS ADC delay unit
	0	1	gpadc_adcpsen	Enable power saving mode of TMPSNS ADC (Not Recommended)
	0	0	gpadc_adccompensel	Controls TMPSNS ADC comparator enable in power saving mode

Addr [Hex]	Default [Bin]	Bit	Name	Description
400D	0000	7:4	vga_f3db_tune	Rx LPF tune manual control
	0000	3:0	txdac_f3db_tune	Tx LPF tune manual control
400E	1	7	clkgenrst	Reset of ADC clock generator
	0	6	clkgenen	Enable of ADC clock generator
	000000	5:0	sampleclkpw1	Trimming pulse width of Rx ADC sampling clock
400F	000000	7:2	sampleclkperiod1	Trimming frequency of Rx ADC sampling clock
	00	1:0	lna_cap_tune	Trimming frequency tuning of LNA
4010	000000	7:2	sampleclkpw2	Trimming pulse width of TMPSNS ADC sampling clock
	0	1	paldo_cc_ctrl	Enable PA LDO output compensated mode (Not Tested)
	0	0	rx_lna_ldo_mode	Enable high load current of LNA LDO (Not Tested)
4011	000000	7:2	sampleclkperiod2	Trimming frequency of Rx ADC sampling clock
	0	1	pa_mode_ext	Enable 3 V PA LDO output if "pa_mode_bypass" = 1
	0	0	pa_mode_bypass	Bypass internal auto control of PA LDO output voltage level
4012	000000	7:2	compenpw1	Control Rx ADC comparator enable pulse in power saving mode
	00	1:0	tx_sel_txmixer_1v2	LSB controls of TX mixer CM level
4013	000000	7:2	compenpw2	Control TMPSNS ADC comparator enable pulse in power saving mode
	0	1	rx_mixer_ldo_1p4v_ctrl	Enable 1.4 V mode of Mixer LDO
	0	0	sadcq_adccomprstplssel	Trimming of Rx Q-Channel ADC reset pulse: 0 for 10.5 ns 1 for 12 ns
4014	0000	7:4	rx_pcalib_q_c	Coarse trimming of Rx LO signal of Q-Channel
	0000	3:0	rx_pcalib_q_f	Fine trimming of Rx LO signal of Q-Channel
4015	0000	7:4	tx_pcalib_i_c	Coarse trimming of Tx LO signal of I-Channel
	0000	3:0	tx_pcalib_i_f	Fine trimming of Tx LO signal of I-Channel
4016	000000	7:2	sclk_ndiv	Control Tx DAC sampling clock frequency
	1	1	dem_lfsr_en	Enable Tx DAC LFSR mode for DEM
	1	0	dem_cntr_en	Enable Tx DAC normal counter mode for DEM
4017	0	7	lprlxosc_en	Enable 1 kHz LP RCO
	0000000	6:0	lprlxosc_ftrim	Trimming frequency of 1 kHz LP RCO
4018	0	7	hprlxosc_en	Enable 20 MHz RCO
	0000000	6:0	hprlxosc_ftrim	Trimming frequency of 20 MHz RCO

Addr [Hex]	Default [Bin]	Bit	Name	Description
4019	111	7:5	lpref_vtrim	Trimming LDODIG LP reference voltage
	1111	4:1	lpref_itrim	Trimming LDODIG LP reference current
	0	0	lpref_1p2en	Enable 1.2 V mode of LDODIG
401A	00000000	7:0	chp_id[7:0]	Reserved for Chip ID
401B	00000000	7:0	chp_id[15:8]	Reserved for Chip ID
401C	00000000	7:0	chp_id[23:16]	Reserved for Chip ID
401D	00000000	7:0	chp_id[31:24]	Reserved for Chip ID
401E	0	7	padsmt	Enable Schmitt trigger in digital IO pads
	0	6	pade1	Trimming drive strength of digital IO pads
	0	5	pade2	
	0	4	padpos	Power on start control of digital IO pads: 0 – active pull down disabled for loss of core power 1 – active pull down enabled for loss of core power
	0	3	padsr	Slew rate control of digital IO pads: 0 for slow (half frequency) 1 for fast
	0	2	padp2	Disabled state control of digital IO pads [Hi-Z/Pull-up/Pull-down/Repeater]
	0	1	padp1	
	0	0	padsel18	Pad-frame control: 1 for DVDD ≤ 1.8 V 0 for DVDD ≥ 2.5 V
401F	1		mixer_cm_ctrl_q	Trimming CM level of Rx Q mixer
	1		rx_ldo_mixer_en	Enable mixer LDO
	1		tx_sel_txmixer_1v2[2]	MSB of trimming Tx mixer CM level
	1		vcoldo_1p4enb	Enable VCO LDO 1.4 V mode (active low)
	1		lnaldo_1p4enb	Enable LNA LDO 1.4 V mode (active low)
	1		lckdet_mode	Define mode of auto PLL lock detection: 0 for narrow range detection 0.4 V - 0.8 V 1 for wide range detection 0.2 V - 1 V
	1		lckdet_en	Enable auto PLL lock detection
	1		tx_vga_ictrl[1]	MSB of Tx VGA bias current trimming

12.2 Non-eFuse Writable Registers

These are the registers that can be changed by SPI write command, but they have no eFuse mirror and can't be loaded automatically at Chip power up.

SPI write commands should be executed after power up for any change of value required in these registers.

Table 20. K5553BB015 Non-eFuse Writable Registers

Addr [Hex]	Default [Bin]	Bit	Name	Description
0000		7:0	rx_msg[7:0]	Received message. After reading of the data it's required to write "0" in this register.
...		...	...	
001F		7:0	rx_msg[255:248]	
0020		7	rx_control[7]	IRQ flag
		6	rx_control[6]	FFT buffer ready
		5	rx_control[5]	Not Used
		4	rx_control[4]	Not Used
		3	rx_control[3]	Not Used
		2	rx_control[2]	Not Used
		1	rx_control[1]	Not Used
		0	rx_control[0]	Rx control: 1 for stop Rx and enable access to the registers 0x0021 - 0x0032, 0 for start Rx. Set required values to the registers 0x0021 - 0x0032 only when rx_control[0] = 1.
0021	0	2:0	rx_mode	000 for Rx data rate of 50 bps 001 for Rx data rate of 400 bps 010 for Rx data rate of 3200 bps 011 for Rx data rate of 25600 bps 100 for Rx data rate of 100 bps with Frequency Hopping
0022	0	7:0	rx_sync_th[7:0]	Rx Detection threshold. Possible values from 0 to 2047[Dec]. The 800[Dec] is recommended.
0023	100	2:0	rx_sync_th[10:8]	
		7:3	rx_sync_th[15:11]	Not Used
0024	10000000	7:0	rx_noise_start_bit	Message noise estimator start bit. The 128[Dec] is recommended.
0025	101	4:0	rx_alpha_shift	Alpha filter shift
0026	100100	7:0	rx_hop_lenght	Rx hop length: 100100 for 36 bits 101000 for 40 bits
0027	0	4:0	rx_mbist_on	Rx text mode: 0 for DFT mode 1 for MBIST mode
0028	10010111	7:0	rx_preamble_id[7:0]	Rx Packet Preamble
0029	10101	7:0	rx_preamble_id[15:8]	
002A	1111010	7:0	rx_preamble_id[23:16]	

Addr [Hex]	Default [Bin]	Bit	Name	Description
002B	1101111	7:0	rx_preamble_id[31:24]	
002C	1111	4:0	rx_fft_msb[4:0]	Most significant bit of 16-bits FFT sample to be read: from 15 to 23. The 23 is recommended (so data from 8 to 23 bit of FFT sample will be read). 32 FFT samples will be available in registers 0x0080 - 0x00FC (rrx_fft_read_buf[1023:0]).
	0	5	rx_fft_msb[5]	Not Used
	0	6	rx_fft_msb[5]	Not Used
	1	7	rx_fft_msb[7]	FFT sample's source: 0 for signal quadrature, 1 for signal magnitude. The 1 is recommended.
002E	10110111	7:0	rx_crc[7:0]	CRC polynomial for calculation the Rx Packet Checksum
002F	11101	7:0	rx_crc[15:8]	
0030	11000001	7:0	rx_crc[23:16]	
0031	100	7:0	rx_crc[31:24]	
0032	1100111	7:0	rx_hop_table[7:0]	Rx hopping mapping table. See section 11.9 for details.
0033	1000101	7:0	rx_hop_table[15:8]	
0034	100011	7:0	rx_hop_table[23:16]	
0035	1	7:0	rx_hop_table[31:24]	
0038		7	tx_init[7]	Enable Tx Frequency Hopping
		6	tx_init[6]	Abort transfer
		5	tx_init[5]	Set Tx hop length: 0 for 36 bits (for the message size of 288 bits) 1 for 40 bits (for the message size of 320 bits)
		4	tx_init[4]	Enable Tx Sent IRQ
		3	tx_init[3]	Enable Tx Error IRQ
		2	tx_init[2]	Not Used
		1	tx_init[1]	Clear Tx IRQ
		0	tx_init[0]	Start Tx
0039		7:0	tx_period[7:0]	TX rate control: tx_period = TX_SPEED/1000000 * (1<<24); TX_SPEED is a data rate in bps, from 1 to 25600
003A		7:0	tx_period[15:8]	
003B		7:0	tx_period[23:16]	
003C		7:0	tx_data[7:0]	Data to send via IQ modulator. Use message size of 288 bits if tx_init[5] = 0, 320 bits if tx_init[5] = 1.
...		…	…	
0063		7:0	tx_data[319:312]	
0064		7:0	tx_hop_table[7:0]	Tx hopping mapping table. See section 11.8 for details.
...		7:0	…	
006B		7:0	tx_hop_table[63:56]	

Addr [Hex]	Default [Bin]	Bit	Name	Description
0080		31:0	rx_fft_read_buf[31:0]	Spectrum buffer. Spectrum buffer is only possible to read at Rx data rate of 50 bps (rx_mode = 000) and when rx_control[6] = 1. The data from 31 to 16 bit of each register contains the imaginary part of FFT magnitude/ quadrature, the data from 15 to 0 bit of each register contains the real part of FFT magnitude/quadrature. Read the 1024-bits buffer and then write "0" to this register.
0084		31:0	rx_fft_read_buf[63:32]	
...		…	…	
0F8		31:0	rrx_fft_read_buf[991:960]	
00FC		31:0	rrx_fft_read_buf[1023:992]	
4020	00000101	7:0	eFuse_addr	eFuse bit address
4021	0	7	eFuse_ctrl_doProgByte	Enable eFuse program byte
	0	6	eFuse_ctrl_doReloadByte	Enable eFuse reload byte
	0	5	eFuse_ctrl_doProgAll	Enable eFuse program all
	0	4	eFuse_ctrl_doProg	Enable eFuse program bit
	0	3	eFuse_ctrl_doReload	Enable eFuse reload all
	0	3	eFuse_ctrl_doRead	Enable eFuse read byte
	10	1:0	eFuse_ctrl_Mode	Control mode of eFuse
4022	0	7	pll_cal_add_sub	Add/Sub from VCO tuner output
	00	6:5	pll_cal_offset	Insert offset from VCO tuner output
	10000	4:0	pll_cal_tune	Manual control of VCO instead of VCO tuner
4023	100001	7:2	pll_nint	PLL integer division ratio
	00	1:0	pll_icp_offset	PLL CP offset control
4024	1	7	pll_1gdiv2mux_1gdiv2_en	Enable 1 GHz post VCO divider
	1	6	pll_1gdiv2mux_en	Enable 1 GHz post VCO mux
	1	5	pll_2gdiv2_en	Enable 2 GHz post VCO divider
	1	4	pll_cp_en	Enable PLL CP
	1	3	pll_cp_nmirror_en	Enable PLL CP NMOS
	1	2	pll_cp_opamp1_en	Enable PLL CP 1st opamp
	1	1	pll_cp_opamp2_en	Enable PLL CP 2nd opamp
	1	0	pll_cp_pmirror_en	Enable PLL CP PMOS
4025	1	7	pll_en	Enable PLL
	1	6	pll_pfd_en	Enable PLL PFD
	1	5	pll_buffer_en	Enable PLL buffer
	1	4	pll_ldo_en	Enable PLL LDO
	1	3	pll_vco_ibias_en	Enable PLL VCO bias current
	1	2	pll_vco_en	Enable PLL VCO

Addr [Hex]	Default [Bin]	Bit	Name	Description
	1	1	pll_vco_varactor_en	Enable PLL VCO varactor
	1	0	pll_vco_capbank_en	Enable PLL VCO capacitor bank
4026	00000000	7:0	pll_nfrac[7:0]	PLL fractional division ratio
4027	00000000	7:0	pll_nfrac[15:8]	PLL fractional division ratio
4028	0	7		Not Used
	0000000	6:1	pll_nfrac[21:16]	PLL fractional division ratio
	1	0	tx_mixer_en	Enable Tx mixer
4029	1000	7:4	vga1_gctrl	Control gain of 1st stage in Rx VGA: 0000 for max gain 1000 for min gain
	1000	3:0	vga2_gctrl	Control gain of 2nd stage in Rx VGA: 0000 for max gain 1000 for min gain
402A	011	7:5	txmixer_gctrl	Control gain of Tx CML buffer
	1	4	tx_en	Enable Tx
	0	3	bb_tuner_start	Enable LPF tuner
	1	2	tx_bb_dac_en	Enable Tx DAC
	1	1	tx_bb_dac_lpf_en	Enable Tx LPF
	0	0	tx_vga_ictrl[0]	LSB of Tx VGA bias current control
402B	1	7	sadcrefbuffen	Enable ADC reference buffer
	0	6	sadcrefbuff_highz	Enable Hi-Z mode of ADC reference buffer
	1	5	sadci_adcen	Enable Rx I-Channel ADC
	0	4	Spare_wr_C_1	Not Used
	1	3	sadcq_adcen	Enable Rx Q-Channel ADC
	0	2	gpadc_adcen	Enable TMPSNS ADC
	0	1	gpadc_calbstart	Rising edge trigger of TMPSNS ADC manual calibration
	0	0	gpadc_calbseqstart	Rising edge trigger of TMPSNS ADC auto calibration
402C	1	7	rx_biquad_en	Enable Rx VGA
	1	6	rx_biquad_filt1_en	Enable Rx 1st LPF stage
	1	5	rx_biquad_filt2_en	Enable Rx 2nd LPF stage
	1	4	rx_ldo_en	Enable LNA LDO
	1	3	rx_lna_en	Enable LNA
	1	2	rx_mixer_en	Enable Rx Mixer
	1	1	rx_mixer_gmstage_en	Enable Rx Mixer Gm stage
	1	0	rx_mixer_tia_en	Enable Rx Mixer TIA
402D	000	7:5	txdac_gctrl	Control Tx DAC output swing

Addr [Hex]	Default [Bin]	Bit	Name	Description
	00000	4:0	txpa_gctrl	Control Tx PA output power: 10101 for 16 dBm power 10000 for 8 dBm power
402E	1	7	tx_pa_en	Enable Tx PA
	1	6	tx_bb_bias_en	Enable Tx baseband bias current
	1	5	tx_bb_dac_10b_en	Enable Tx DAC core
	1	4	tx_bb_en	Enable Tx baseband
	1	3	tx_ldopa_en	Enable PA LDO
	1	2	tx_mixer_cml_opamp_en	Enable CML opamp
	1	1	tx_mixer_cml_en	Enable CML stage
	1	0	tx_mixer_dif2s_en	Enable Tx post mixer buffers
402F	1	7	txrx_poff	Enable deep sleep mode
	0	6	txrx_sleep	Enable sleep mode
	0	5	txrx_idle	Enable Idle mode
	0	4	txon_rxoff	1 for Tx mode 0 for Rx mode
	0	3	ana_test_mux_en	Not Used
	000	2:0	ana_test	Not Used
4030	1	7	pll_muxoverride_n	Enable VCO control voltage manual selection (active low)
	0	6	pll_sel_loop	Enable PLL closed loop
	0	5	pll_cal_start	Rising edge trigger of PLL auto tuning (VCO tuner)
	0	4	pll_cal_bp	Bypass VCO tuner
	0	3	adc_calclken	Enable ADC calibration clock
	0	2	rxadc_calbstart	Rising edge trigger for manual calibration of Rx ADCs
	1	1	rx_en	Enable Rx
	0	0	tmpsns_en	Enable TMPSNS
4031	0	7	ldodig_highz	Enable Hi-Z mode of LDODIG
	0	6	ldogp_en	Enable auxiliary LDO for TMPSNS & TMUX operation
	0	5	RXADC_CALBSEQSTART	Rising edge trigger for auto calibration of Rx ADCs
	0000	4:1	DMUX_MODE	Define DMUX mode
	0	0	DMUX_EN	Enable DMUX
4032	0000	7:4	TMUX_MODE	Define TMUX mode
	0	3	TMUX1_EN	Enable TMUX
	0	2	TMUX2_EN	Not Used

Addr [Hex]	Default [Bin]	Bit	Name	Description
	0	1	TMUX3_EN	Not Used
	0	0	TMUX4_EN	Not Used
4033	000000	7:2	Spare_wr_B_8[7:2]	Not Used
	0	1	ldomixer_byp	Bypass mixer LDO
	0	0	adc_tst_en	Enable testing mode of TMPSNS ADC

12.3 Read Only Registers

These are the registers that can only be read by SPI read command and they have no eFuse mirror.

Table 21. K5553BB015 Read Only Registers

Addr [Hex]	Bit	Name	Description
0038	7:0	tx_status[7:0]	Transmitted bits counter[7:0]
0039	7	tx_status[15]	1 for Tx in progress
	6	tx_status[14]	1 for Hop repeat
	5	tx_status[13]	Not Used
	4	tx_status[12]	1 for Tx IRQ flag was set
	3	tx_status[11]	1 for Tx error flag was set
	2	tx_status[10]	Not Used
	1	tx_status[9]	Not Used
	0	tx_status[8]	Transmitted bits counter[8]
003A	7:0	tx_total_sent	Transmitted packets counter
003B	7:0	tx_error_cnt	Transmission error counter
003C	7:0	tx_cancel_cnt	Counter of cancelled Tx packets
003F	7:0	Current output I[7:0]	I-Channel current value
0040	7:0	Current output I[15:8]	
0041	7:0	Current output Q[7:0]	Q-Channel current value
0042	7:0	Current output Q[15:8]	
4034	7:3	pll_vco_calout	VCO tuner output
	2	pll_vco_calibdone	Flag of VCO tuning operation
	1	tune_f3dB_rx_caldone	Flag of Rx LPF tuning operation
	0	tune_f3dB_tx_caldone	Flag of Tx LPF tuning operation
4035	7:4	tune_f3dB_rx	Rx LPF tuner output
	3:0	tune_f3dB_tx	Tx LPF tuner output
4036	7:0	sadci_adcout[7:0]	Output of Rx I-Channel ADC
4037	7:4	sadci_adcout[11:8]	
	3	sadci_calbdone	Flag of Rx I-Channel ADC calibration
	2	sadci_calbfailed	Flag of Rx I-Channel ADC calibration failure
	1	rxadc_calbseqdone	Flag of Rx ADCs auto calibration
	0	gpadc_calbseqdone	Flag of TMPSNS ADC auto calibration
4038	7:0	sadcq_adcout[7:0]	Output of Rx Q-Channel ADC
4039	7:4	sadcq_adcout[11:8]	
	3	sadcq_calbdone	Flag of Rx Q-Channel ADC calibration
	2	sadcq_calbfailed	Flag of Rx Q-Channel ADC calibration failure
	1	eFuse_st_cmdDone	Flag to indicate end of eFuse operation

Addr [Hex]	Bit	Name	Description
	0	regfile_init_done	Flag to indicate Chip initialization
403A	7:0	gpadc_adcout[7:0]	Output of TMPSNS ADC
403B	7:4	gpadc_adcout[11:8]	
	3	gpadc_calbdone	Flag of Rx TMPSNS ADC calibration
	2	gpadc_calbfailed	Flag of Rx TMPSNS ADC calibration failure
	1	adc_clk_ready	Flag of PLL locking (1: PLL is locked)
	0	pmode_switch_busy	Flag to indicate Chip modes transition is complete (0 for transition is complete)
403C	7:0	eFuse_Read_Data	eFuse read data output of read operation
403D	7:2	gpadc_trimmingcalb	Output of TMPSNS ADC calibration
	1:0	spare_rd_B_2	Not Used
403E	7:2	sadcq_trimmingcalb	Output of Rx Q-Channel ADC calibration
	1:0	spare_rd_C_2	Not Used
403F	7:2	sadci_trimmingcalb	Output of Rx I-Channel ADC calibration
	1:0	spare_rd_D_2	Not Used

13 Packaging Information

13.1 Package Outline Drawing

The K5553BB015 transceiver is available in a 32-lead QFN package as shown in Figure 6.

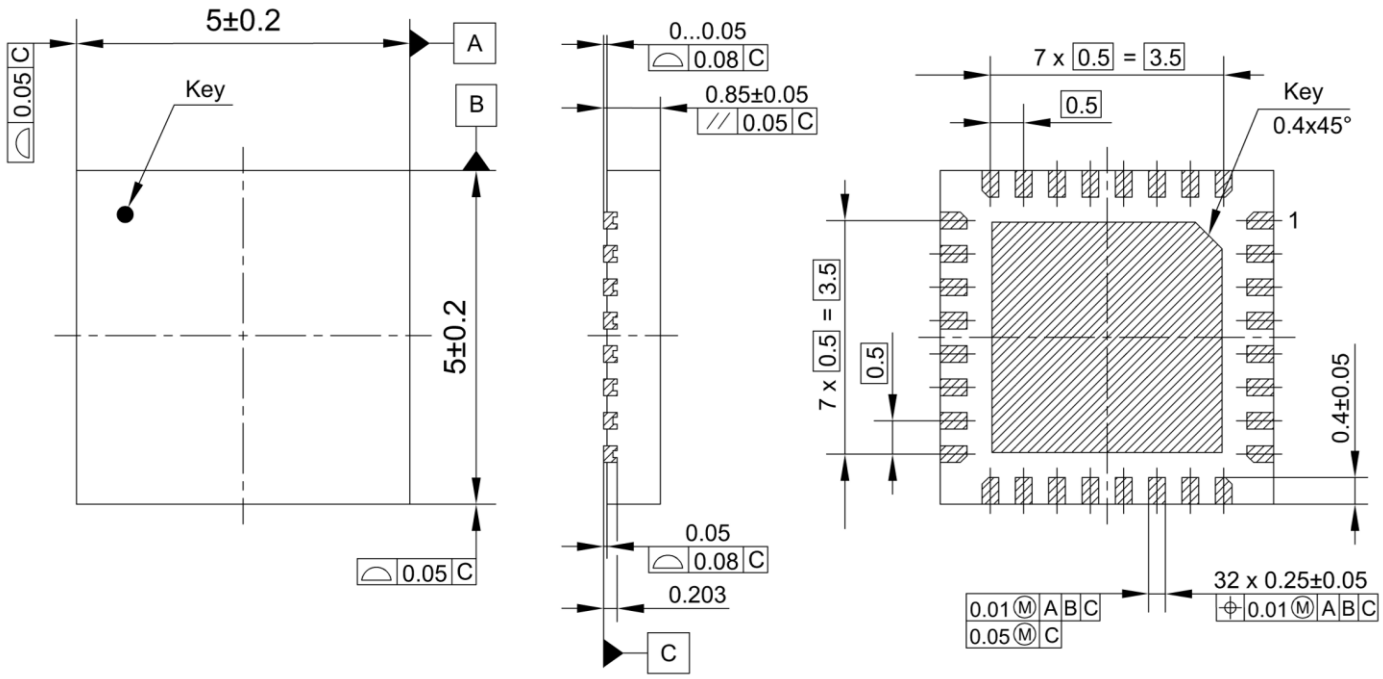


Figure 6. Package Outline Drawing

13.2 Package Marking

Figure 7. Marking Diagram

Table 22. K5553BB015 marking

Row No.	Label	Description
1	NB-Fi	Name of the supported IoT communication standard, a registered trademark
2	K5553BB015	Part Number
3	B001YYWW	B0 – Chip version
		01 – Package Version
		YYWW – Date of assembly in the form of Year, Week Number

13.3 Tape & Reel Information

The K5553BB015 transceivers are supplied in reels of 5 000 pieces, or in tapes for quantities not multiples of 5 000 pieces. The K5553BB015 is marked as shown in Figure 7. Description of labels on the K5553BB015 transceiver marking is given in Table 22. Tape & Reel Dimensions and Box Dimensions are provided in Figures 8 and 9.

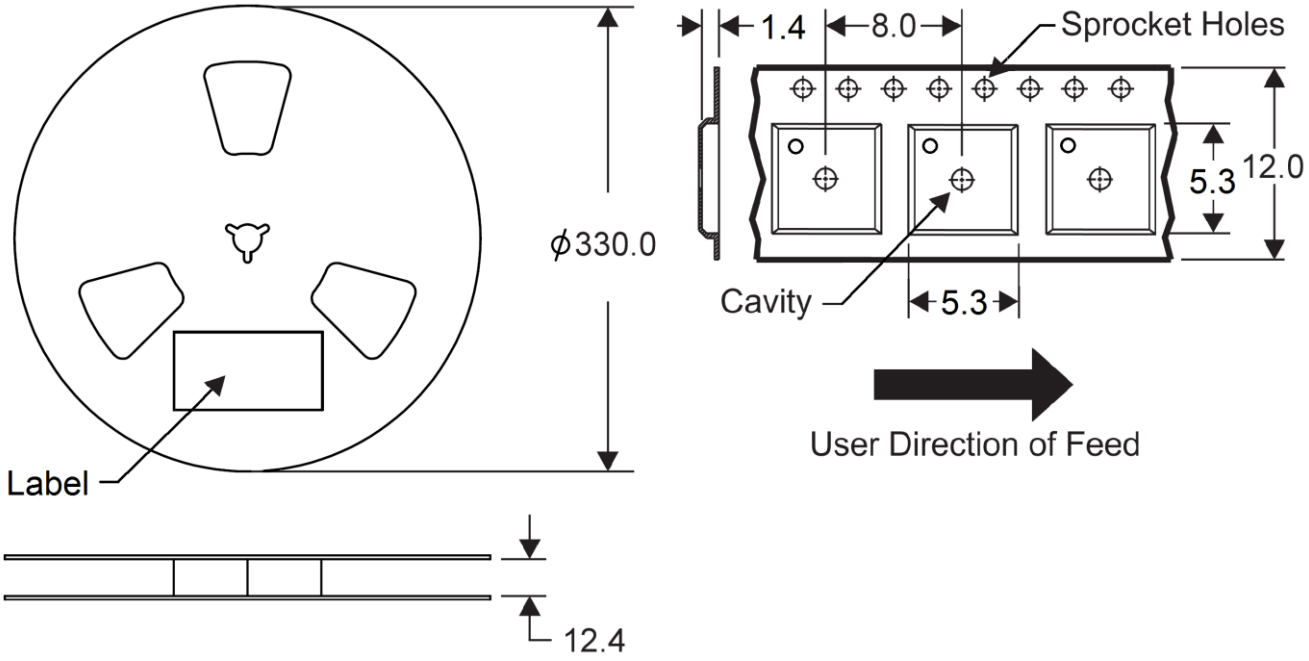


Figure 8. Tape & Reel Dimensions

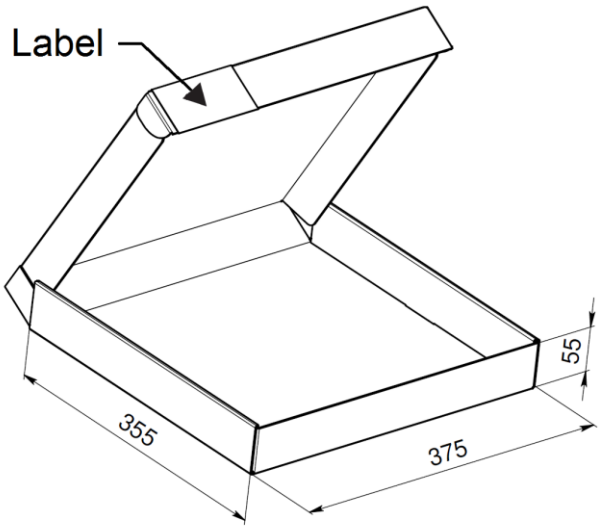


Figure 9. Tape & Reel Box Dimensions

14 Contact and Ordering Information

WAVIoT Integrated Systems, LLC

United States of America

Business ID: DL133522

E-MAIL: SALES@WAVIOT.COM