Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

HCS370

KEELOQ® Code Hopping Encoder

FEATURES

Security

- Two programmable 32-bit serial numbers
- Two programmable 64-bit encoder keys
- Two programmable 60-bit seed values
- Each transmission is unique
- 67/69-bit transmission code length
- 32-bit hopping code
- Crypt keys are read protected

Operating

- 2.05-5.5V operation
- Six button inputs
- 15 functions available
- Four selectable baud rates
- Selectable minimum code word completion
- Battery low signal transmitted to receiver
- Nonvolatile synchronization data
- PWM, VPWM, PPM, and Manchester modulation
- Button queue information transmitted
- Dual Encoder functionality

Other

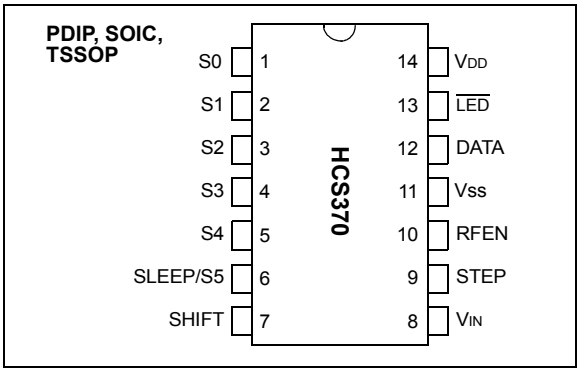
- On-chip EEPROM
- On-chip tuned oscillator (±10% over voltage and temperature)
- Button inputs have internal pull-down resistors
- $\overline{\text{LED}}$ output
- PLL control for ASK and FSK
- Low external component count
- Step-up voltage regulator

Typical Applications

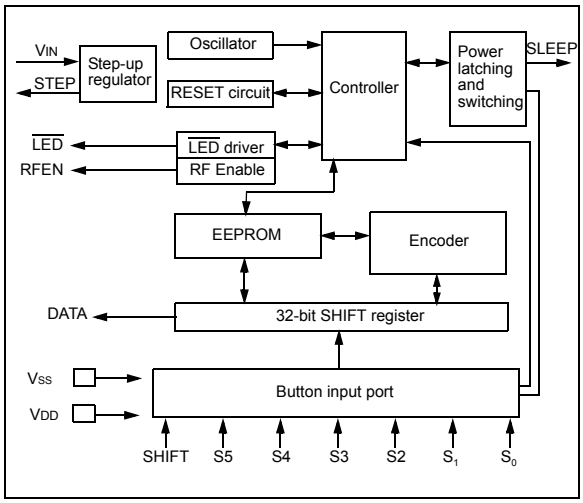
The HCS370 is ideal for Remote Keyless Entry (RKE) applications. These applications include:

- Automotive RKE systems
- Automotive alarm systems
- Automotive immobilizers
- Gate and garage door openers
- Identity tokens
- Burglar alarm systems

PACKAGE TYPES



HCS370 BLOCK DIAGRAM



GENERAL DESCRIPTION

The HCS370 is a code hopping encoder designed for secure Remote Keyless Entry (RKE) and secure remote control systems. The HCS370 utilizes the KEELOQ® code hopping technology, which incorporates high security, a small package outline, and low cost to make this device a perfect solution for unidirectional authentication systems and access control systems.

The HCS370 combines a hopping code generated by a nonlinear encryption algorithm, a serial number, and status bits to create a secure transmission code. The length of the transmission eliminates the threat of code scanning and code grabbing access techniques.

The crypt key, serial number, and configuration data are stored in an EEPROM array which is not accessible via any external connection. The EEPROM data is programmable but read protected. The data can be verified only after an automatic erase and programming operation. This protects against attempts to gain access to keys or manipulate synchronization values. In addition, the HCS370 supports a dual encoder. This allows two manufacturers to use the same device without having to use the same manufacturer's code in each of the encoders. The HCS370 provides an easy to use serial interface for programming the necessary keys, system parameters, and configuration data.

1.0 SYSTEM OVERVIEW

Key Terms

The following is a list of key terms used throughout this data sheet. For additional information on KEELOQ and code hopping, refer to Technical Brief (TB003).

- **RKE** - Remote Keyless Entry
- **Button Status** - Indicates what button input(s) activated the transmission. Encompasses the 6 button status bits S5, S4, S3, S2, S1 and S0 (Figure 3-2).
- **Code Hopping** - A method by which a code, viewed externally to the system, appears to change unpredictably each time it is transmitted.
- **Code Word** - A block of data that is repeatedly transmitted upon button activation (Figure 3-2).
- **Transmission** - A data stream consisting of repeating code words (Figure 4-1).
- **Crypt Key** - A unique and secret 64-bit number used to encrypt and decrypt data. In a symmetrical block cipher such as the KEELOQ algorithm, the encryption and decryption keys are equal and will therefore be referred to generally as the crypt key.
- **Encoder** - A device that generates and encodes data.
- **Encryption Algorithm** - A recipe whereby data is scrambled using a crypt key. The data can only be interpreted by the respective decryption algorithm using the same crypt key.
- **Decoder** - A device that decodes data received from an encoder (i.e., HCS5XX).
- **Decryption Algorithm** - A recipe whereby data scrambled by an encryption algorithm can be unscrambled using the same crypt key.
- **Learn** – Learning involves the receiver calculating the transmitter's appropriate crypt key, decrypting the received hopping code and storing the serial number, synchronization counter value, and crypt key in EEPROM. The KEELOQ product family facilitates several learning strategies to be implemented on the decoder. The following are examples of what can be done.
  - **Simple Learning**
    The receiver uses a fixed crypt key. The crypt key is common to every component used by the same manufacturer.
  - **Normal Learning**
    The receiver derives a crypt key from the encoder serial number. Every transmitter has a unique crypt key.
  - **Secure Learning**
    The receiver derives a crypt key from the encoder seed value. Every encoder has a unique seed value that is only transmitted by a special button combination.
- **Manufacturer's Code** – A unique and secret 64-bit number used to derive crypt keys. Each encoder is programmed with a crypt key that is a function of the manufacturer's code. Each decoder is programmed with the manufacturer code itself.

The HCS370 code hopping encoder is designed specifically for keyless entry systems. In particular, typical applications include vehicles and home garage door openers. The encoder portion of a keyless entry system is integrated into a transmitter carried by the user. The transmitter is operated to gain access to a vehicle or restricted area. The HCS370 is meant to be a cost-effective yet secure solution to such systems requiring very few external components (Figure 2-1).

Most low end keyless entry transmitters are given a fixed identification code that is transmitted every time a button is pushed. The number of unique identification codes in a low end system is usually a relatively small number. These shortcomings provide an opportunity for a sophisticated thief to create a device that 'grabs' a transmission and retransmits it later or a device that quickly 'scans' all possible identification codes until the correct one is found.

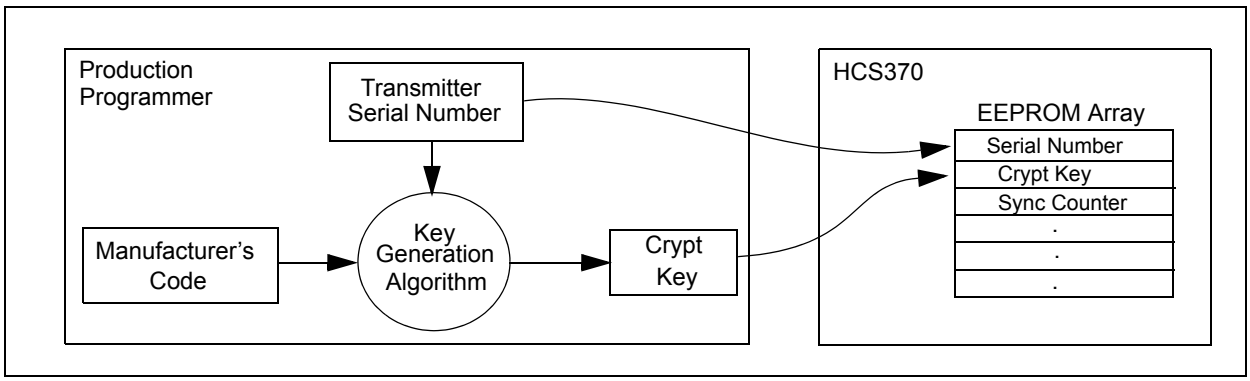
The HCS370, on the other hand, employs the KEELOQ code hopping technology coupled with a transmission length of 67 bits to virtually eliminate the use of code 'grabbing' or code 'scanning'. The high security level of the HCS370 is based on the patented KEELOQ technology. A block cipher based on a block length of 32 bits and a key length of 64 bits is used. The algorithm obscures the information in such a way that if a single hopping code data bit changes (before encryption), statistically more than 50% of the encrypted data bits will change.

As indicated in the block diagram on page one, the HCS370 has a small EEPROM array which must be loaded with several parameters before use; most often programmed by the manufacturer at the time of production. The most important of these are:

- A serial number, typically unique for every encoder
- A crypt key
- An initial synchronization value

The crypt key generation typically inputs the transmitter serial number and 64-bit manufacturer's code into the key generation algorithm (Figure 1-1). The manufacturer's code is chosen by the system manufacturer and must be carefully controlled as it is a pivotal part of the overall system security.

FIGURE 1-1: CREATION AND STORAGE OF CRYPT KEY DURING PRODUCTION



The synchronization counter is the basis behind the transmitted code word changing for each transmission; it increments each time a button is pressed. Each increment of the synchronization value results in more than 50% of the hopping code bits changing.

Figure 1-2 shows how the key values in EEPROM are used in the encoder. Once the encoder detects a button press, it reads the button inputs and updates the synchronization counter. The synchronization counter and crypt key are input to the encryption algorithm and the output is 32 bits of encrypted information. This data will change with every button press while its value will appear to 'randomly hop around'. Hence, this data is referred to as the hopping portion of the code word. The 32-bit hopping code is combined with the button information and serial number to form the code word transmitted to the receiver. The code word format is explained in greater detail in Section 4.1.

A receiver may use any type of controller as a decoder. Typically, it is a microcontroller with compatible firmware that allows the decoder to operate in conjunction with an HCS370 based transmitter.

A transmitter must first be 'learned' by the receiver before its use is allowed in the system. Learning includes calculating the transmitter's appropriate crypt key, decrypting the received hopping code, storing the serial number, storing the synchronization counter value, and storing crypt key in EEPROM.

In normal operation, each received message of valid format is evaluated. The serial number is used to determine if it is from a learned transmitter. If the serial number is from a learned transmitter, the message is decrypted and the synchronization counter is verified. Finally, the button status is checked to see what operation is requested. Figure 1-3 shows the relationship between some of the values stored by the receiver and the values received from the transmitter.

For detailed decoder operation, see Section 7.0.

FIGURE 1-2: BUILDING THE TRANSMITTED CODE WORD (ENCODER)

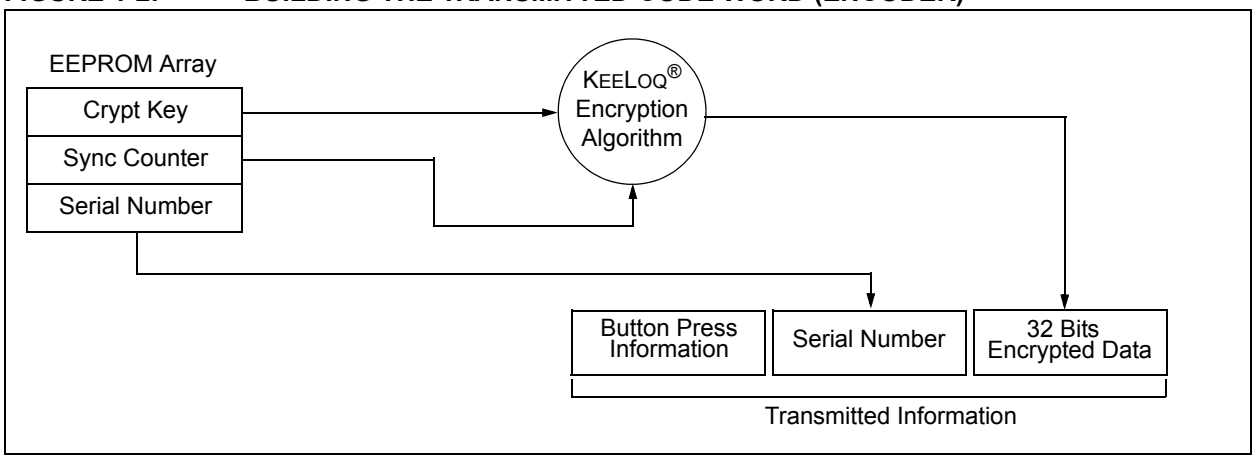
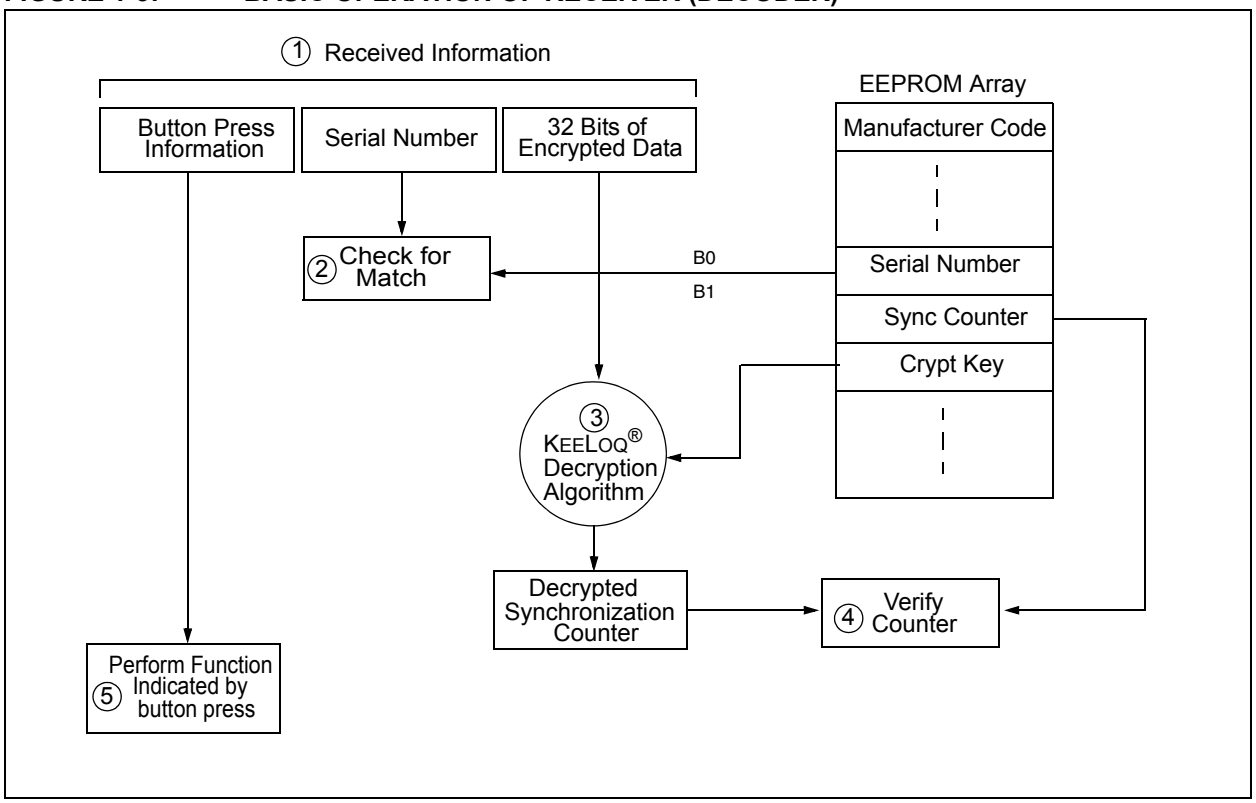


FIGURE 1-3: BASIC OPERATION OF RECEIVER (DECODER)



NOTE: Circled numbers indicate the order of execution.

2.0 DEVICE DESCRIPTION

As shown in the typical application circuits (Figure 2-1), the HCS370 is an easy device to use. It requires only the addition of buttons and RF circuitry for use as the encoder in your security application. A description of each pin is described in Table 2-1. Refer to Figure 2-3 for information on the I/O pins.

Note:	S0-S5 and SHIFT inputs have pull-down resistors. VIN should be tied high if the step-up regulator is not used.

TABLE 2-1: PIN DESCRIPTIONS

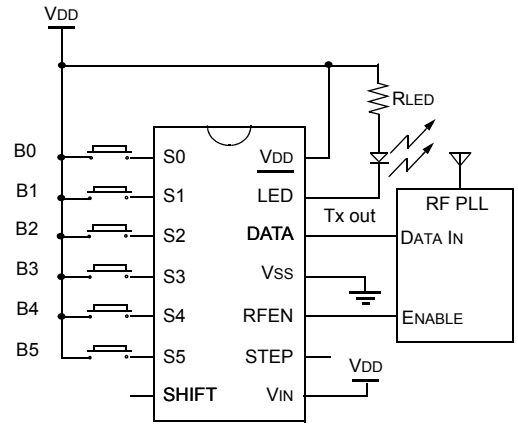
Name	Pin Number	Description
S0	1	Switch input S0
S1	2	Switch input S1
S2	3	Switch input S2
S3	4	Switch input S3
S4	5	Switch input S4
S5/SLEEP	6	Switch input S5, or SLEEP output
SHIFT	7	SHIFT input
VIN	8	Step-up regulator input
STEP	9	Step-up pulses output
RFEN	10	RF enable output
VSS	11	Ground reference
DATA	12	Transmission output pin
$\overline{\text{LED}}$	13	Open drain output for $\overline{\text{LED}}$ with pull-up resistor
VDD	14	Positive supply voltage

The HCS370 will normally be in a low power SLEEP mode. When a button input is taken high, the device will wake-up, start the step-up regulator, and go through the button debounce delay of TDB before the button code is latched. In addition, the device will then read the configuration options. Depending on the configuration options and the button code, the device will determine what the data and modulation format will be for the transmission. The transmission will consist of a stream of code words and will be transmitted TPU after the button is pressed for as long as the buttons are held down or until a time-out occurs. The code word format can be either a code hopping format or a seed format.

The time-out time can be selected with the Time-out Select (TSEL) configuration option. This option allows the time-out to be set to 0.8s, 3.2s, 12.8s, or 25.6s. When a time-out occurs, the device will go into SLEEP mode to protect the battery from draining when a button gets stuck. This option must be chosen to meet maximum transmission length regulatory limits which vary by country.

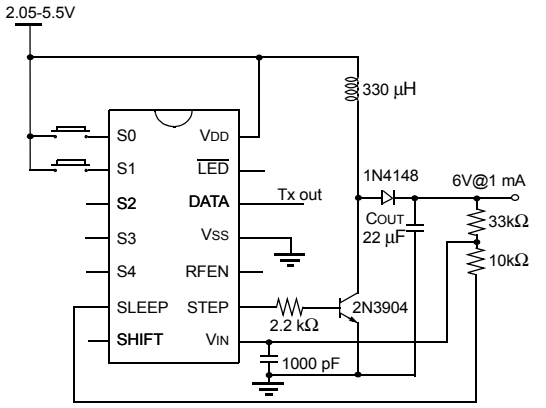
FIGURE 2-1: TYPICAL CIRCUITS

Figure 2-1(A)



Six Button remote with PLL control

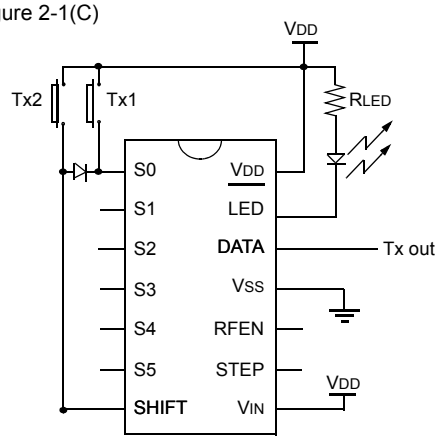
Figure 2-1(B)



Two Button remote with Step-up circuit

Note: Using SLEEP output low instead of grounding the resistor divider reduces battery drain between transmissions

Figure 2-1(C)



DUAL Transmitter remote control

If the device is in the transmit process and detects that a new button is pressed, the current code word will be aborted, a new code word will be transmitted and the time-out counter will RESET. If all the buttons are released, a minimum number of code words will still be completed. The minimum code words can be set to 1, 2, 4, or 8 using the Minimum Code Words (MTX) configuration option. If the time for transmitting the minimum code words is longer than the time-out time, the device will not complete the minimum code words.

The HCS370 has an onboard nonvolatile EEPROM. This EEPROM is used to store user programmable data and the synchronization counter. The data is programmed at the time of production and includes the security related information such as encoder keys, serial numbers, discrimination values, and seed values. All the security related options are read protected.

The initial counter value is also programmed at the time of production. From then on the device maintains the counter itself. The HCS370 has built in redundancy for counter protection and can recover from counter corruption.

The counter will not increment if the previous write was corrupted by low voltage RESET or power failure during TPLL. Instead, the counter will revert back to the previous count and the HCS370 will attempt to correct the bad bits. This will continue on every button press until the voltage increases and the counter is successfully corrected.

FIGURE 2-2: I/O CIRCUITS

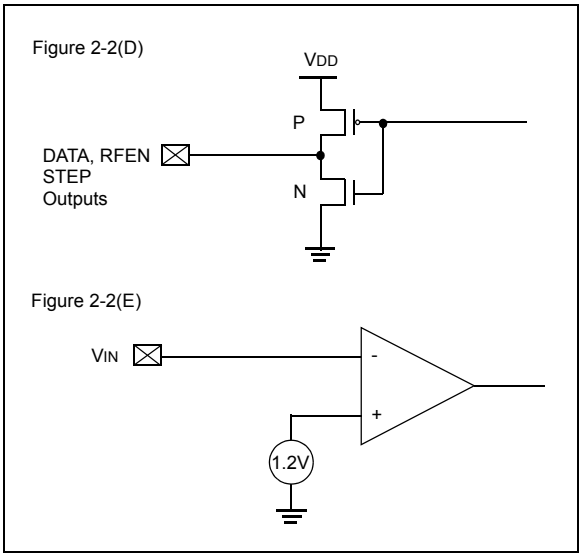


FIGURE 2-3: I/O CIRCUITS (*CONTINUED*)

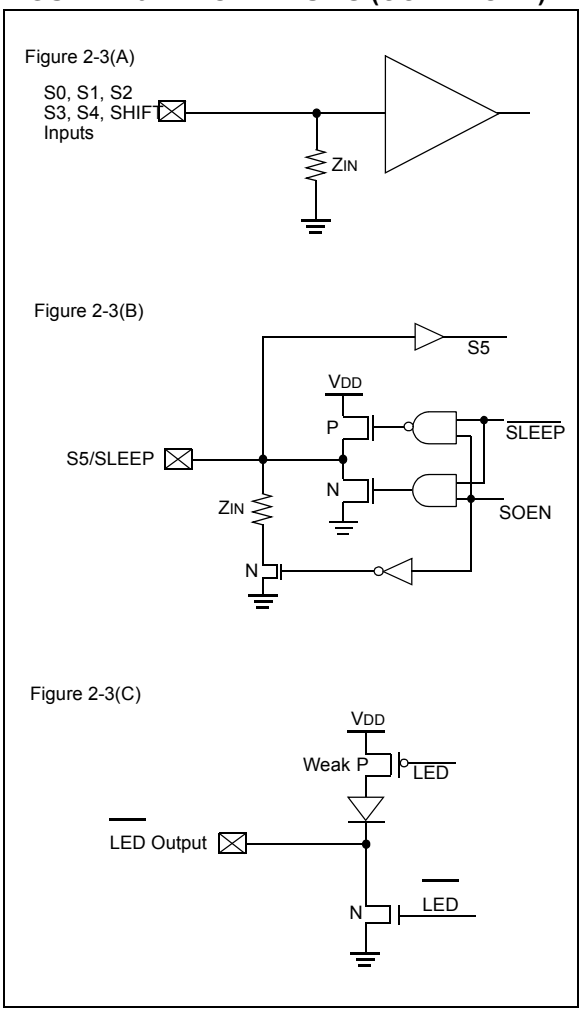
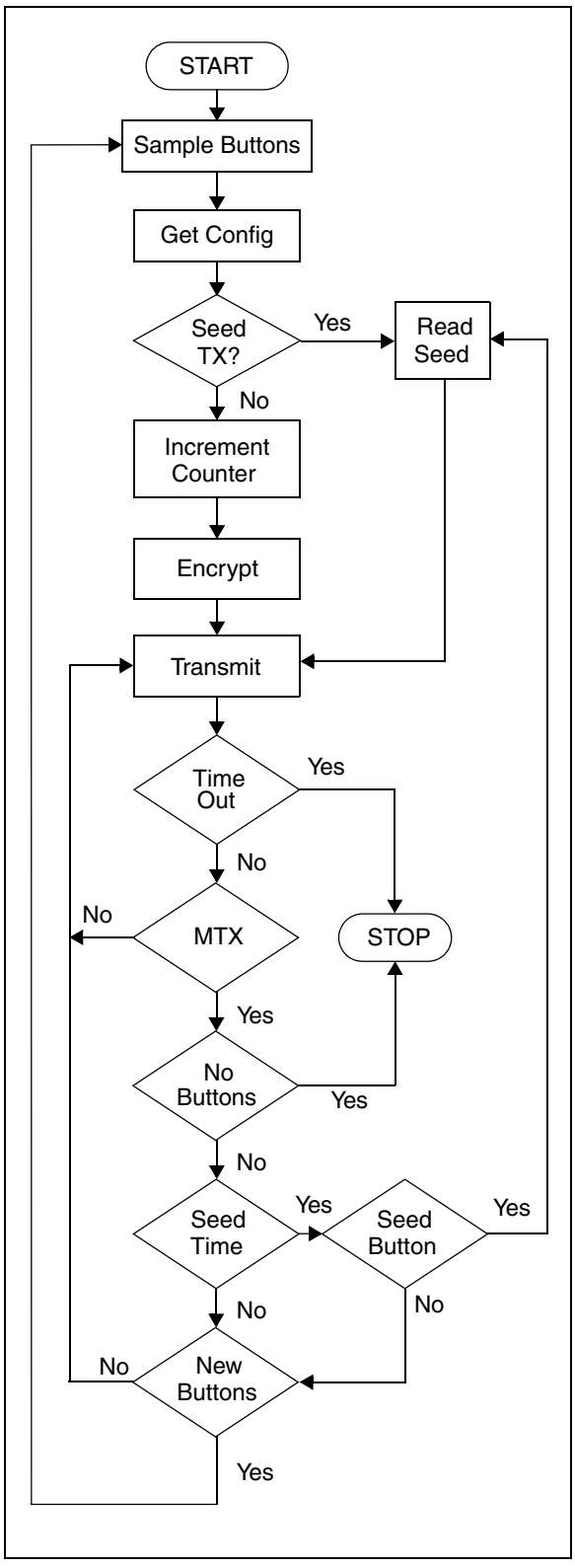


FIGURE 2-4: BASIC FLOW DIAGRAM OF THE DEVICE OPERATION



3.0 EEPROM ORGANIZATION

A summary of the HCS370 EEPROM organization is shown in the three tables below. The address column shows the starting address of the option, and its length or bit position. Options larger than 8 bits are stored with the Most Significant bits at the given address. Enough consecutive 8-bit blocks are reserved for the entire option size. Options such as SEED1, which have a length that is not an exact multiple of 8 bits, is stored right justified in the reserved space. Additional smaller options such as SDBT1 may be stored in the same address as the Most Significant bits.

TABLE 3-1: ENCODER1 OPTIONS (SHIFT = 0)

Symbol	$Address_{16}$:Bits	Description(1)			Reference Section
KEY1	1E: 64 bits	Encoder Key			3.2.2
SEED1	14: 60 bits	Encoder Seed Value			3.3
SYNC1	00: 20 bits 00: 18 bits	Encoder Synchronization Counter (CNTSEL=1) Encoder Synchronization Counter (CNTSEL=0) plus overflow			3.2, 3.2.1
SER1	10: 32 bits	Encoder Serial Number			3.2.2
DISC1	1C: 10 bits	Encoder Discrimination value			3.2, 3.2.1
MSEL1	1C: ---- 32--	Transmission Modulation Format	$Value_2$	Format	4.1
			00	PWM	
			01	Manchester	
			10	VPWM	
			11	PPM	
HSEL1	1C: ---4 ----	Header Select	4 TE = 0	10 TE = 1	4.1
XSER1	1C: --5- ----	Extended Serial Number	28 bits = 0	32 bits = 1	3.2
QUEN1	1C: -6-- ----	Queue counter Enable	Disable = 0	Enable = 1	5.6
STEN1	1C: 7--- ----	START/STOP Pulse Enable	Disable = 0	Enable = 1	4.1
LEDBL1	3F: -6-- ----	Low Voltage LED Blink	Never = 0	Once = 1	5.3
LEDOS1	3F: 7--- ----	LED On Time Select(1)	50 ms = 0	100 ms = 1	5.3
SDLM1	3C: ---- ---0	Limited Seed	Disable = 0	Enable = 1	3.3
SDMD1	3C: ---- --1-	Seed Mode	User = 0	Production = 1	3.3
SDBT1	14: 7654 ----	Seed Button Code			3.3
SDTM1	3C: ---- 32--	Time Before Seed Code Word(1)	$Value_2$	Time (s)	3.3
			00	0.0	
			01	0.8	
			10	1.6	
			11	3.2	
BSEL1	3C: --54 ----	Transmission Baud Rate Select(1)	$Value_2$	TE (µs)	4.1
			00	100	
			01	200	
			10	400	
			11	800	
GSEL1	3C: 76-- ----	Guard Time Select(1)	$Value_2$	Time (ms)	4.1, 5.2
			00	2 TE	
			01	6.4	
			10	51.2	
			11	102.4	

Note 1: All Timing values vary ±10%.

TABLE 3-2: ENCODER2 OPTIONS (SHIFT = 1)

Symbol	$Address_{16}$:Bits	Description(1)			Reference Section
KEY2	34: 64 bits	Encoder Key			3.2.1
SEED2	2A: 60 bits	Encoder Seed Value			3.3
SYNC2	08: 20 bits 08: 18 bits	Encoder Synchronization Counter (CNTSEL=1) Encoder Synchronization Counter (CNTSEL=0) plus overflow			3.2, 3.2.1
SER2	26: 32 bits	Encoder Serial Number			3.2, 3.2.2
DISC2	32: 10 bits	Encoder Discrimination value			3.2, 3.2.1
MSEL2	32: ---- 32--	Transmission Modulation Format	$Value_2$	Format	4.1
			00	PWM	
			01	Manchester	
			10	VPWM	
			11	PPM	
HSEL2	32: ---4 ----	Header Select	4 TE = 0	10 TE = 1	4.1
XSER2	32: --5- ----	Extended Serial Number	28 bits = 0	32 bits = 1	3.2
QUEN2	32: -6-- ----	Queue counter Enable	Disable = 0	Enable = 1	5.6
STEN2	32: 7--- ----	START/STOP Pulse Enable	Disable = 0	Enable = 1	4.1
LEDBL2	3D: -6-- ----	Low Voltage LED Blink	Never = 0	Once = 1	5.3
LEDOS2	3D: 7--- ----	LED On Time Select(1)	50 ms = 0	100 ms = 1	5.3
SDLM2	3E: ---- ---0	Limited Seed	Disable = 0	Enable = 1	3.3
SDMD2	3E: ---- --1-	Seed Mode	User = 0	Production = 1	3.3
SDBT2	2A: 7654 ----	Seed Button Code			3.3
SDTM2	3E: ---- 32--	Time Before Seed Code word(1)	$Value_2$	Time (s)	3.3
			00	0.0	
			01	0.8	
			10	1.6	
			11	3.2	
BSEL2	3E: --54 ----	Transmission Baud Rate Select(1)	$Value_2$	TE (µs)	4.1
			00	100	
			01	200	
			10	400	
			11	800	
GSEL2	3E: 76-- ----	Guard Time Select(1)	$Value_2$	Time (ms)	4.1, 5.2
			00	2 TE	
			01	6.4	
			10	51.2	
			11	102.4	

Note 1: All Timing values vary ±10%.

TABLE 3-3: DEVICE OPTIONS

Symbol	$Address_{16}$:Bits	Description(1)			Reference Section
WAKE	3F: ---- --10	Wake-up(1)	$Value_2$	Value	4.1
			00	No Wake-up	
			01	75 ms 50%	
			10	50 ms 33.3%	
			11	100 ms 16.7%	
CNTSEL	3F: ---- -2--	Counter Select	16 bits = 0	20 bits = 1	3.2.1
VLOWL	3F: ---- 3---	Low Voltage Latch Enable	Disable = 0	Enable = 1	3.2.3.1
VLOWSEL	3F: ---4 ----	Low Voltage Trip Point Select(2)	2.2 V = 0	3.2V = 1	3.2.3.1
PLLSEL	3F: --5- ----	PLL Interface Select	ASK = 0	FSK = 1	5.2
MTX	3D: ---- --10	Minimum Code Words	$Value_2$	Value	2.0
			00	1	
			01	2	
			10	4	
			11	8	
SOEN	3D: ---- 3---	SLEEP Output Enable	Disable = 0	Enable = 1	5.4
WAIT	3D: ---- -2--	Wait for Step-Up Regulator	Disable = 0	Enable = 1	5.2, 5.4
TSEL	3D: --54 ----	Time-out Select(1)	$Value_2$	Time(s)	2.0
			00	0.8	
			01	3.2	
			10	12.8	
			11	25.6	

Note 1: All Timing values vary ±10%.
2: Voltage thresholds are ±150 mV.

3.1 Dual Encoder Operation

The HCS370 contains two transmitter configurations with separate serial numbers, encoder keys, discrimination values, syncronization counters, and seed values. The code word is calculated using one of two possible encoder configurations. Most options for code word and modulation formats can be different from Encoder 1 and Encoder 2. However, LED and RF transmitter options have to be the same. The SHIFT input pin is used to select between the encoder configurations. A low on the SHIFT pin will select Encoder 1 and a high will select Encoder 2.

3.2 Code Word Format

A KEELOQ code word consists of 32 bits of hopping code data, 32 bits of fixed code data, and between 3 to 5 bits of status information. Various code word formats are shown in Figure 3-1 and Figure 3-2.

3.2.1 HOPPING CODE PORTION

The hopping code portion is calculated by encrypting the counter, discrimination value, and function code with the Encoder Key (KEY). The hopping code is calculated when a button press is debounced and remains unchanged until the next button press.

The synchronization counter can be either a 16- or 20-bit value. The Configuration Option Counter Select (CNTSEL) will determine this. The counter select option must be the same for both Encoder 1 and Encoder 2.

If the 16-bit counter is selected, the discrimination value is 10 bits long and there are 2 counter overflow bits (OVR0, OVR1). Set both bits in production and OVR0 will be cleared on the first counter overflow and OVR1 on the second. Clearing OVR0 with OVR1 set will only detect the first overflow. Clearing both OVR bits will effectively give 12 constant bits for discrimination.

If the counter is 20 bits, the discrimination value is 8 bits long and there are no overflow bits. The rest of the 32 bits are made up of the function code also known as the button inputs.

The discrimination value can be programmed with any value to serve as a post decryption check on the decoder end. In a typical system, this will be programmed with the 8 or 10 Least Significant bits of the serial number. This will be stored by the receiver system after a transmitter has been learned. The discrimination bits are part of the information that is to form the encrypted portion of the transmission.

3.2.2 FIXED CODE PORTION

The 32 bits of fixed code consist of 28 bits of the serial number (SER) and a copy of the 4-bit function code. This can be changed to contain the whole 32-bit serial number by setting the Extended Serial Number (XSER) configuration option to a 1. If more than one button is pressed, the function codes are logically OR'ed together. The function code is repeated in the encrypted and unencrypted data of a transmission.

TABLE 3-4: FUNCTION CODES

Button	Function Code$_2$
S0	xx1x$_2$
S1	x1xx$_2$
S2	1xxx$_2$
S3	xxx1$_2$
S4	111x$_2$
S5	11x1$_2$

3.2.3 STATUS INFORMATION

The status bits will always contain the output of the Low Voltage (VLOW) detector and Cyclic Redundancy Check (CRC). If Queue (QUEN) is enabled, button queue information will be included in the code words.

FIGURE 3-1: CODE WORD DATA FORMAT (16-BIT COUNTER)

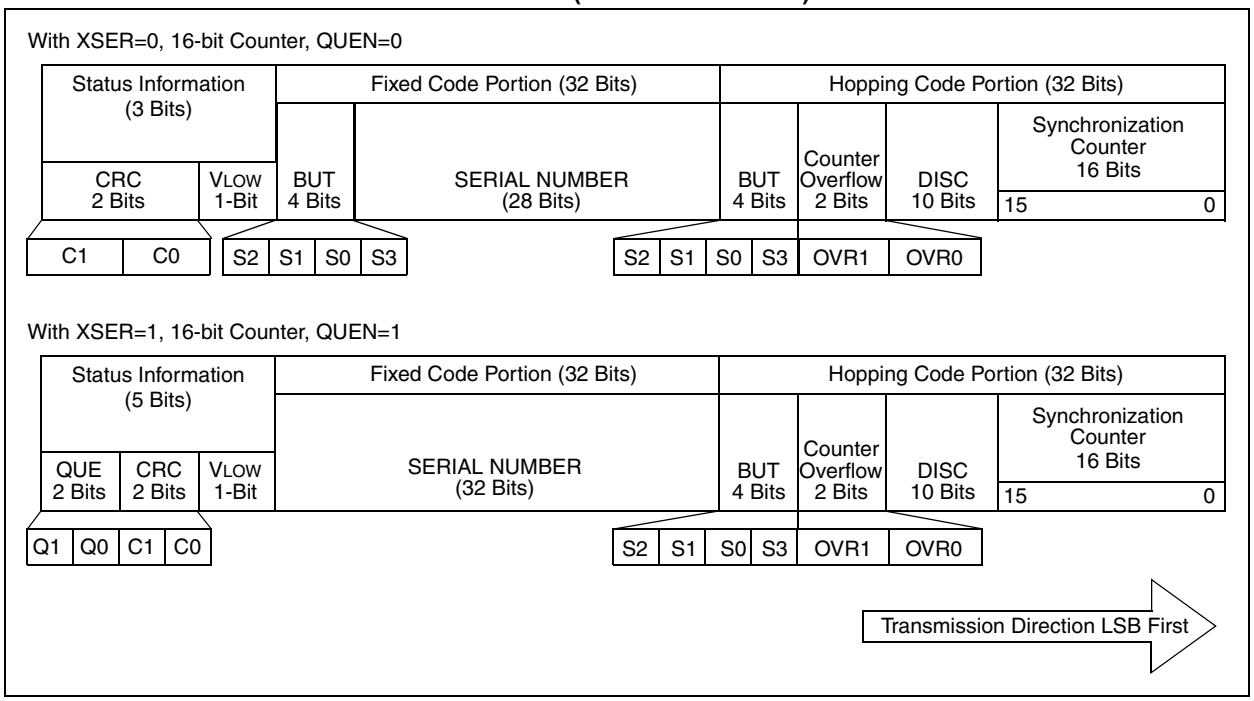
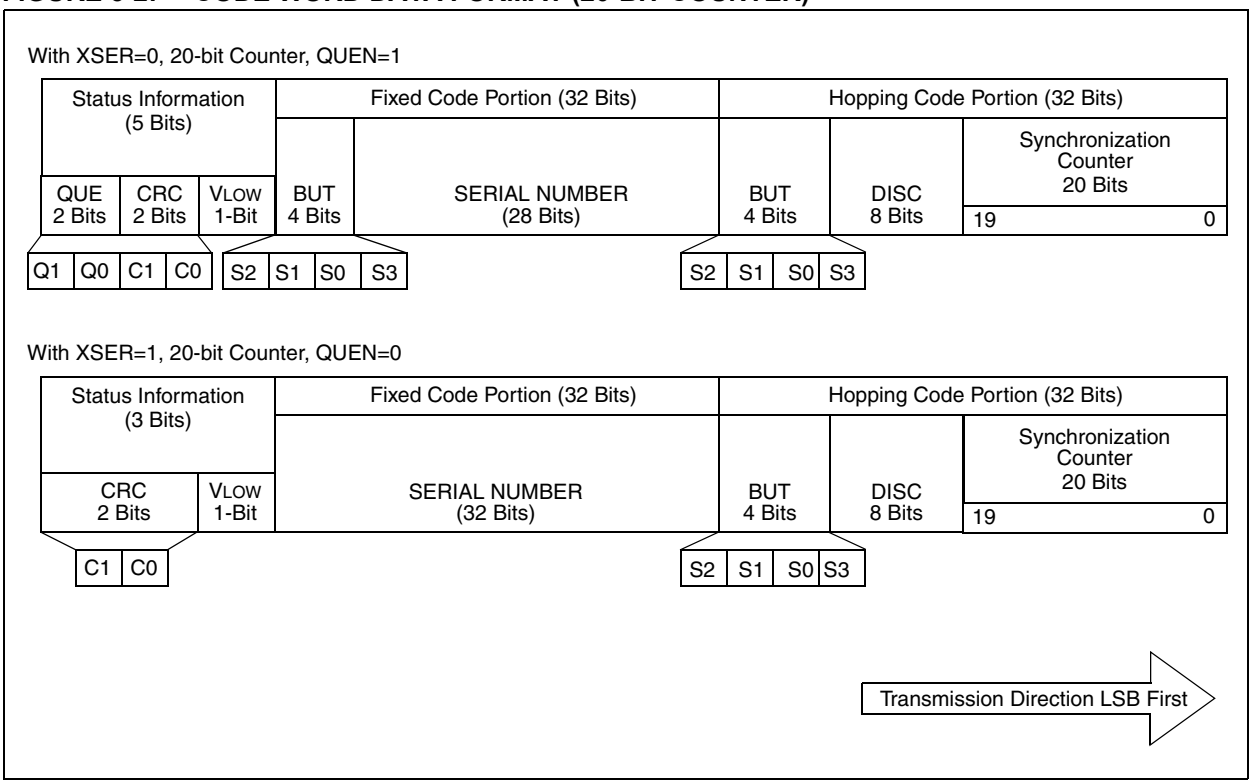


FIGURE 3-2: CODE WORD DATA FORMAT (20-BIT COUNTER)



3.2.3.1 Low Voltage Detector Status (VLOW)

A low battery voltage detector onboard the HCS370 can indicate when the operating voltage drops below a predetermined value. There are two options available depending on the Low Voltage Trip Point Select (VLOWSEL) configuration option. The two options provided are:

- A 2.2V nominal level for 3V operation
- A 3.2V nominal level for 5V operation

The output of the low voltage detector is checked on the first preamble pulse of each code word with the LED momentarily turned off. The VLOW bit is transmitted in each code word so the decoder can give an indication to the user that the transmitter battery is low. Operation of the LED changes as well to further indicate that the battery is low and needs replacing.

The output of the Low Voltage Detector can also be latched once it has dropped below the selected value. The Low Voltage Latch (VLOWL) configuration option enables this option. If this option is enabled, the detector level is raised to 3V or 5V once a low battery voltage has been detected, like a Schmitt Trigger.

This will effectively hold the VLOW bit high until the battery is replaced. If the Low Voltage Latch is enabled, then the low TE after the first preamble pulse can stretch by 4 ms one time as the latch changes state.

3.3 Seed Code Word Data Format

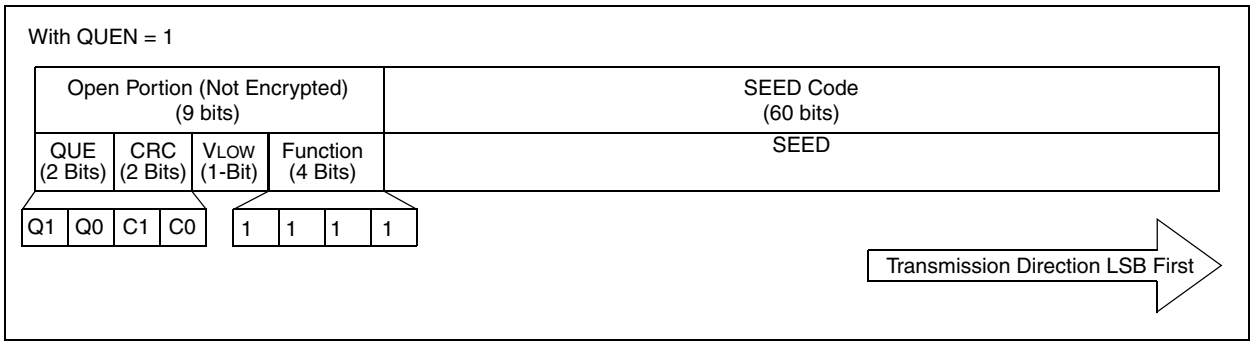
A seed transmission transmits a code word that consists of 60 bits of fixed data that is stored in the EEPROM. This can be used for secure learning of encoders or whenever a fixed code transmission is required. The seed code word is identified by the function bits = 1111_2. The seed code word also contains the status information (VLOW, CRC, and QUEUE). The Seed code word format is shown in Figure 3-3. The function code for seed code words is always 1111_2.

Seed code words for Encoder 1 and Encoder 2 can be configured as follows:

- Enabled with the Seed Button Code (SDBT) configuration option, or disabled if SDBT = 0000_2.
- If the Limited Seed (SDLM) configuration option is set, seed transmissions will be disabled when the synchronization counter is bigger than 127. Seed transmissions remain disabled even if the 16/20-bit counter rolls over to 0.
- The delay before the seed transmission is sent can be set to 0.0s, 0.8s, 1.6s and 3.2s with the Seed Time (SDTM) configuration option. When SDTM is set to a value other than 0.0s, the HCS370 will transmit a code hopping transmission until the selected time expires. After the selected time expires, the seed code words are transmitted. This is useful for the decoder to learn the serial number and the seed from a single button press.
- The button code for transmitting a seed code word can be selected with the Seed Button (SDBT) configuration option. SDBT bits 0 to 3 correspond to button inputs S0 to S3. Set the bits high for the button combination that should trigger a seed transmission (i.e., If SDBT = 1010_2 then, S3+S1 will trigger a seed transmission).
- The seed transmissions before the counter increments past 128 can be modified with the Seed Mode (SDMD) configuration option. Setting this bit for Production mode will cause the selected seed button combination to first transmit a normal hopping code word for the selected Minimum Code words (MTX) and then at least MTX seed code words until all buttons are released. This mode is disabled after the counter reaches 128 even if the 16/20-bit counter rolls over to 0.
- The limit of 127 for SDLM or SDMD can be reduced by using an initial counter value >0.

Note:	The synchronization counter only increments on code hopping transmissions. The counter will not advance on a seed transmission unless Seed Delay or Production mode options are on.

FIGURE 3-3: SEED CODE WORD FORMAT



4.0 TRANSMITTED WORD

4.1 Transmission Modulation Format

The HCS370 transmission is made up of several code words. Each code word contains a preamble, header, and data. A code word is separated from another code word by guard time. The Guard Time Select (GSEL) configuration option can be set to 0 ms, 6.4 ms, 51.2 ms, or 102.4 ms.

All other timing specifications for the modulation formats are based on a basic timing element (T_E). This Timing Element can be set to 100 µs, 200 µs, 400 µs or 800 µs with the Baud Rate Select (BSEL) configuration option. The Header time can be set to $4T_E$ or $10T_E$ with the Header Select (HSEL) configuration option. These options can all be set individually for Encoder 1 and Encoder 2.

There are four different modulation formats available, the Modulation Select (MSEL) Configuration Option is used to select between:

- Pulse Width Modulation (PWM)
- Manchester (MAN)
- Variable Pulse Width Modulation (VPWM)
- Pulse Position Modulation (PPM)

FIGURE 4-1: PULSE WIDTH MODULATION (PWM)

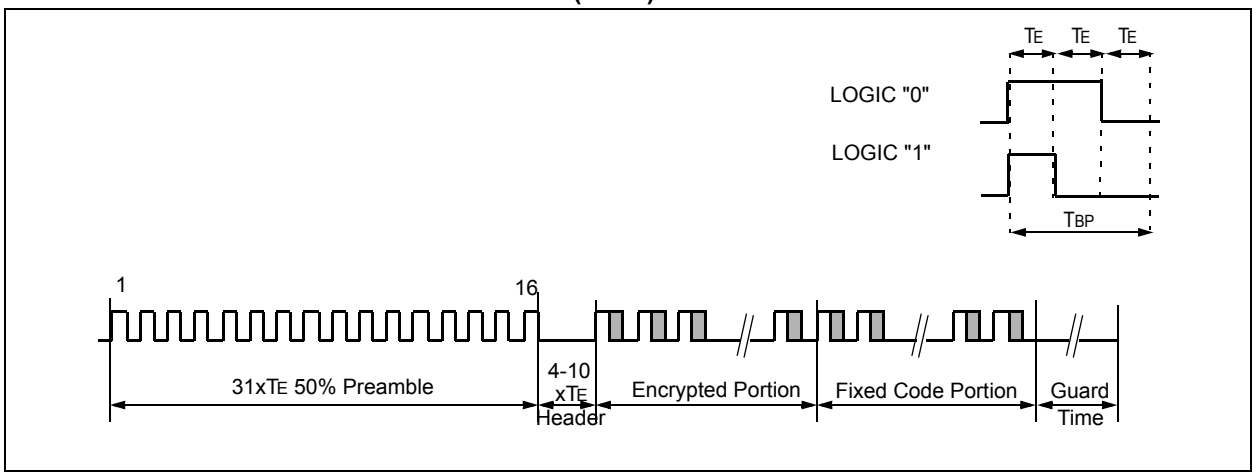


FIGURE 4-2: MANCHESTER (MAN)

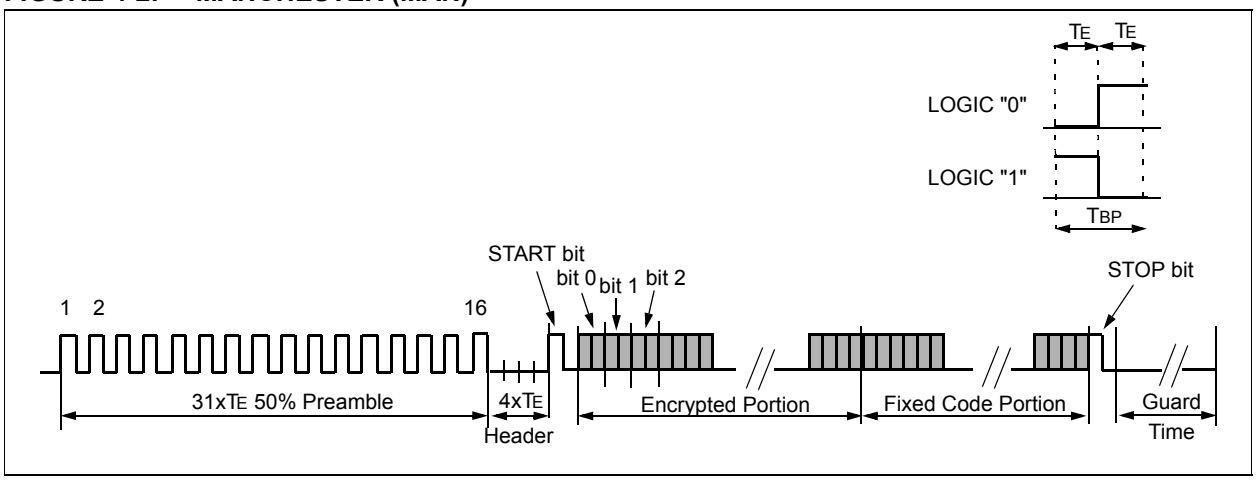


FIGURE 4-3: VARIABLE PULSE WIDTH MODULATION (VPWM)

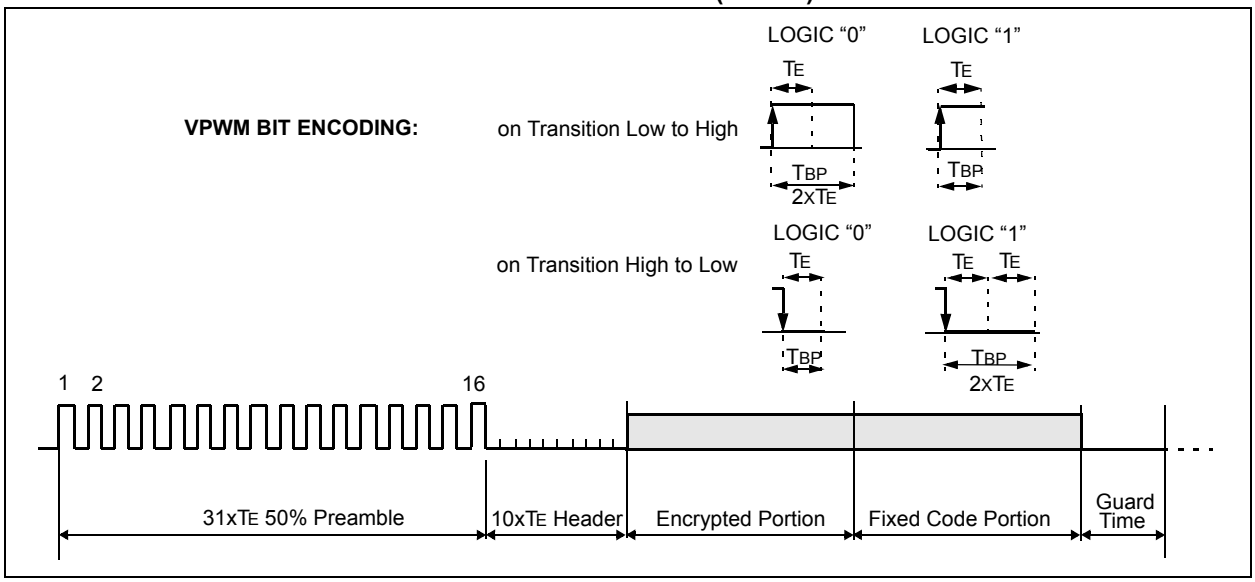
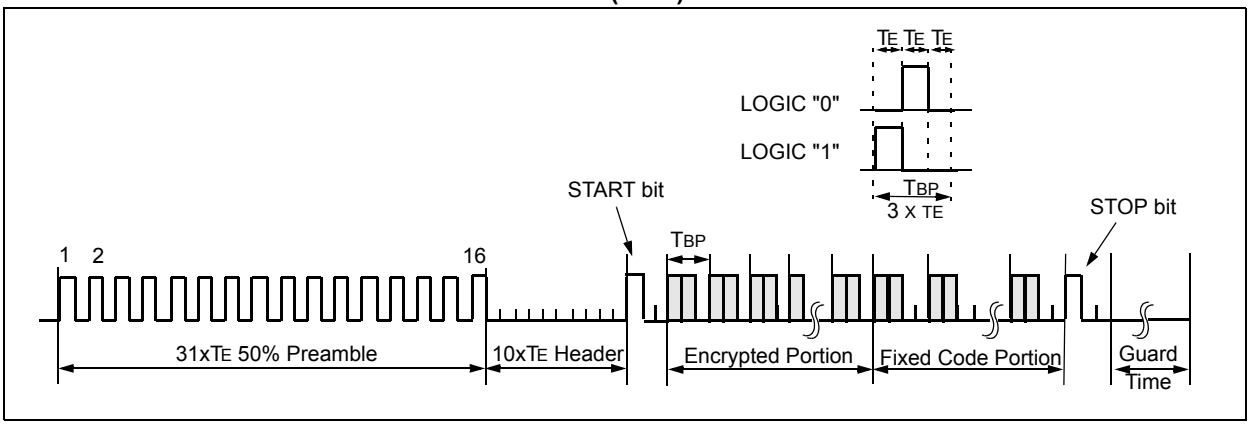


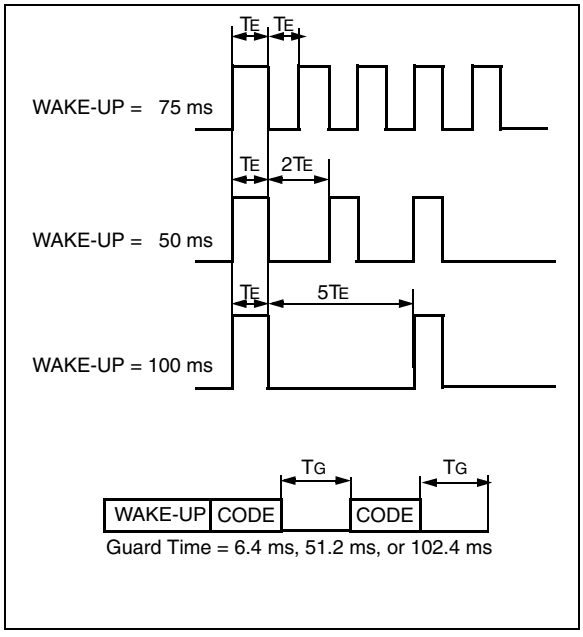
FIGURE 4-4: PULSE POSITION MODULATION (PPM)



In addition to the Modulation Format, Guard Time, and Baud Rate, the following options are also available to change the transmission format:

- If the START/STOP Pulse Enable (STEN) configuration option is enabled, the HCS370 will place a leading and trailing '1' on each code word. This is necessary for modulation formats such as Manchester and PPM to interpret the first and last data bit.
- A wake-up sequence can be transmitted before the transmission starts. The wake-up sequence is configured with the Wake-up (WAKE) configuration option and can be disabled or set to 50 ms, 75 ms, or 100 ms of pulses as indicated in Figure 4-5.
- The WAKE option is the same for both Encoder 1 and Encoder 2.

FIGURE 4-5: WAKE-UP ENABLE



5.0 SPECIAL FEATURES

5.1 Internal RC Oscillator

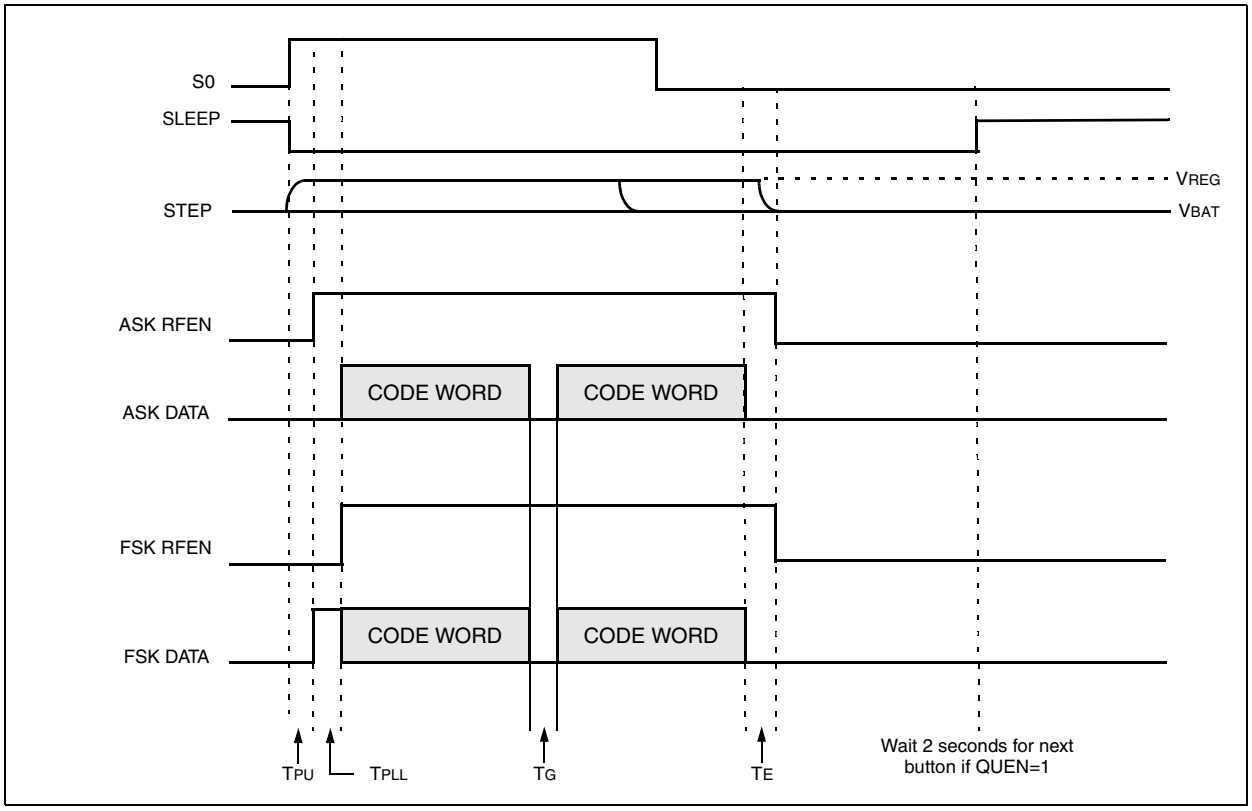
The HCS370 has an onboard RC oscillator that controls all the logic output timing characteristics. The oscillator frequency varies over temperature and voltage variances, but stays within ±10% of the tuned value. All the timing values specified in this document are subject to this oscillator variation.

5.2 RF Enable and PLL Interface

The RFEN pin will be driven high whenever data is transmitted through the DATA pin.

The RFEN and DATA outputs also interface with RF PLL's. The PLL Interface Select (PLLSEL) configuration option selects between ASK and FSK interfaces. Figure 5-1 shows the startup sequence for both ASK and FSK interface options. The RFEN signal will go low at the end of the last code word, including the guard time (T_G). The power-up time (T_{PU}) is the debounce time plus the step-up regulator ramp up delay if the Wait For Step-Up Regulator (WAIT) configuration option is a '1'. The PLL step-up time (T_{PLL}) is also used to update the EEPROM counter.

FIGURE 5-1: ASK/FSK INTERFACE

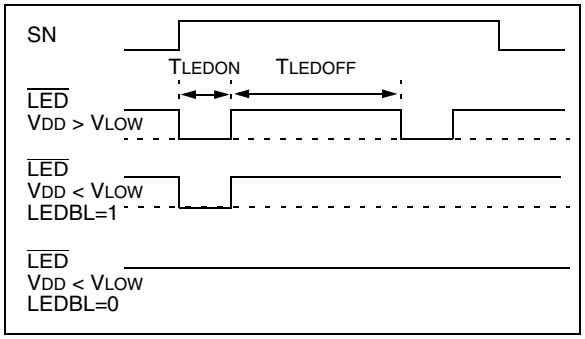


5.3 LED Output

The LED pin will be driven low while the HCS370 is transmitting data. The LED On Time (T_{LEDON}) can be selected between 50 ms and 100 ms with the LED On Time Select (LEDOS) configuration option. The LED Off Time (T_{LEDOFF}) is fixed at 500 ms. When the V_{DD} voltage drops below the selected V_{LOW} trip point, the LED will not blink unless the LED Blink (LEDBL) option is set. If LEDBL is set and V_{DD} is low, then the LED will only flash once. Waveforms of the LED behavior are shown in Figure 5-2.

For circuits with V_{DD} greater than 3 volts, be sure to limit the LED circuit with a series resistor. The LED output can safely sink up to 25 mA but adding an external resistor will conserve battery power. This is an open drain output but it does have a weak pull-up capable of driving a CMOS input.

FIGURE 5-2: $\overline{\text{LED}}$ OPERATION



5.4 Step-Up Voltage Regulator

To create your own step-up regulator circuit, first decide on an output voltage. Second, set the VIN resistor divider to drop it down to 1.2 volts. Keep the sum of the two resistors around 100 kΩ Third, put your maximum load on the output and increase the inductance until COUT charges from 0 volts to your output voltage in about 30 ms from the minimum input voltage. Finally, test over your temperature and input voltage ranges.

The WAIT option will delay RF transmissions until COUT is charged. This permits a trade off in slower button response times to save money on cheaper inductors. This can also optimize performance for good batteries and let response times drift for weak batteries. Also, this option will indicate failure to reach regulation voltage after 250 ms by not transmitting and not flashing the LED. If WAIT is disabled, the step-up regulator still operates and transmissions will always start 30 ms after a button press.

The SLEEP Output Enable (SOEN) option can be enabled if S5 is not used. This reconfigures S5 to be an output high when the HCS370 is sleeping. S5 will be an output low when a button press wakes it up. One way to use this option is to save power on the step-up regulator. The problem is that the VIN resistor divider makes a DC path through the inductor and diode to discharge the battery. By tying the bottom of the divider to SLEEP as shown in Figure 2-1, the path is broken between transmissions.

5.5 Cyclic Redundancy Check (CRC)

The CRC bits are calculated on the 65 previously transmitted bits. These bits contain the 32-bit hopping code, 32-bit fixed code, and VLOW bit. The decoder can use the CRC bits to check the data integrity before processing starts. The CRC can detect all single bit errors and 66% of double bit errors. The CRC is computed as follows:

EQUATION 5-1: CRC Calculation

$$CRC[1]_{n+1} = CRC[0]_n \oplus Di_n$$

and

$$CRC[0]_{n+1} = (CRC[0]_n \oplus Di_n) \oplus CRC[1]_n$$

with

$$CRC[1,\ 0]_0 = 0$$

and Di_n the nth transmission bit 0 <= n <= 64

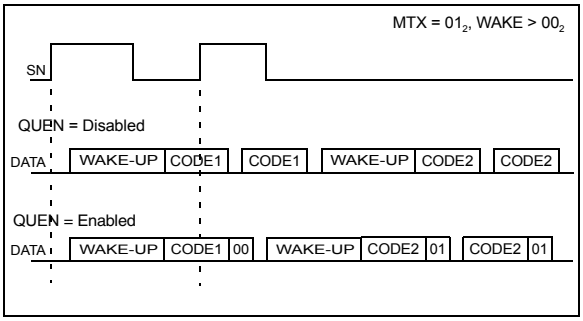
5.6 Button Queue Information (QUEUE)

The queuing or repeated pressing of the same buttons can be handled in two ways on the HCS370. This is controlled with the Queue Counter Enable (QUEN) configuration option. This option can be different for Encoder 1 and Encoder 2.

When the QUEN option is disabled, the device will register up to two sequential button presses. In this case, the device will complete the minimum code words selected with the MTX option before the second code word is calculated and transmitted. The code word will be 67 bits in this case, with no additional queue bits transmitted.

If the QUEN option is enabled, the queue bits are added to the standard code word. The queue bits are a 2-bit counter that does not wrap. The counter value starts at 00_2 and is incremented if a button is pushed within 2 seconds from the start of the previous button press. The current code word is terminated when a button is queued. This allows additional functionality for double or triple button presses.

FIGURE 5-3: CODE WORD COMPLETION WITH QUEN SETTINGS



6.0 PROGRAMMING SPECIFICATIONS

Refer to the “HCS370 Programming Specifications” document (DS41157) in Microchip Literature.

7.0 INTEGRATING THE HCS370 INTO A SYSTEM

Use of the HCS370 in a system requires a compatible decoder. This decoder is typically a microcontroller with compatible firmware. Microchip will provide (via a license agreement) firmware routines that accept transmissions from the HCS370 and decrypt the hopping code portion of the data stream. These routines provide system designers the means to develop their own decoding system.

7.1 Learning a Transmitter to a Receiver

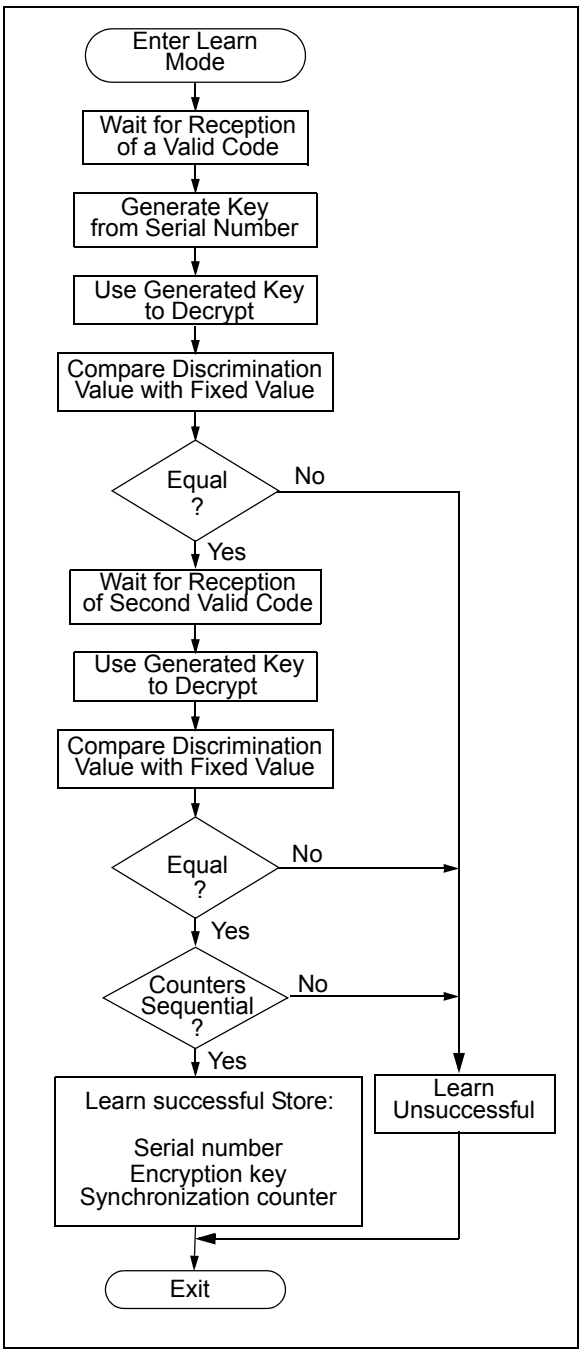
A transmitter must first be 'learned' by a decoder before its use is allowed in the system. Several learning strategies are possible. Figure 7-1 details a typical learn sequence. The decoder must minimally store each learned transmitter's serial number and current synchronization counter value in EEPROM. Additionally, the decoder typically stores each transmitter's unique crypt key. The maximum number of learned transmitters will therefore be relative to the available EEPROM.

A transmitter's serial number is transmitted in the 32-bit fixed code, but the synchronization counter only exists in the code word's encrypted portion. The decoder obtains the counter value by decrypting using the same key used to encrypt the information. The KEELOQ algorithm is a symmetrical block cipher so the encryption and decryption keys are identical and referred to generally as the crypt key. The encoder receives its crypt key during manufacturing. The decoder typically calculates the crypt key by running the encoder serial number or seed through the key generation routine.

Figure 7-1 summarizes a typical learn sequence. The decoder receives and authenticates a first transmission; first button press. Authentication involves generating the appropriate crypt key, decrypting, validating the correct key usage via the discrimination bits, and buffering the counter value. A second transmission is received and authenticated. A final check verifies the counter values were sequential; consecutive button presses. If the learn sequence is successfully completed, the decoder stores the learned transmitter's serial number, current synchronization counter value, and appropriate crypt key. From now on, the crypt key will be retrieved from EEPROM during normal operation instead of recalculating it for each transmission received.

Certain learning strategies have been patented by 3rd parties and care must be taken not to infringe.

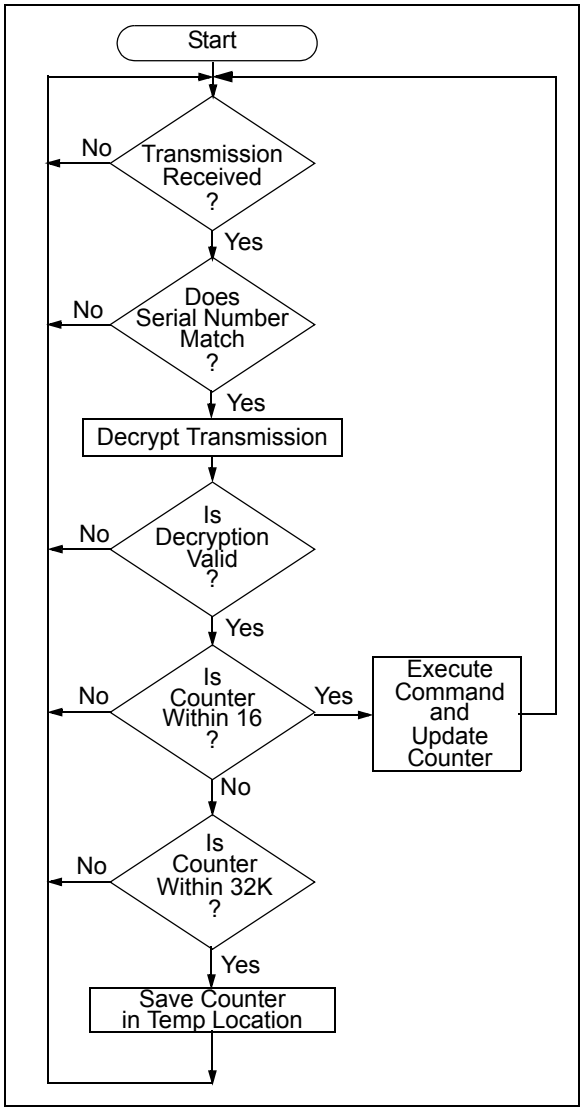
FIGURE 7-1: TYPICAL LEARN SEQUENCE



7.2 Decoder Operation

Figure 7-2 summarizes normal decoder operation. The decoder waits until a transmission is received. The received serial number is compared to the EEPROM table of learned transmitters to first determine if this transmitter's use is allowed in the system. If from a learned transmitter, the transmission is decrypted using the stored crypt key and authenticated via the discrimination bits for appropriate crypt key usage. If the decryption was valid the synchronization value is evaluated.

FIGURE 7-2: TYPICAL DECODER OPERATION



7.3 Synchronization with Decoder (Evaluating the Counter)

The KEELOQ technology patent scope includes a sophisticated synchronization technique that does not require the calculation and storage of future codes. The technique securely blocks invalid transmissions while providing transparent resynchronization to transmitters inadvertently activated away from the receiver.

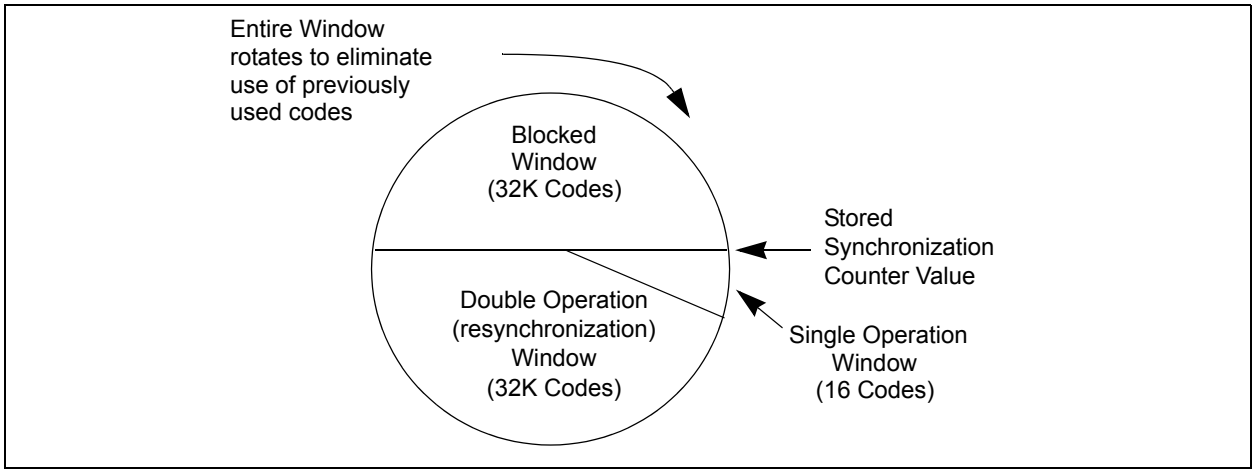
Figure 7-3 shows a 3-partition, rotating synchronization window. The size of each window is optional but the technique is fundamental. Each time a transmission is authenticated, the intended function is executed and the transmission's synchronization counter value is stored in EEPROM. From the currently stored counter value there is an initial "Single Operation" forward window of 16 codes. If the difference between a received synchronization counter and the last stored counter is within 16, the intended function will be executed on the single button press and the new synchronization counter will be stored. Storing the new synchronization counter value effectively rotates the entire synchronization window.

A "Double Operation" (resynchronization) window further exists from the “Single Operation” window up to 32K codes forward of the currently stored counter value. It is referred to as "Double Operation" because a transmission with synchronization counter value in this window will require an additional, sequential counter transmission prior to executing the intended function. Upon receiving the sequential transmission the decoder executes the intended function and stores the synchronization counter value. This resynchronization occurs transparently to the user as it is human nature to press the button a second time if the first was unsuccessful.

The third window is a "Blocked Window" ranging from the double operation window to the currently stored synchronization counter value. Any transmission with synchronization counter value within this window will be ignored. This window excludes previously used, perhaps code grabbed transmissions from accessing the system.

Note:	The synchronization method described in this section is only a typical implementation and because it is usually implemented in firmware, it can be altered to fit the needs of a particular system.

FIGURE 7-3: SYNCHRONIZATION WINDOW



7.4 Security Considerations

The strength of this security is based on keeping a secret inside the transmitter that can be verified by encrypted transmissions to a trained receiver. The transmitter's secret is the manufacturer's key, not the encryption algorithm. If that key is compromised then a smart transceiver can capture any serial number, create a valid code word, and trick all receivers trained with that serial number. The key cannot be read from the EEPROM without costly die probing but it can be calculated by brute force decryption attacks on transmitted code words. The cost for these attacks should exceed what you would want to protect.

To protect the security of other receivers with the same manufacturer's code, you need to use the random seed for secure learn. It is a second secret that is unique for each transmitter. Its transmission on a special button press combination can be disabled if the receiver has another way to find it, or limited to the first 127 transmissions for the receiver to learn it. This way, it is very unlikely to ever be captured. Now if a manufacturer's key is compromised, clone transmitters can be created, but without the unique seed they have to be relearned by the receiver. In the same way if the transmissions are decrypted by brute force on a computer, the random seed hides the manufacturer's key and prevents more than one transmitter from being compromised.

The length of the code word at these baud rates makes brute force attacks that guess the hopping code take years. To make the receiver less susceptible to this attack, make sure that you test all the bits in the decrypted code for the correct value. Do not just test low counter bits for sync and the bit for the button input of interest.

The main benefit of hopping codes is to prevent the retransmission of captured code words. This works very well for code words that the receiver decodes. Its weakness is if a code is captured when the receiver misses it, the code may trick the receiver once if it is used before the next valid transmission. To make the receiver more secure it could increment the counter on questionable code word receptions. To make the transmitter more secure, it could use separate buttons for lock and unlock functions. Another way would be to require two different buttons in sequence to gain access.

There are more ways to make KEELOQ systems more secure, but they all have trade offs. You need to find a balance between security, design effort, and usability, particularly in failure modes. For example, if a button sticks or kids play with it, the counter should not end up in the blocked code window rendering the transmitter useless or requiring retraining.

8.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

8.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either C or assembly)
- One-touch compile or assemble, and download to emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (C or assembly)
  - Mixed C and assembly
  - Machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost-effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increased flexibility and power.

8.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

8.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

8.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

8.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

8.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

8.7 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC® MCUs and dsPIC® DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB C Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

8.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

8.9 MPLAB ICD 3 In-Circuit Debugger System

MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost effective high-speed hardware debugger/programmer for Microchip Flash Digital Signal Controller (DSC) and microcontroller (MCU) devices. It debugs and programs PIC® Flash microcontrollers and dsPIC® DSCs with the powerful, yet easy-to-use graphical user interface of MPLAB Integrated Development Environment (IDE).

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

8.10 PICkit 3 In-Circuit Debugger/ Programmer and PICkit 3 Debug Express

The MPLAB PICkit 3 allows debugging and programming of PIC® and dsPIC® Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB Integrated Development Environment (IDE). The MPLAB PICkit 3 is connected to the design engineer's PC using a full speed USB interface and can be connected to the target via an Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the reset line to implement in-circuit debugging and In-Circuit Serial Programming™.

The PICkit 3 Debug Express include the PICkit 3, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

8.11 PICkit 2 Development Programmer/Debugger and PICkit 2 Debug Express

The PICkit™ 2 Development Programmer/Debugger is a low-cost development tool with an easy to use interface for programming and debugging Microchip's Flash families of microcontrollers. The full featured Windows® programming interface supports baseline (PIC10F, PIC12F5xx, PIC16F5xx), midrange (PIC12F6xx, PIC16F), PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 families of 8-bit, 16-bit, and 32-bit microcontrollers, and many Microchip Serial EEPROM products. With Microchip's powerful MPLAB Integrated Development Environment (IDE) the PICkit™ 2 enables in-circuit debugging on most PIC® microcontrollers. In-Circuit-Debugging runs, halts and single steps the program while the PIC microcontroller is embedded in the application. When halted at a breakpoint, the file registers can be examined and modified.

The PICkit 2 Debug Express include the PICkit 2, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

8.12 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an MMC card for file storage and data applications.

8.13 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

9.0 ELECTRICAL CHARACTERISTICS

9.1 Maximum Ratings*

Ambient temperature under bias ..... -40°C to +125°C

Storage temperature ..... -65°C to +150°C

Voltage on V_{DD} w/respect to V_{SS} ..... -0.3 to +7.5V

Voltage on $\overline{LED}$ w/respect to V_{SS} ..... -0.3 to +11V

Voltage on all other pins w/respect to V_{SS} ..... -0.3V to V_{DD} + 0.3V

Total power dissipation **(Note 1)** ..... 500 mW

Maximum current out of V_{SS} pin ..... 100 mA

Maximum current into V_{DD} pin ..... 100 mA

Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ± 20 mA

Output clamp current, I_{OK} (Vo < 0 or Vo >V_{DD}) ..... ± 20 mA

Maximum output current sunk by any Output pin ..... 25 mA

Maximum output current sourced by any Output pin ..... 25 mA

***Notice:** Stresses above those listed under “Maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

Note 1: Power dissipation is calculated as follows: Pdis=V_{DD} x {I_{DD} - Â I_{OH}} + Â {(V_{DD}-V_{OH}) x I_{OH}} + Â(V_{OI} x I_{OL}).