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

HCS300

KEELOQ® Code Hopping Encoder

FEATURES

Security

- Programmable 28-bit serial number
- Programmable 64-bit encryption key
- Each transmission is unique
- 66-bit transmission code length
- 32-bit hopping code
- 28-bit serial number, 4-bit button code, 2-bit status
- Crypt keys are read protected

Operating

- 2.0V - 6.3V operation
- Four button inputs
- No additional circuitry required
- 15 functions available
- Selectable baud rate
- Automatic code word completion
- Low battery signal transmitted to receiver
- Non-volatile synchronization data

Other

- Easy-to-use programming interface
- On-chip EEPROM
- On-chip oscillator and timing components
- Button inputs have internal pull-down resistors
- Current limiting on $\overline{\text{LED}}$ output
- Low external component cost

Typical Applications

The HCS300 is ideal for Remote Keyless Entry (RKE) applications. These applications include:

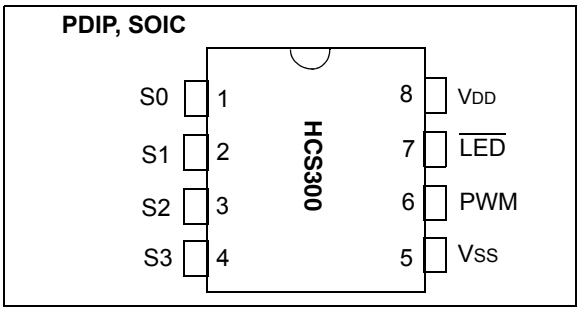
- Automotive RKE systems
- Automotive alarm systems
- Automotive immobilizers
- Gate and garage door openers
- Identity tokens
- Burglar alarm systems

DESCRIPTION

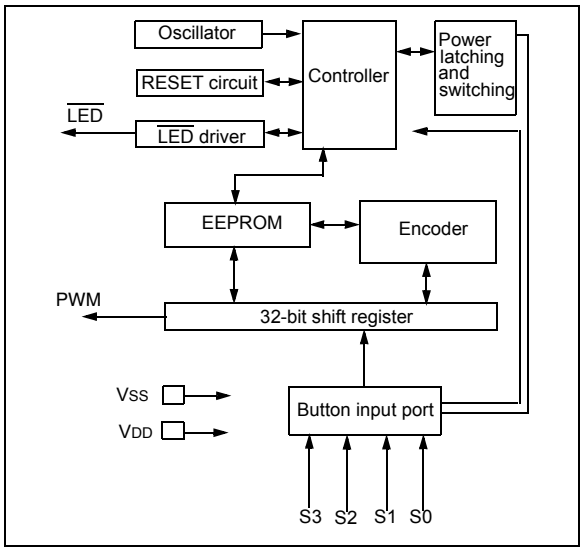
The HCS300 from Microchip Technology Inc. is a code hopping encoder designed for secure Remote Keyless Entry (RKE) systems. The HCS300 utilizes the KEELOQ® code hopping technology, incorporating high security, a small package outline and low cost. The HCS300 is a perfect solution for unidirectional remote keyless entry systems and access control systems.

PACKAGE TYPES

PDIP, SOIC



HCS300 BLOCK DIAGRAM



The HCS300 combines a 32-bit hopping code, generated by a nonlinear encryption algorithm, with a 28-bit serial number and 6 information bits to create a 66-bit code word. The code word length eliminates the threat of code scanning and the code hopping mechanism makes each transmission unique, thus rendering code capture and resend schemes useless.

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. The HCS300 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 3 (TB003).

- **RKE** - Remote Keyless Entry
- **Button Status** - Indicates what button input(s) activated the transmission. Encompasses the 4 button status bits S3, S2, S1 and S0 (Figure 4-2).
- **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 4-1).
- **Transmission** - A data stream consisting of repeating code words (Figure 9-1).
- **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.
- **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, common to all components of all systems by the same manufacturer, to decrypt the received code word's encrypted portion.
  - **Normal Learning**
    The receiver uses information transmitted during normal operation to derive the crypt key and decrypt the received code word's encrypted portion.
  - **Secure Learn**
    The transmitter is activated through a special button combination to transmit a stored 60-bit seed value used to generate the transmitter's crypt key. The receiver uses this seed value to derive the same crypt key and decrypt the received code word's encrypted portion.
- **Manufacturer's code** – A unique and secret 64-bit number used to generate unique encoder 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 HCS300 code hopping encoder is designed specifically for keyless entry systems; primarily vehicles and home garage door openers. The encoder portion of a keyless entry system is integrated into a transmitter, carried by the user and operated to gain access to a vehicle or restricted area. The HCS300 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 HCS300 on the other hand, employs the KEELOQ code hopping technology coupled with a transmission length of 66 bits to virtually eliminate the use of code 'grabbing' or code 'scanning'. The high security level of the HCS300 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 even if the transmission information (before coding) differs by only one bit from that of the previous transmission, the next

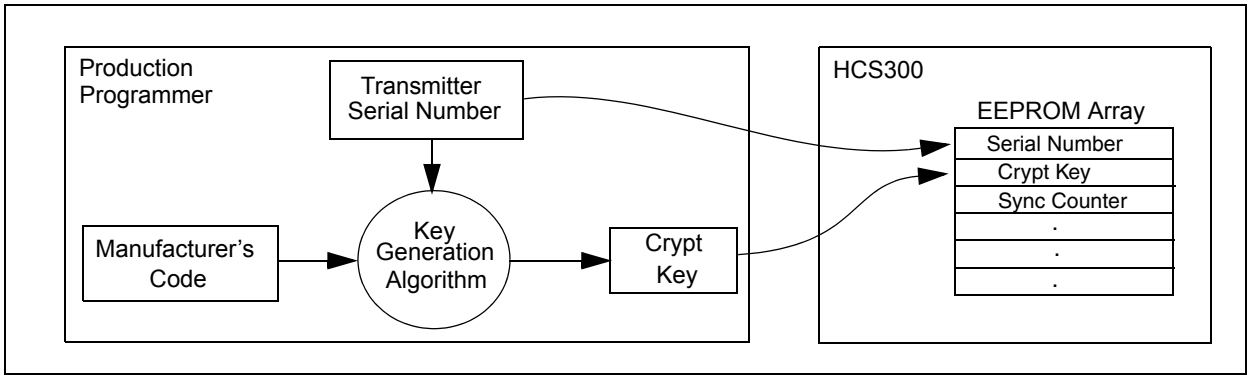
coded transmission will be completely different. Statistically, if only one bit in the 32-bit string of information changes, greater than 50 percent of the coded transmission bits will change.

As indicated in the block diagram on page one, the HCS300 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 28-bit serial number, typically unique for every encoder
- A crypt key
- An initial 16-bit synchronization value
- A 16-bit configuration 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 16-bit synchronization counter is the basis behind the transmitted code word changing for each transmission; it increments each time a button is pressed. Due to the code hopping algorithm's complexity, each increment of the synchronization value results in greater than 50% of the bits changing in the transmitted code word.

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, its value appearing externally to 'randomly hop around', hence it 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 4.0.

A receiver may use any type of controller as a decoder, but it is typically a microcontroller with compatible firmware that allows the decoder to operate in conjunction with an HCS300 based transmitter. Section 7.0 provides detail on integrating the HCS300 into a system.

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 and storing the serial number, synchronization counter value and 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 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.

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

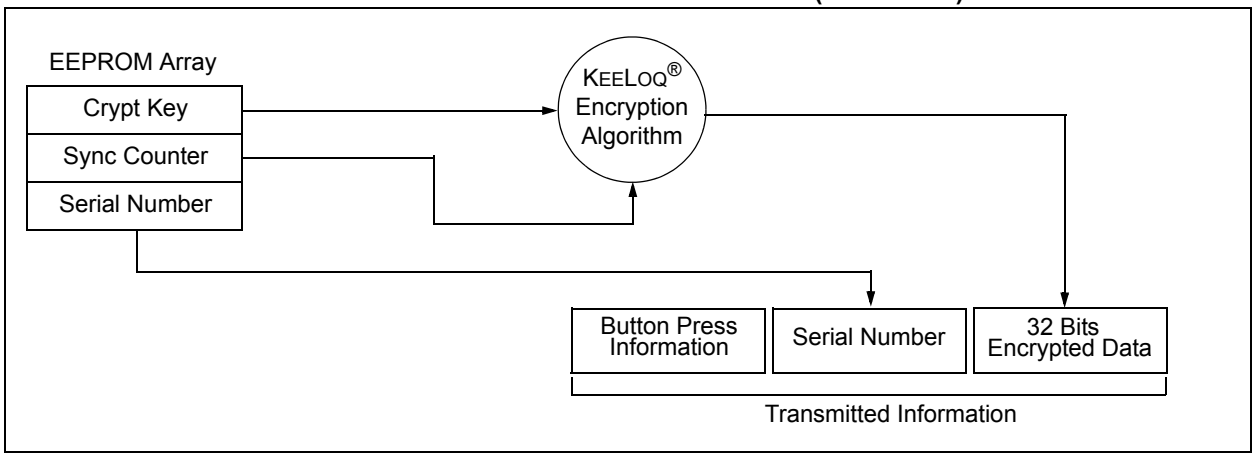
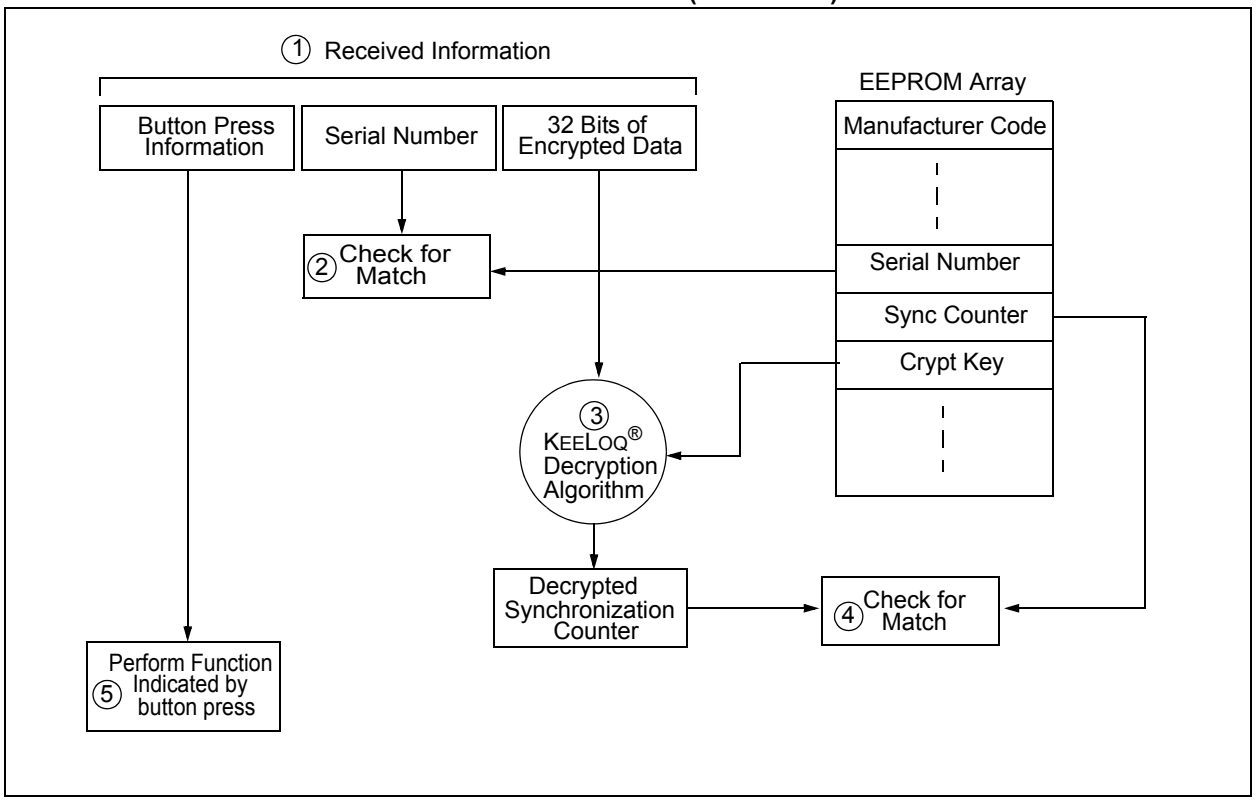


FIGURE 1-3: BASIC OPERATION OF RECEIVER (DECODER)

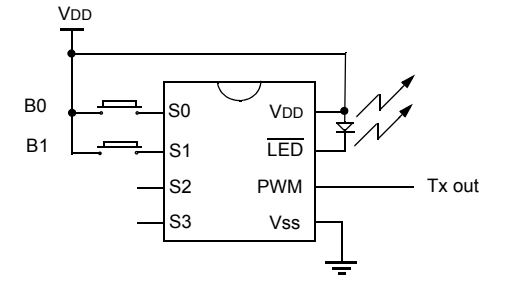


NOTE: Circled numbers indicate the order of execution.

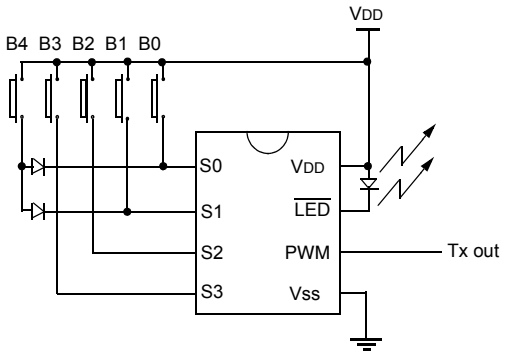
2.0 ENCODER OPERATION

As shown in the typical application circuits (Figure 2-1), the HCS300 is a simple device to use. It requires only the addition of buttons and RF circuitry for use as the transmitter in your security application. A description of each pin is given in Table 2-1.

FIGURE 2-1: TYPICAL CIRCUITS



2 button remote control



5 button remote control **(Note)**

Note: Up to 15 functions can be implemented by pressing more than one button simultaneously or by using a suitable diode array.

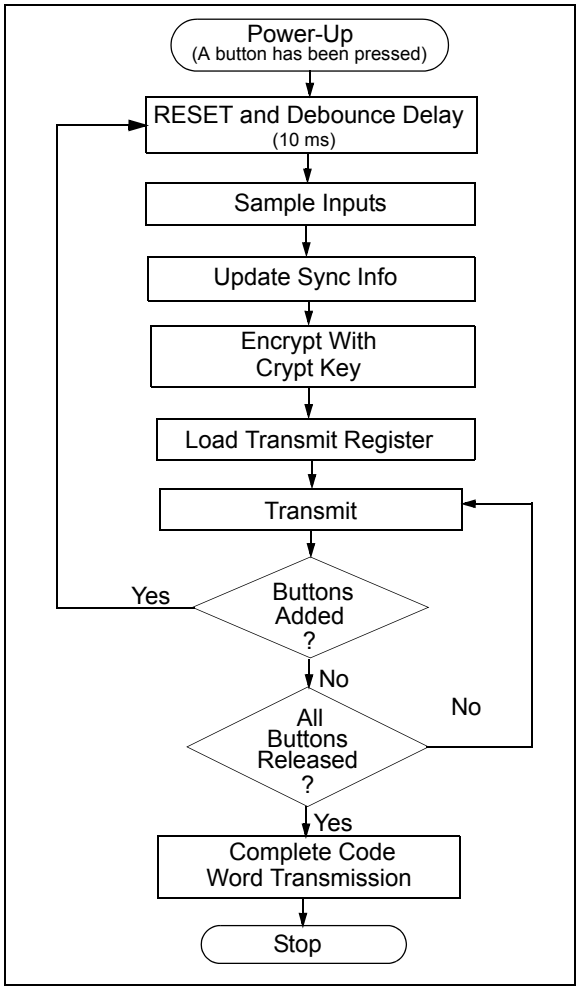
TABLE 2-1: PIN DESCRIPTIONS

Name	Pin Number	Description
S0	1	Switch input 0
S1	2	Switch input 1
S2	3	Switch input 2 / Clock pin when in Programming mode
S3	4	Switch input 3
VSS	5	Ground reference
PWM	6	Pulse Width Modulation (PWM) output pin / Data pin for Programming mode
$\overline{LED}$	7	Cathode connection for LED
VDD	8	Positive supply voltage

The HCS300 will wake-up upon detecting a button press and delay approximately 10 ms for button debounce (Figure 2-2). The synchronization counter, discrimination value and button information will be encrypted to form the hopping code. The hopping code portion will change every transmission, even if the same button is pushed again. A code word that has been transmitted will not repeat for more than 64K transmissions. This provides more than 18 years of use before a code is repeated; based on 10 operations per day. Overflow information sent from the encoder can be used to extend the number of unique transmissions to more than 192K.

If in the transmit process it is detected that a new button(s) has been pressed, a RESET will immediately occur and the current code word will not be completed. Please note that buttons removed will not have any effect on the code word unless no buttons remain pressed; in which case the code word will be completed and the power-down will occur.

FIGURE 2-2: ENCODER OPERATION



3.0 EEPROM MEMORY ORGANIZATION

The HCS300 contains 192 bits (12 x 16-bit words) of EEPROM memory (Table 3-1). This EEPROM array is used to store the encryption key information, synchronization value, etc. Further descriptions of the memory array is given in the following sections.

TABLE 3-1: EEPROM MEMORY MAP

WORD ADDRESS	MNEMONIC	DESCRIPTION
0	KEY_0	64-bit encryption key (word 0) LSb's
1	KEY_1	64-bit encryption key (word 1)
2	KEY_2	64-bit encryption key (word 2)
3	KEY_3	64-bit encryption key (word 3) MSb's
4	SYNC	16-bit synchronization value
5	RESERVED	Set to 0000H
6	SER_0	Device Serial Number (word 0) LSb's
7	SER_1 **(Note)**	Device Serial Number (word 1) MSb's
8	SEED_0	Seed Value (word 0)
9	SEED_1	Seed Value (word 1)
10	RESERVED	Set to 0000H
11	CONFIG	Config Word

Note: The MSB of the serial number contains a bit used to select the Auto-shutoff timer.

3.1 KEY_0 - KEY_3 (64-Bit Crypt Key)

The 64-bit crypt key is used to create the encrypted message transmitted to the receiver. This key is calculated and programmed during production using a key generation algorithm. The key generation algorithm may be different from the KEELOQ algorithm. Inputs to the key generation algorithm are typically the transmitter's serial number and the 64-bit manufacturer's code. While the key generation algorithm supplied from Microchip is the typical method used, a user may elect to create their own method of key generation. This may be done providing that the decoder is programmed with the same means of creating the key for decryption purposes.

3.2 SYNC (Synchronization Counter)

This is the 16-bit synchronization value that is used to create the hopping code for transmission. This value will increment after every transmission.

3.3 Reserved

Must be initialized to 0000H.

3.4 SER_0, SER_1 (Encoder Serial Number)

SER_0 and SER_1 are the lower and upper words of the device serial number, respectively. Although there are 32 bits allocated for the serial number, only the lower order 28 bits are transmitted. The serial number is meant to be unique for every transmitter.

3.5 SEED_0, SEED_1 (Seed Word)

The 2-word (32-bit) seed code will be transmitted when all three buttons are pressed at the same time (see Figure 4-2). This allows the system designer to implement the secure learn feature or use this fixed code word as part of a different key generation/tracking process.

3.5.1 AUTO-SHUTOFF TIMER ENABLE

The Most Significant bit of the serial number (Bit 31) is used to turn the Auto-shutoff timer on or off. This timer prevents the transmitter from draining the battery should a button get stuck in the on position for a long period of time. The time period is approximately 25 seconds, after which the device will go to the Time-out mode. When in the Time-out mode, the device will stop transmitting, although since some circuits within the device are still active, the current draw within the Shutoff mode will be higher than Standby mode. If the Most Significant bit in the serial number is a one, then the Auto-shutoff timer is enabled, and a zero in the Most Significant bit will disable the timer. The length of the timer is not selectable.

3.6 CONFIG (Configuration Word)

The Configuration Word is a 16-bit word stored in EEPROM array that is used by the device to store information used during the encryption process, as well as the status of option configurations. The following sections further explain these bits.

TABLE 3-2: CONFIGURATION WORD

Bit Number	Bit Description
0	Discrimination Bit 0
1	Discrimination Bit 1
2	Discrimination Bit 2
3	Discrimination Bit 3
4	Discrimination Bit 4
5	Discrimination Bit 5
6	Discrimination Bit 6
7	Discrimination Bit 7
8	Discrimination Bit 8
9	Discrimination Bit 9
10	Overflow Bit 0 (OVR0)
11	Overflow Bit 1 (OVR1)
12	Low Voltage Trip Point Select ($V_{LOW\ SEL}$)
13	Baud rate Select Bit 0 (BSL0)
14	Baud rate Select Bit 1 (BSL1)
15	Reserved, set to 0

3.6.1 DISCRIMINATION VALUE (DISC0 TO DISC9)

The discrimination value aids the post-decryption check on the decoder end. It may be any value, but in a typical system it will be programmed as the 12 Least Significant bits of the serial number. Values other than this must be separately stored by the receiver when a transmitter is learned. The discrimination bits are part of the information that form the encrypted portion of the transmission (Figure 4-2). After the receiver has decrypted a transmission, the discrimination bits are checked against the receiver's stored value to verify that the decryption process was valid. If the discrimination value was programmed as the 12 LSb's of the serial number then it may merely be compared to the respective bits of the received serial number; saving EEPROM space.

3.6.2 OVERFLOW BITS (OVR0, OVR1)

The overflow bits are used to extend the number of possible synchronization values. The synchronization counter is 16 bits in length, yielding 65,536 values before the cycle repeats. Under typical use of 10 operations a day, this will provide nearly 18 years of use before a repeated value will be used. Should the system designer conclude that is not adequate, then the overflow bits can be utilized to extend the number of unique values. This can be done by programming OVR0 and OVR1 to 1s at the time of production. The encoder will automatically clear OVR0 the first time that the synchronization value wraps from 0xFFFF to 0x0000 and clear OVR1 the second time the counter wraps. Once cleared, OVR0 and OVR1 cannot be set again, thereby creating a permanent record of the counter overflow. This prevents fast cycling of 64K counter. If the decoder system is programmed to track the overflow bits, then the effective number of unique synchronization values can be extended to 196,608.

3.6.3 BAUD RATE SELECT BITS (BSL0, BSL1)

BSL0 and BSL1 select the speed of transmission and the code word blanking. Table 3-3 shows how the bits are used to select the different baud rates and Section 5.7 provides detailed explanation in code word blanking.

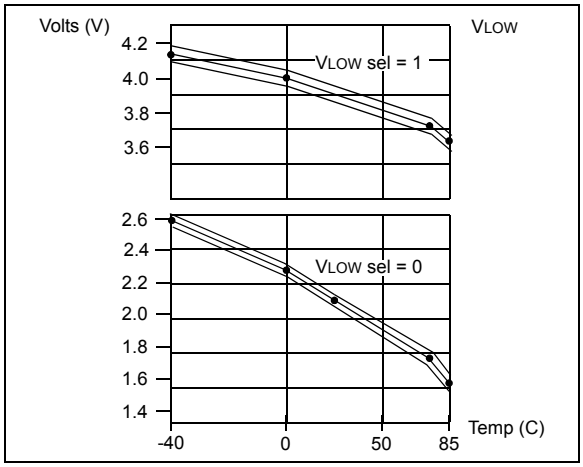
TABLE 3-3: BAUD RATE SELECT

BSL1	BSL0	Basic Pulse Element	Code Words Transmitted
0	0	400 µs	All
0	1	200 µs	1 out of 2
1	0	100 µs	1 out of 2
1	1	100 µs	1 out of 4

3.6.4 LOW VOLTAGE TRIP POINT SELECT ($V_{LOW\ SEL}$)

The low voltage trip point select bit is used to tell the HCS300 what V_{DD} level is being used. This information will be used by the device to determine when to send the voltage low signal to the receiver. When this bit is set to a one, the V_{DD} level is assumed to be operating from a 5V or 6V V_{DD} level. If the bit is set low, then the V_{DD} level is assumed to be 3.0 volts.

FIGURE 3-1: V_{LOW} CHARACTERISTICS



4.0 TRANSMITTED WORD

4.1 Code Word Format

The HCS300 code word is made up of several parts (Figure 4-1). Each code word contains a 50% duty cycle preamble, a header, 32 bits of encrypted data and 34 bits of fixed data followed by a guard period before another code word can begin. Refer to Table 9-4 for code word timing.

4.2 Code Word Organization

The HCS300 transmits a 66-bit code word when a button is pressed. The 66-bit word is constructed from a Fixed Code portion and an Encrypted Code portion (Figure 4-2).

The 32 bits of **Encrypted Data** are generated from 4 button bits, 12 discrimination bits and the 16-bit sync value. The encrypted portion alone provides up to four billion changing code combinations.

The 34 bits of **Fixed Code Data** are made up of 2 status bits, 4 button bits and the 28-bit serial number. The fixed and encrypted sections combined increase the number of code combinations to 7.38 x 10^{19}.

FIGURE 4-1: CODE WORD FORMAT

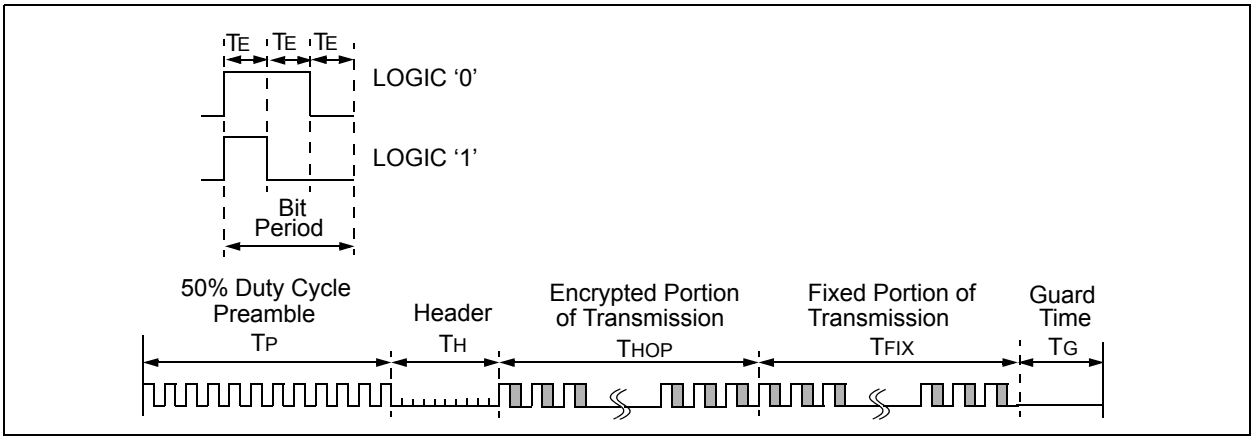
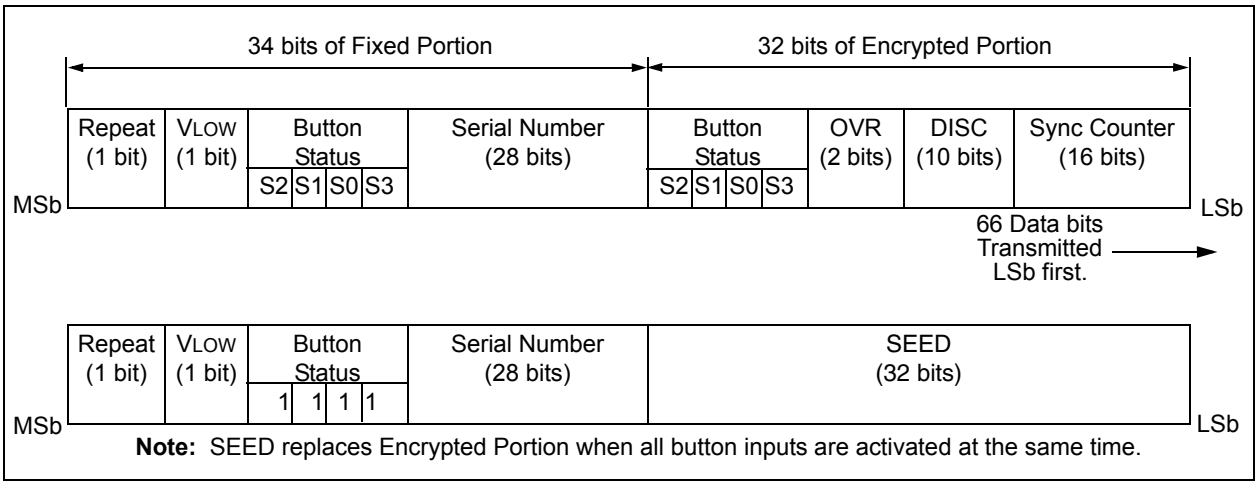


FIGURE 4-2: CODE WORD ORGANIZATION



Note: SEED replaces Encrypted Portion when all button inputs are activated at the same time.

4.3 Synchronous Transmission Mode

Synchronous Transmission mode can be used to clock the code word out using an external clock.

To enter Synchronous Transmission mode, the Programming mode start-up sequence must be executed as shown in Figure 4-3. If either S1 or S0 is set on the falling edge of S2 (or S3), the device enters Synchronous Transmission mode. In this mode, it functions as a normal transmitter, with the exception that the timing of the PWM data string is controlled externally and 16 extra bits are transmitted at the end with the code word.

The button code will be the S0, S1 value at the falling edge of S2 or S3. The timing of the PWM data string is controlled by supplying a clock on S2 or S3 and should not exceed 20 kHz. The code word is the same as in PWM mode with 16 reserved bits at the end of the word. The reserved bits can be ignored. When in Synchronous Transmission mode S2 or S3 should not be toggled until all internal processing has been completed as shown in Figure 4-4.

FIGURE 4-3: SYNCHRONOUS TRANSMISSION MODE

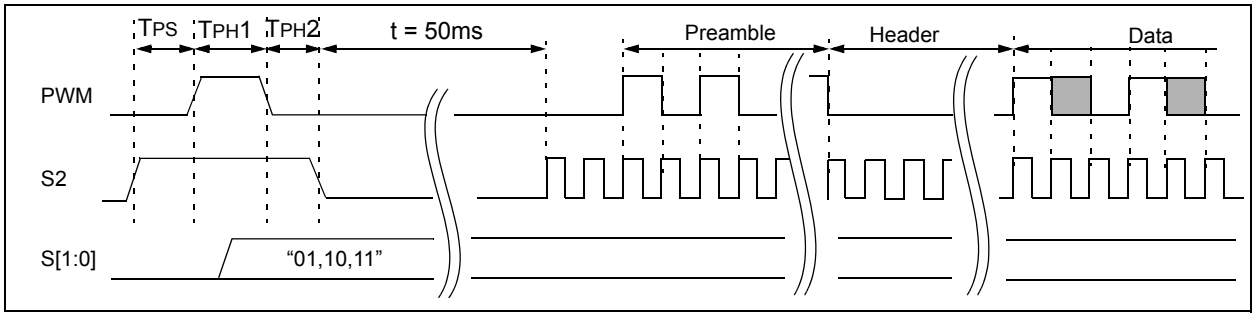
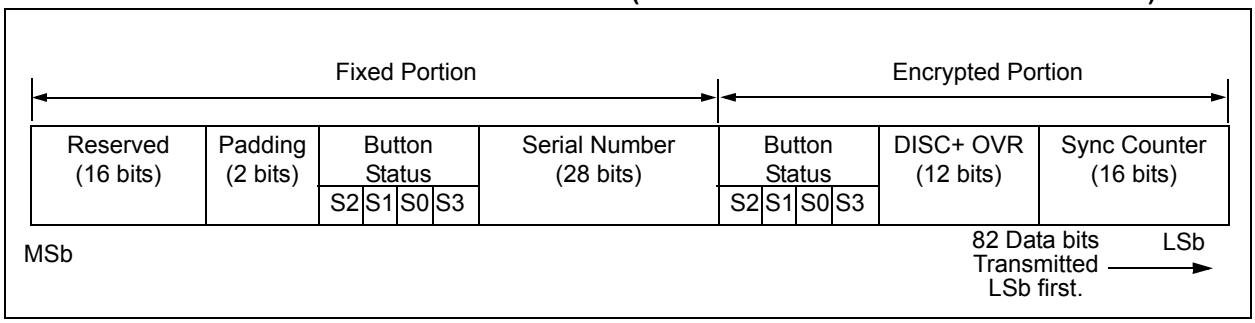


FIGURE 4-4: CODE WORD ORGANIZATION (SYNCHRONOUS TRANSMISSION MODE)



5.0 SPECIAL FEATURES

5.1 Code Word Completion

The code word completion feature ensures that entire code words are transmitted, even if the button is released before the code word is complete. If the button is held down beyond the time for one code word, multiple code words will result. If another button is activated during a transmission, the active transmission will be aborted and a new transmission will begin using the new button information.

5.2 $\overline{\text{LED}}$ Output Operation

During normal transmission the $\overline{\text{LED}}$ output is LOW. If the supply voltage drops below the low voltage trip point, the $\overline{\text{LED}}$ output will be toggled at approximately 5Hz during the transmission (Section 3.6.4).

5.3 RPT: Repeat Indicator

This bit will be low for the first transmitted word. If a button is held down for more than one transmitted code word, this bit will be set to indicate a repeated code word and remain set until the button is released.

5.4 VLOW: Voltage LOW Indicator

The VLOW signal is transmitted so the receiver can give an indication to the user that the transmitter battery is low. The VLOW bit is included in every transmission (Figure 4-2 and Figure 9-4) and will be transmitted as a zero if the operating voltage is above the low voltage trip point. Refer to Figure 4-2. The trip point is selectable based on the battery voltage being used. See Section 3.6.3 for a description of how the low voltage trip point is configured.

5.5 Auto-shutoff

The Auto-shutoff function automatically stops the device from transmitting if a button inadvertently gets pressed for a long period of time. This will prevent the device from draining the battery if a button gets pressed while the transmitter is in a pocket or purse. This function can be enabled or disabled and is selected by setting or clearing the Auto-shutoff bit (see Section 3.5.1). Setting this bit high will enable the function (turn Auto-shutoff function on) and setting the bit low will disable the function. Time-out period is approximately 25 seconds.

5.6 Seed Transmission

In order to increase the level of security in a system, it is possible for the receiver to implement what is known as a secure learn function. This can be done by utilizing the seed value stored in EEPROM, transmitted only when all four button inputs are pressed at the same time (Table 5-1). Instead of the normal key generation inputs being used to create the crypt key, this seed value is used.

TABLE 5-1: PIN ACTIVATION TABLE

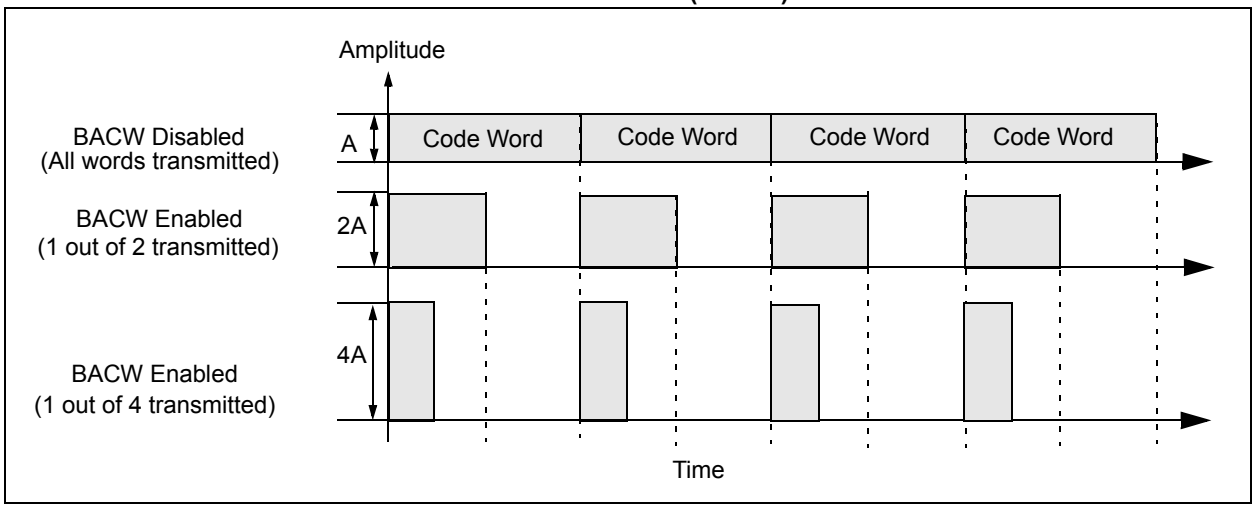
	Function	S3	S2	S1	S0
Standby	0	0	0	0	0
Hopping Code	1	0	0	0	1
	2	0	0	1	0
	-	-	-	-	-
	13	1	1	0	1
	14	1	1	1	0
Seed Code	15	1	1	1	1

5.7 Blank Alternate Code Word

Federal Communications Commission (FCC) part 15 rules specify the limits on worst case average fundamental power and harmonics that can be transmitted in a 100 ms window. For FCC approval purposes, it may therefore be advantageous to minimize the transmission duty cycle. This can be achieved by minimizing the duty cycle of the individual bits as well as by blanking out consecutive code words. Blank Alternate Code Word (BACW) may be used to reduce the average power of a transmission by transmitting only every second code word (Figure 5-1). This is a selectable feature that is determined in conjunction with the baud rate selection bit BSL0.

Enabling the BACW option may likewise allow the user to transmit a higher amplitude transmission as the time averaged power is reduced. BACW effectively halves the RF on time for a given transmission so the RF output power could theoretically be doubled while maintaining the same time averaged output power.

FIGURE 5-1: BLANK ALTERNATE CODE WORD (BACW)

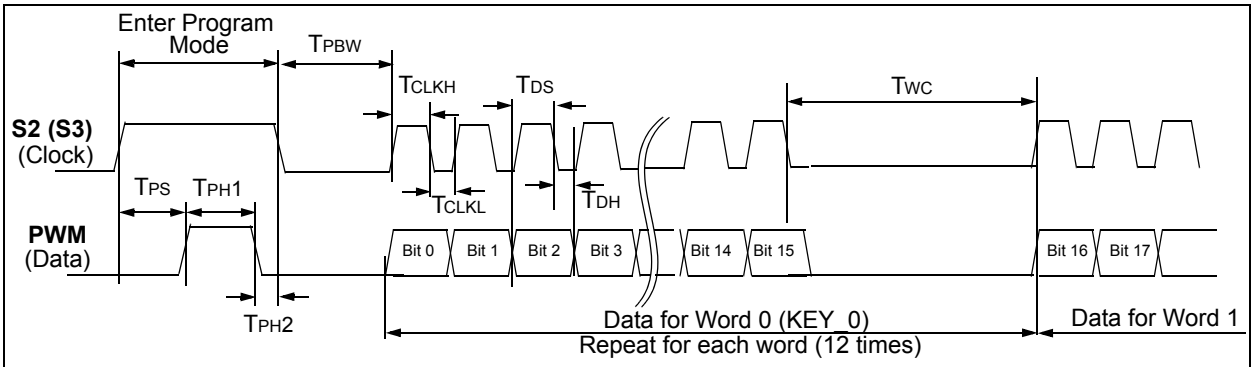


6.0 PROGRAMMING THE HCS300

When using the HCS300 in a system, the user will have to program some parameters into the device including the serial number and the secret key before it can be used. The programming cycle allows the user to input all 192 bits in a serial data stream, which are then stored internally in EEPROM. Programming will be initiated by forcing the PWM line high, after the S2 (or S3) line has been held high for the appropriate length of time line (Table 6-1 and Figure 6-1). After the Program mode is entered, a delay must be provided to the device for the automatic bulk write cycle to complete. This will set all locations in the EEPROM to zeros. The device can then be programmed by clocking in 16 bits at a time, using S2 (or S3) as the clock line and PWM as the data in line. After each 16-bit word is loaded, a programming delay is required for the internal program cycle to complete. This delay can take up to TWC. At the end of the programming cycle, the device can be verified (Figure 6-2) by reading back the EEPROM. Reading is done by clocking the S2 (or S3) line and reading the data bits on PWM. For security reasons, it is not possible to execute a verify function without first programming the EEPROM. **A Verify operation can only be done once, immediately following the Program cycle**.

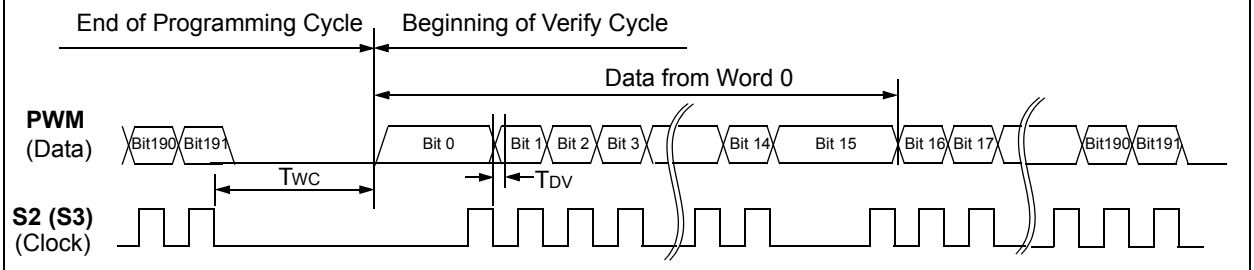
Note:	To ensure that the device does not accidentally enter Programming mode, PWM should never be pulled high by the circuit connected to it. Special care should be taken when driving PNP RF transistors.

FIGURE 6-1: PROGRAMMING WAVEFORMS



Note 1: Unused button inputs to be held to ground during the entire programming sequence.
2: The VDD pin must be taken to ground after a Program/Verify cycle.

FIGURE 6-2: VERIFY WAVEFORMS



Note: If a Verify operation is to be done, then it must immediately follow the Program cycle.

TABLE 6-1: PROGRAMMING/VERIFY TIMING REQUIREMENTS

V_{DD} = 5.0V ± 10%, 25 °C ± 5 °C				
Parameter	**Symbol**	**Min.**	**Max.**	**Units**
Program mode setup time	T_{PS}	3.5	4.5	ms
Hold time 1	T_{PH1}	3.5	—	ms
Hold time 2	T_{PH2}	50	—	µs
Bulk Write time	T_{PBW}	4.0	—	ms
Program delay time	T_{PROG}	4.0	—	ms
Program cycle time	T_{WC}	50	—	ms
Clock low time	T_{CLKL}	50	—	µs
Clock high time	T_{CLKH}	50	—	µs
Data setup time	T_{DS}	0	—	µs(1)
Data hold time	T_{DH}	30	—	µs(1)
Data out valid time	T_{DV}	—	30	µs(1)

Note 1: Typical values - not tested in production.

7.0 INTEGRATING THE HCS300 INTO A SYSTEM

Use of the HCS300 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 HCS300 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

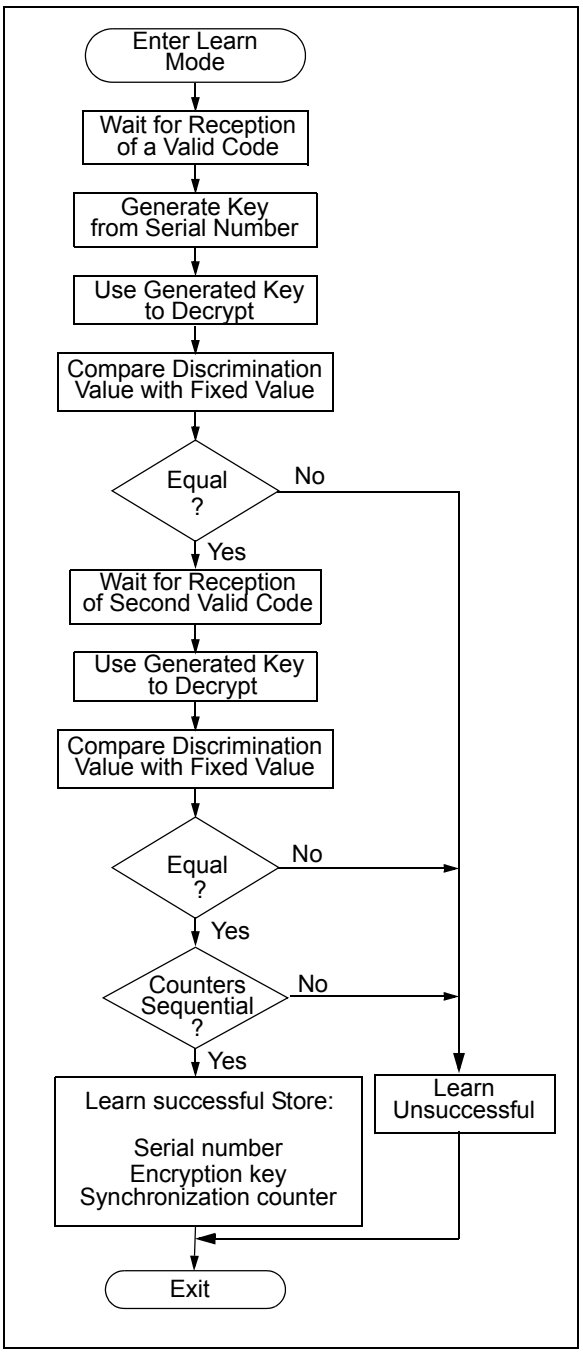
A transmitter must first be 'learned' by a decoder before its use is allowed in the system. Several learning strategies are possible, Figure 7-1 details a typical learn sequence. Core to each, the decoder must minimally store each learned transmitter's serial number and current synchronization counter 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 clear 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 is programmed with the ability to generate a crypt key as well as all but one required input to the key generation routine; typically the transmitter's serial number.

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 complete, 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 and care must be taken not to infringe.

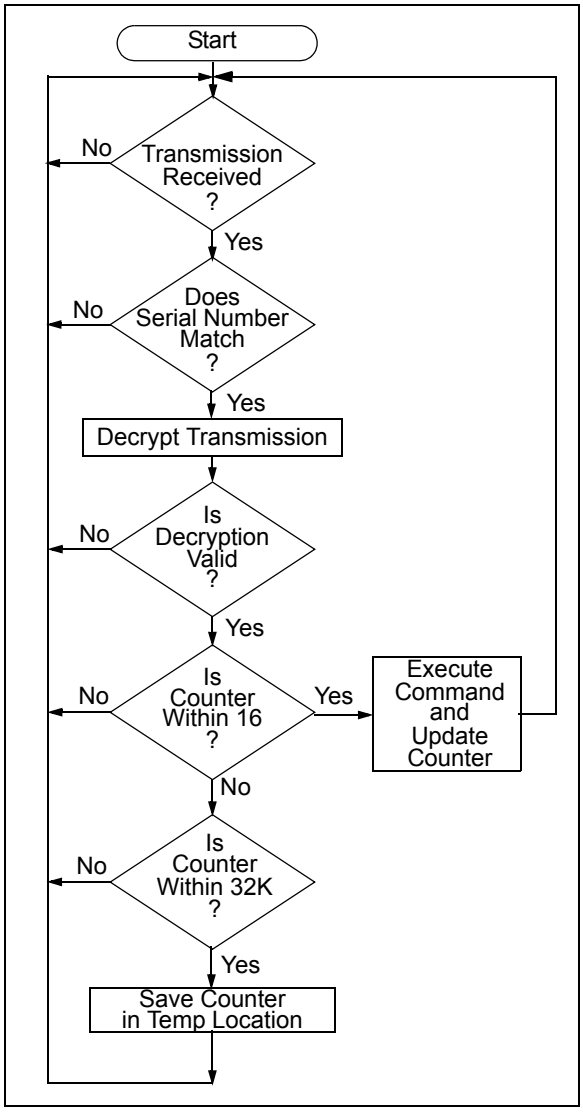
FIGURE 7-1: TYPICAL LEARN SEQUENCE



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.

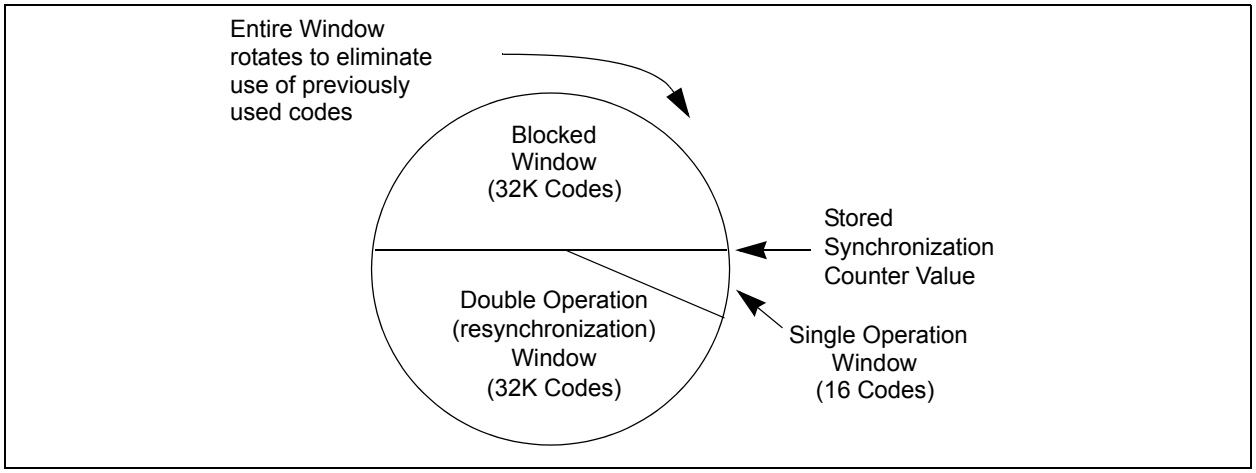
Figure 7-3 shows a 3-partition, rotating synchronization window. The size of each window is optional but the technique is fundamental. Each time a transmission is authenticated, the intended function is executed and the transmission's synchronization counter value is stored in EEPROM. From the currently 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



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 VDDMIN and VDDMAX 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

TABLE 9-1: ABSOLUTE MAXIMUM RATINGS

Symbol	Item	Rating	Units
V_{DD}	Supply voltage	-0.3 to 6.6	V
V_{IN}	Input voltage	-0.3 to V_{DD} + 0.3	V
V_{OUT}	Output voltage	-0.3 to V_{DD} + 0.3	V
I_{OUT}	Max output current	50	mA
T_{STG}	Storage temperature	-55 to +125	°C **(Note)**
T_{LSOL}	Lead soldering temp	300	°C **(Note)**
V_{ESD}	ESD rating	4000	V

Note: Stresses above those listed under “ABSOLUTE MAXIMUM RATINGS” may cause permanent damage to the device.

TABLE 9-2: DC CHARACTERISTICS

Commercial (C): Tamb = 0 °C to +70 °C Industrial (I): Tamb = -40 °C to +85 °C									
		2.0V < V_{DD} < 3.0			3.0 < V_{DD} < 6.3				
Parameter	**Sym.**	**Min.**	**Typ.[1]**	**Max.**	**Min.**	**Typ.[1]**	**Max.**	**Unit**	**Conditions**
Operating current (avg)[2]	I_{CC}		0.2	1		1.0	2.5	mA	V_{DD} = 3.0V V_{DD} = 6.3V
Standby current	I_{CCS}		0.1	1.0		0.1	1.0	µA	
Auto-shutoff current[3,4]	I_{CCS}		40	75		160	650	µA	
High level Input voltage	V_{IH}	0.55V_{DD}		V_{DD}+ 0.3	0.55V_{DD}		V_{DD}+ 0.3	V	
Low level input voltage	V_{IL}	-0.3		0.15V_{DD}	-0.3		0.15V_{DD}	V	
High level output voltage	V_{OH}	0.6V_{DD}			0.6V_{DD}			V	I_{OH} = -1.0 mA V_{DD} = 2.0V I_{OH} = -2.0 mA V_{DD} = 6.3V
Low level output voltage	V_{OL}			0.08V_{DD}			0.08V_{DD}	V	I_{OL} = 1.0 mA V_{DD} = 2.0V I_{OL} = 2.0 mA V_{DD} = 6.3V
$\overline{\text{LED}}$ sink current[5]	I_{LED}	1.0	1.8	2.5	2.0	2.7	3.7	mA	V_{LED}[6] = 1.5V V_{DD} = 3.0V V_{LED}[6] = 1.5V V_{DD} = 6.3V
Pull-down Resistance; S0-S3	R_{S0-3}	40	60	80	40	60	80	kΩ	V_{DD} = 4.0V
Pull-down Resistance; PWM	R_{PWM}	80	120	160	80	120	160	kΩ	V_{DD} = 4.0V

Note 1: Typical values are at 25 °C.
2: No load.
3: Auto-shutoff current specification does not include the current through the input pull-down resistors.
4: These values are characterized but not tested.
5: With V_{LOW} Sel = 0 for operation from 2.0V to 3.0V and V_{LOW} Sel = 1 for operation from 3.0V to 6.3V.
6: V_{LED} is the voltage drop across the terminals of the LED.

FIGURE 9-1: POWER-UP AND TRANSMIT TIMING

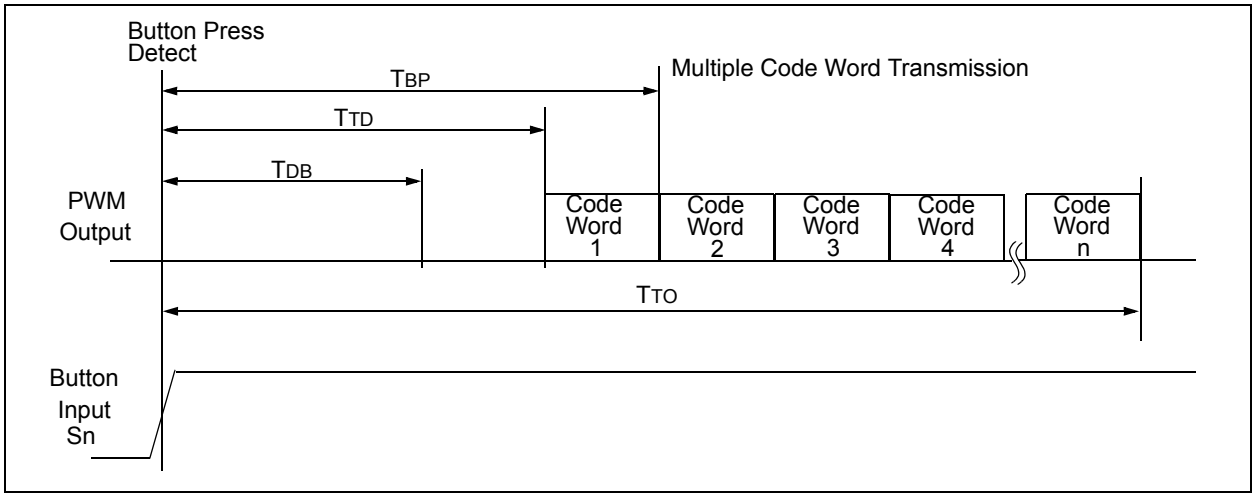


TABLE 9-3: POWER-UP AND TRANSMIT TIMING(2)

V_{DD} = +3.5 to 13.0V Commercial(C): Tamb = 0°C to +70°C Industrial(I): Tamb = -40°C to +85°C					
Symbol	**Parameter**	**Min**	**Max**	**Unit**	**Remarks**
T_{BP}	Time to second button press	10 + Code Word	26 + Code Word	ms	**(Note 1)**
T_{TD}	Transmit delay from button detect	10	26	ms	
T_{DB}	Debounce Delay	6	15	ms	
T_{TO}	Auto-shutoff time-out period	20	120	s	

Note 1: T_{BP} is the time in which a second button can be pressed without completion of the first code word and the intention was to press the combination of buttons.

2: Typical values - not tested in production.

FIGURE 9-2: CODE WORD FORMAT

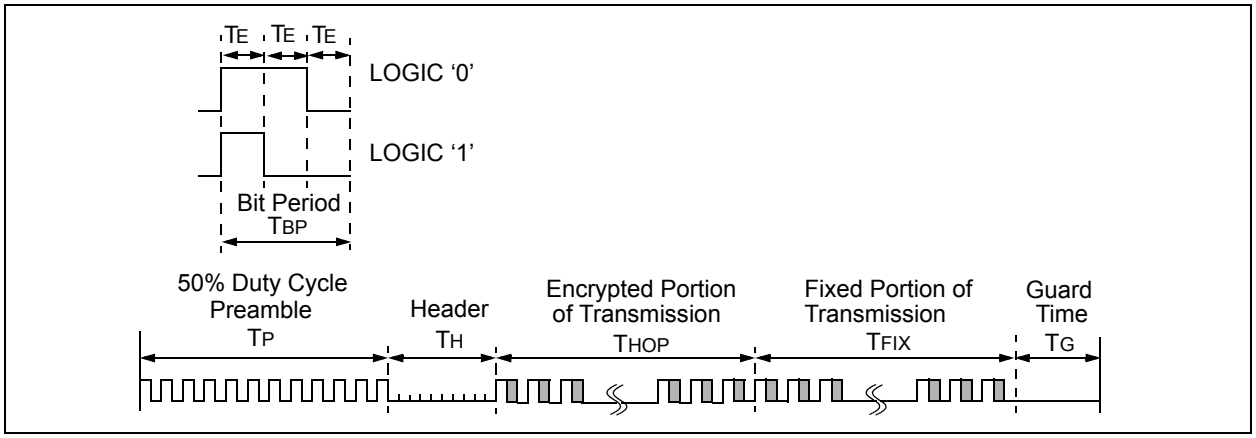


FIGURE 9-3: CODE WORD FORMAT: PREAMBLE/HEADER PORTION

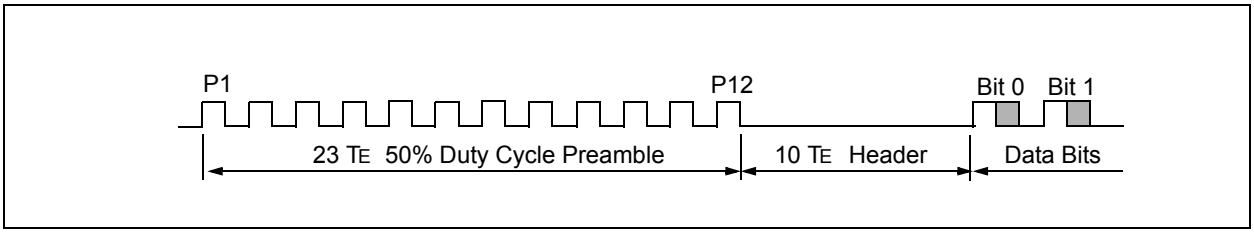


FIGURE 9-4: CODE WORD FORMAT: DATA PORTION

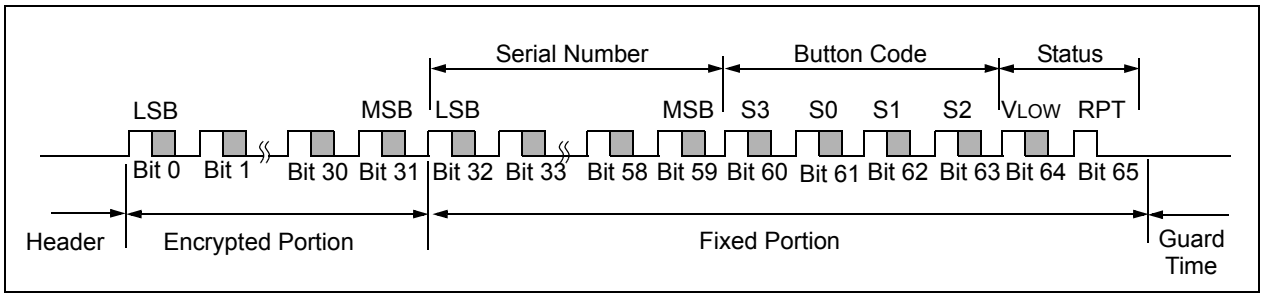
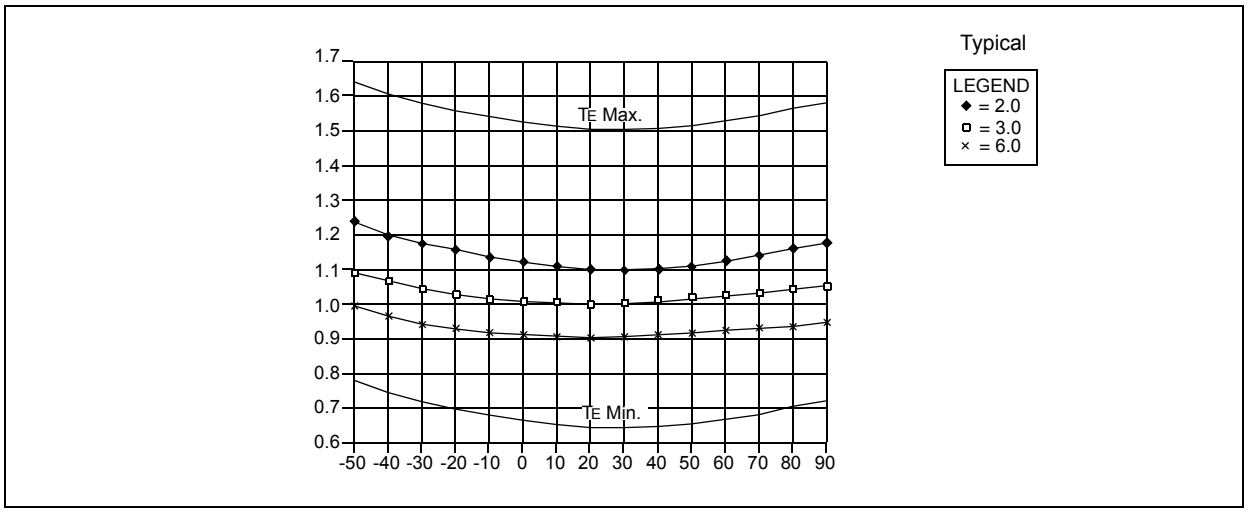


TABLE 9-4: CODE WORD TRANSMISSION TIMING REQUIREMENTS

VDD = +2.0 to 6.0V Commercial(C):Tamb = 0 °C to +70 °C Industrial(I):Tamb = -40 °C to +85 °C			Code Words Transmitted									
			All			1 out of 2			1 out of 4			
Symbol	**Characteristic**	**Number of TE**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Units**
TE	Basic pulse element	1	260	400	660	130	200	330	65	100	165	µs
TBP	PWM bit pulse width	3	780	1200	1980	390	600	990	195	300	495	µs
TP	Preamble duration	23	6.0	9.2	15.2	3.0	4.6	7.6	1.5	2.3	3.8	ms
TH	Header duration	10	2.6	4.0	6.6	1.3	2.0	3.3	0.7	1.0	1.7	ms
THOP	Hopping code duration	96	25.0	38.4	63.4	12.5	19.2	31.7	6.2	9.6	15.8	ms
TFIX	Fixed code duration	102	26.5	40.8	67.3	13.3	20.4	33.7	6.6	10.2	16.8	ms
TG	Guard Time	39	10.1	15.6	25.7	5.1	7.8	12.9	2.5	3.9	6.4	ms
—	Total Transmit Time	270	70.2	108.0	178.2	35.1	54.0	89.1	17.6	27.0	44.6	ms
—	PWM data rate	—	1282	833	505	2564	1667	1010	5128	3333	2020	bps

Note: The timing parameters are not tested but derived from the oscillator clock.

FIGURE 9-5: HCS300 TE VS. TEMP



10.0 PACKAGING INFORMATION

10.1 Package Marking Information

8-Lead PDIP

Example

8-Lead SOIC

Example

Legend:	XX...X	Customer specific information*
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

* Standard PIC device marking consists of Microchip part number, year code, week code, and traceability code. For PIC device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.