

## **APPLICATION NOTE**

# AN463 68HC05K0 Infra-red Remote Control

Tony Breslin, MCU Applications Group, Freescale Ltd., East Kilbride, Scotland

The MC68HC05K0 is a low cost, low pin count single chip microcomputer with 504 bytes of user ROM and 32 bytes of RAM. The MC68HC05K0 is a member of the 68HC05K series of devices which are available in 16-pin DIL or SOIC packages. It uses the same CPU as the other devices in the 68HC05 family and has the same instructions and registers. Additionally, the device has a 15-stage multi-function timer and 10 general purpose bi-directional I/0 lines. A mask option is available for software programmable pull-downs on all of the I/O pins and four of the pins are capable of generating interrupts.

The device is ideally suited for remote-control keyboard applications because the pull-downs and the interrupt drivers on the port pins allow keyboards to be built without any external components except the keys themselves. There is no need for external pull-up or pull-down resistors, or diodes for wired-OR interrupts, as these features are already designed into the device.

This application makes use of many of the device features to control an infra-red television remote control. The application could be very easily modified to control any device with a similar transmission protocol. It will run on any of the 'K' devices without modification.

### **Remote Control Specifications**

The basic purpose of a television remote control is to transmit a control instruction to the television. The instruction is generated by a keystroke on the remote control keyboard. The detection and decoding of a key press and the transmission encoding is carried out by the remote control micro controller. When a key on the remote control keypad is pressed, the micro controller must first determine what key is being pressed and generate an individual code for the key. The key code is then converted to a instruction code that is inserted into the transmission command which, using a defined protocol, is transmitted to the television receiver. The command is continually transmitted as long as the key is being held down.

As the remote control is battery powered it needs to use as little power as possible. This is achieved by entering STOP mode when no keys are being pressed and effectively switches off the device. The micro controller comes out of STOP mode upon receipt of an interrupt request that is generated when a key is pressed.

### **Remote Control Keyboard**

The 68HC05K0 has ten general purpose I/O pins. One of these is used for the transmission signal output leaving nine pins for the keyboard control. Of these, four pins on PortA have internal interrupt request hardware. Using these four pins as inputs allows key presses to be detected without any external interrupt hardware. This leaves the five remaining pins for outputs.

Using the internal pull-down facility and the rising edge interrupt request on the four inputs permits interrupts to be generated. If the five outputs are set to logic '1', so driving an input from logic '0' to logic '1' when a key is pressed, an interrupt request can be generated. Using this arrangement a five by four keyboard matrix can be used. An extra four keys can be controlled if the Vdd line is used to drive one row of four keys to logic '1'. Therefore the maximum amount of keys

	•			חחע	3	31		32		34		38
1	2	3	NORM	VUU	11		12		13		00	
4	5	6	MUTE	A7	7 14	1	15	72	16	74	01	78
7	8	9	VOL+	A6	b 17	1	18	b2	19	b4	06	b8
0	PC+	PC-	VOL-	A5	d	1		d2		d4		d8
					10		2c		2d		07	
TV/	МІХ	TIME	CON+	Α4	e	1		e2		e4		e8
TEXT	WIIX		00111	,,,,	39		3b		3a		0c	
STOP	SUB-		CON-	BO	f	1		f2		f4		f8
0101	PAGE	INDEX		DU	3e		3d		3c		0d	
					A0		A	1	A	2	A	3

Figure 1 Keyboard layout with associated scanned and transmitted codes

A depressed key will set one of the input columns to logic '1'. By scanning the columns, and setting each row output to logic '0' and then checking if the inputs all become logic '0', the associated row for the key can be determined. If rotating the logic '0' through the five output pins fails to identify a key column, then the key must be connected to the Vdd line. This process gives an individual code for each key which is a combination of the code from the column inputs and the row outputs. This can then be decoded to an instruction that is inserted into the output signal for transmission.

Figure 1 shows the layout of the keyboard on the left and the scanned and transmitted codes on the right. The keyboard layout incorporates the various television controls plus controls for TELETEXT. On the left hand side the codes returned from scanning the keyboard are shown in the upper right-hand corner of each key and the code sent for transmission for that key instruction are shown in the bottom left-hand corner. The I/O pins for each row and column are also shown for each key.

## **Transmission Protocol**

The transmission protocol in this application is that used by the MC144105 IR Remote Control Transmitter. It uses a binary coded 9-bit data word with the LSB being transmitted first. Each bit of the transmitted signal is in the form of a bi-phase pulse code modulated (PCM) signal, whose bit coding is shown in Figure 2. For a transmitted '0' there is a 512 $\mu$ s pause followed by a 32kHz pulse train for 512 $\mu$ s. For a transmitted '1' there is 32kHz pulse train followed by a 512 $\mu$ s pause. This gives a bit time of 1024 $\mu$ s for all bits.This is shown asFigure 2.



Figure 2 Bit coding of PCM signal



A complete transmission command consists of several messages. Each command begins with a start message of nine 1's followed by the message appropriate to the key pressed. This message is repeatedly transmitted until the key is released. The transmission is terminated after the key is released by a end message of nine 1's.

Every message consists of a pre-bit, a pre-bit pause, a start bit and nine data bits. The pre-bit and the start-bit are always logical '1'. The pre-bit allows for the set up of the automatic gain control in the receiving preamplifier. Figure 3 gives the exact timing relationships for the transmissions. The command timing in Figure 3 shows that after the start transmission the message is continually re-transmitted at intervals of 131ms (approximately 8Hz) until the key is released. This is shown as time (c). The control timing shows the nine bit instruction 111001110 being transmitted starting with the LSB. The pre-bit pause is equal to two bit periods and is followed by a start-bit of logical '1'. The pulse train is continuous during the transition between transmitting a logic '0' and a logic '1'. The modulating pulse train has a frequency of approximately 32kHz with a mark-to-space ratio of one to three.

The signal for transmission is output through one port pin and is used to drive an IR diode amplifier circuit.



Figure 3 Circuit timing

### For More Information On This Product, Go to: www.freescale.com



Figure 4 is a flow diagram showing the operation of the remote control on power-up or reset. After the initial set-up of the ports as inputs or outputs the remote control goes into STOP mode. It will remain in STOP mode as long as the device is not reset or a key is not pressed. When a key is pressed an interrupt request is generated. A short time delay makes sure that it is a true key press and not noise and also allows time for any switching effects on the inputs to pass prior to checking the inputs.

The keyboard is then read to find which key has been pressed and the code for the key is decoded into an instruction and transmitted to the television. If the key is held down the instruction is re-transmitted until the key is released. This is useful for the instructions which count through the television channels or adjust the volume, colour or brightness controls.

When the key is released a terminating instruction is sent to the receiver to inform it that the next message received is a separate instruction. This is useful in the case of a one time instruction like sending a channel number. In this example the receiver will tune to a channel only once; to tune to another channel the key must be released and a new instruction sequence received.

After terminating the transmission the ports are reset ready for the next key press and the processor returns to the STOP mode.



Figure 4 Flow diagram





### Hardware

The remote control circuit is shown in Figure 5. The hardware consists of the keyboard, the oscillator and the infra-red amplifier. The oscillator can be a crystal or a ceramic resonator with a frequency of 2MHz. The oscillator frequency is important since the transmission timing is based around a 1MHz internal clock frequency.

The infra-red amplifier uses two transistors and two standard diodes to limit the current through the IR diodes to approximately 1A. There is a need for a large capacitor close to the IR diodes because of the high switching current of the circuit.



Figure 5 Infra-red remote control





### Software

The listing of the remote control assembler code is contained at the end of this application note. The first section of the listing sets up the ports prior to going into STOP mode and waiting for a key to be pressed. PortA bits 0-3 are set up as inputs with the pull-downs enabled. Bits 4-7 are set up as outputs logic '1' as is PortB bit 0. PortB bit 1 is set-up as output logic '0' to switch off the IR amplifier before going into STOP mode.

The next section of code named 'presd' is the routine pointed to by the interrupt vector and is entered when a key is pressed. This routine first calls the keyboard scanning routine to determine which key has been pressed. It then calls the decoding routine to convert the code from the keyboard to a code that will be accepted by the television. The start message is then transmitted and is followed by the instruction message. There is then a check to see if the same key is still being pressed. If it is then the instruction message is re-transmitted until the key is released and the end message is transmitted.

As the transmission protocol requires nine data bits and only one byte instructions are being decoded a flag has to be set for the ninth bit of the transmission routine. For the start and end transmissions this flag is set to 1 to give the nine 1's message. For all instructions the ninth bit is 0 so the flag is cleared.

The decoding routine compares the code from the keyboard scan routine with data array 'keydat'. On a match it takes the corresponding element from the array 'tvdat' as the instruction code for transmission.

The values of the instruction codes shown in the right-hand side of Figure 1 are specific for the receiver application. Each receiver using the same communications protocol will receive the same nine bit instruction but what the instruction does is

dependent upon the receiver software. In this example the eight bit instruction '14' changes the channel to number four. In another receiver application the receiver may interpret the instruction code '14' as increase volume.

The transmission routine is entered with the instruction for transmission in 'keyst3'. After the pre-bit and the start-bit are transmitted the instruction byte is rotated (LSB first) into the carry flag. A logic '1' is sent for transmission if the flag is set after rotation and a logic '0' is sent for transmission if the flag is cleared. Each bit is transmitted as shown in Figure 1. The routines 'send0' and 'send1' send a pause of 512us followed by a 32kHz pulse train for 512µs and a 32kHz pulse train for  $512\mu$ s followed by a  $512\mu$ s pause respectively. In the situation when a '1' follows a '0' then a pulse train of  $1024\mu s$  is required. To avoid breaks in this pulse train the 'send0' routine checks the next bit to be transmitted to see if a double length pulse train must be transmitted. The 'send1' routine then has to check that a double length pulse train has not been sent in the previous one and a half bit periods before sending a pulse train.

The routine 'burst' produces the 32kHz pulse train for a duration set by a count in the accumulator. As the instruction time for setting the PortB bit 1 pin high or low is five clock cycles then the minimum processor clock period is derived by dividing the minimum output state time, which is 8 $\mu$ s when the output is high, by the minimum number of clock cycles to change this state. This gives an internal clock period of 8 $\mu$ s/5 equalling 1.6 $\mu$ s. Adding a three cycle delay will require an internal clock period of 8 $\mu$ s/8 = 1 $\mu$ s, allowing a 2MHz oscillator to be used.

The code size is approximately 300 bytes, leaving memory space for more features to be added to the controller.



### Debug

On applying power to the circuit the RESET vector will initialise the program counter at the beginning of the software. When examining the output at PortB bit 1 with an oscilloscope or logic analyser it should be noted that when trying to capture the signal by pressing a key the first signal out will be the start message of nine 1's. To capture the instruction the key should be held down and as the instruction will be continually re-transmitted then the capture can be initiated at this point.



## Listing

0020			* * * * * * *	* * * * * * * *	* * * * * * * * * * * * * * * * * * *	* * * * * * * * * * * * * * * * * * * *
0007			+			r0 vr1
0027			^ INFRA	RED REM	OIE CONIROL FOR F	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
0028						
0029			* WRITT	EN BY A.	BRESLIN 13.1.9	*
0030			* * * * * * *	* * * * * * * *	* * * * * * * * * * * * * * * * * * *	* * * * * * * * * * * * * * * * * * * *
0031			* THIS	PROGRAM 3	READS AND ENCODES	5 A KEY FROM A 24 KEY KEYBOARD *
0032			* TO A 1	FORM OF	BIPHASE PULSE COI	DE MODULATION (PCM) FOR INFRA *
0033			* RED T	RANSMISS	ION. IT USES THE	TRANSMISSION PROTOCOL OF THE *
0034			* MC144	105 IR R	EMOTE CONTROL TRA	ANSMITTER *
0035			******	******	* * * * * * * * * * * * * * * * * * * *	*****
0036						
0030						
0037	0000		nowto		0.0	
0038	0000		porta	equ	00	
0039	0001		portb	equ	10	
0040	0004		ddra	equ	04	
0041	0005		ddrb	equ	05	
0042	0008		tcsr	equ	\$08	
0043	0010		papd	equ	\$10	
0044						
0045	00e0			org	\$e0	
0046				5		
0047	00e0		kevst1	rmb	1	; initial code from keyboard
0048	0001		kovet 2	rmb	1	; keycode
0040	0002		kovat2	rmb	1	; godo transmittod
0049	00e2		dflag	runo	1	; flog for logt and 0th bits
0050	0000		urrag	TIID	T	, ITAY IOF TASE AND SEN DIES
0051						
0052						
0053			* * * * * * *	* * * * * * * *	* * * * * * * * * * * * * * * * * * *	*****
0053 0054			****** * THE P	******** ORTS ARE	**************************************	**************************************
0053 0054 0055			****** * THE P * OF TH	******* ORTS ARE E INTERN	SET UP USING POF AL INTERUPT GENEF	RTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/0 LINES. *
0053 0054 0055 0056			* * * * * * * * * * * * * * * * * * *	******** ORTS ARE E INTERN MODE IS	**************************************	RTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * KEY IS PRESSED *
0053 0054 0055 0056 0057			******* * THE P * OF TH: * STOP I ******	********* ORTS ARE E INTERN MODE IS *******	**************************************	RTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * (EY IS PRESED *
0053 0054 0055 0056 0057 0058			******* * THE PO * OF TH * STOP 1 *****	********* ORTS ARE E INTERN MODE IS ********	*********************** SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F ************	RTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * KEY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059	0200		******* * THE P( * OF TH * STOP ) *****	******** ORTS ARE E INTERN MODE IS : ********	SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F ***********************************	**************************************
0053 0054 0055 0056 0057 0058 0059 0060	0200		******* * THE P( * OF TH * STOP ) *****	ATTS ARE DRTS ARE E INTERN MODE IS ATTERN OTG	SET UP USING POF AL INTERUPT GENER ENTERED UNTIL A F	XTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * XEY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061	0200 0200 9a		******* * THE P * OF TH * STOP 1 ******	ATS ARE E INTERN MODE IS Org	SET UP USING POR AL INTERUPT GENER ENTERED UNTIL A F	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062	0200 0200 9a 0201 ad	04	******* * THE PU * OF TH: * STOP D *******	Cli	SET UP USING POF AL INTERUPT GENER ENTERED UNTIL A F \$200	ATTA 0-3 AS INPUTS MAKING USE * AATION ON THESE I/O LINES. * (EY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062	0200 0200 9a 0201 ad	04	******* * THE P * OF TH * STOP D ******* start wpres	Cli bsr	SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F \$200 setup	ATTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * CEY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063	0200 0200 9a 0201 ad 0203 9c	04	******* * THE P4 * OF TH * STOP 1 ******* start wpres	Cli bsr rsp	SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F ***********************************	XTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * XEY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064	0200 0200 9a 0201 ad 0203 9c 0204 8e	04	******* * THE P * OF TH * STOP P ******* start wpres	Cli bsr rsp stop	SET UP USING POR AL INTERUPT GENER ENTERED UNTIL A F ***********************************	ATTA 0-3 AS INPUTS MAKING USE * AATION ON THESE I/O LINES. * (EY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065	0200 9a 0201 ad 0203 9c 0204 8e 0205 20	04 fa	******* * THE P * OF TH * STOP D ******* start wpres	Cli bsr rsp stop bra	SET UP USING POR AL INTERUPT GENER ENTERED UNTIL A F ***********************************	ATTA 0-3 AS INPUTS MAKING USE * AATION ON THESE I/O LINES. * (EY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20	04 fa	******* * THE P * OF TH * STOP 1 ****** start wpres	Cli bsr stop bra	<pre>\$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$</pre>	ATTA 0-3 AS INPUTS MAKING USE * AATION ON THESE I/O LINES. * KEY IS PRESSED *
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6	04 fa f0	******* * THE P( * OF TH: * STOP ) ******* start wpres	ATS ARE E INTERN. MODE IS Cli bsr rsp stop bra lda	****************** SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F ***********************************	<pre>************************************</pre>
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7	04 fa f0 04	******* * THE PM * OF TH * STOP D ******* start wpres	Cli bsr rsp stop bra lda sta	**************** SET UP USING POF AL INTERUPT GENER ENTERED UNTIL A F ******************************** \$200 setup wpres #\$f0 ddra	<pre>XTA 0-3 AS IMPUTS MAKING USE * RATION ON THESE I/O LINES. * KEY IS PRESSED * ***********************************</pre>
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7	04 fa f0 04 00	******* * THE P( * OF TH * STOP ) ******* start wpres	Cli bsr rsp stop bra lda sta sta	**************** SET UP USING POR AL INTERUPT GENER ENTERED UNTIL A F ***********************************	<pre>XTA 0-3 AS INPUTS MAKING USE * XATION ON THESE I/O LINES. * XEY IS PRESSED * ***********************************</pre>
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7 020b b7	04 fa f0 04 00 10	******* * THE P4 * OF TH * STOP 1 ****** start wpres setup	CRTS ARE E INTERN. MODE IS cli bsr rsp stop bra lda sta sta sta	<pre>setup wpres #\$f0 ddra porta papd</pre>	<pre>// Constant in the second second</pre>
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0071	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 0200 b7 0200 b7 0204 a6	04 fa f0 04 00 10 03	******* * THE P( * OF TH: * STOP ) ******* start wpres setup	CRTS ARE E INTERN. MODE IS cli bsr rsp stop bra lda sta sta lda	<pre>************************************</pre>	<pre>xTA 0-3 AS INPUTS MAKING USE * xTION ON THESE I/0 LINES. * xEY IS PRESSED * ***********************************</pre>
0053 0054 0055 0056 0057 0058 0060 0061 0062 0063 0064 0065 0066 0067 0068 0066 0067 0068 0069 0070 0071	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7 020b b7 020b b7 0201 a6 0211 b7	04 fa f0 04 00 10 03 05	******* * THE P( * OF TH) * STOP ( ******* start wpres	Article Control of the second	<pre>************************************</pre>	<pre>XTA 0-3 AS INPUTS MAKING USE * RATION ON THESE I/O LINES. * XEY IS PRESSED * XEY IS PRESSED * XEY IS PRESSED * XEY A 1 A 1 A 1 A 1 A 1 A 1 A 1 A 1 A 1 A</pre>
0053 0054 0055 0056 0057 0058 0060 0061 0062 0063 0064 0065 00665 00667 0068 0069 0070 0071 0072	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7 020b b7 020b b7 020d b7 0201 a6 0211 b7 0213 a6	04 fa f0 04 00 10 03 05 01	******* * THE PM * OF TH * STOP N ******* start wpres setup	Arrow Construction of the second seco	<pre>************************************</pre>	<pre>x a 0-3 AS INPUTS MAKING USE * x aTION ON THESE I/O LINES. * x (EY IS PRESSED * ***********************************</pre>
0053 0054 0055 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0067 0068 0070 0071 0072 0073	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7 020b b7 020d b7 020f a6 0211 b7 0211 a6 0215 b7	04 fa f0 04 00 10 03 05 01 01	******* * THE P4 * OF TH * STOP 1 ******* start wpres setup	Cli bsr rsp bra lda sta sta lda sta sta lda sta sta sta sta sta sta	<pre>************************************</pre>	<pre>; porta 0-3 inputs ; porta 0-3 inputs ; 4-7 as outputs ; set outputs high ; 0-3 pulldown ; portb 0-1 outputs</pre>
0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070 0071 0072 0073 0074	0200 0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 0200 b7 0200 b7 0200 b7 0201 a6 0211 b7 0213 a6 0215 b7	04 fa f0 04 00 10 03 05 01 01	******* * THE PM * OF TH * STOP D ******* start wpres setup	Arts ARE E INTERN. MODE IS Cli bsr rsp stop bra lda sta sta lda sta lda sta rts	<pre>stress of the second seco</pre>	<pre>XTA 0-3 AS INPUTS MAKING USE * XATION ON THESE I/O LINES. * XEY IS PRESSED * XEY IS PRESSED * XEY A 1 A 2 A 2 A 2 A 2 A 2 A 2 A 2 A 2 A 2</pre>
0053 0054 0055 0056 0057 0058 0060 0061 0062 0063 0066 0065 0066 0067 0068 0069 0070 0071 0072 0073 0074 0075	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 0206 b7 0206 b7 0206 b7 0204 b7 0201 a6 0211 b7 0213 a6 0215 b7 0217 81	04 fa f0 04 00 10 03 05 01 01	****** * THE P( * OF TH: * STOP ) ******* start wpres setup	Arts ARE CRTS ARE E INTERN. MODE IS cli bsr rsp stop bra lda sta sta lda sta lda sta lda sta rts	<pre>************************************</pre>	<pre>XTA 0-3 AS INPUTS MAKING USE * XATION ON THESE I/O LINES. * XEY IS PRESSED * XEY IS PRESSED * XEY A-7 as outputs ; 4-7 as outputs ; set outputs high ; 0-3 pulldown ; portb 0-1 outputs ; set portb 0 high</pre>
0053 0054 0055 0056 0057 0058 0060 0061 0062 0063 0064 0065 0066 0067 0068 0066 0067 0068 0067 0068 0070 0071 0072 0073 0074	0200 9a 0201 ad 0203 9c 0204 8e 0205 20 0207 a6 0209 b7 020b b7 020b b7 020b b7 020b b7 020f a6 0211 b7 0213 a6 0215 b7 0217 81	04 fa f0 04 00 10 03 05 01 01	****** * THE P * OF TH * STOP D ******* start wpres setup	Arts ARE CRTS ARE E INTERN. MODE IS Cli bsr rsp stop bra lda sta sta sta lda sta lda sta rts	<pre>SET UP USING POF AL INTERUPT GENEF ENTERED UNTIL A F ***********************************</pre>	<pre>XTA 0-3 AS IMPUTS MAKING USE * XATION ON THESE I/O LINES. * XEY IS PRESSED * XEY IS PRESSED * XEY 1S PR</pre>



0078	* * * * * * *	* * * * * * *	* * * * * * * * * * * * * * *	*****************************
0079	* THE K	EY READ	IS DECODED FO	R TRANSMISSION. *
0080	* THE T	RANSMIS	SION PROTOCOL	REQUIRES A START MESSAGE OF 9 *
0081	* ONES	FOLLOWE	D BY THE KEYPR	ESSED CODE. THIS CODE IS *
0082	* CONTI	NUALLY	RETRANSMITTED	IF THE KEY IS HELD DOWN. AN END *
0083	* CODE	OF 9 ON	ES TERMINATES	THE TRANSMISSION AND THE DEVICE *
0084	* RETUR	NS TO S	TOP MODE	*
0085	******	******	************	*****
0086				
0007 0010 -4 24		la	1	· not loss meaned
0000 0218 au 34	presu	DSL	keyscii	, get key pressed
0088 021a b6 e1		Ida	keyst2	; save key to check
0089 021c b7 e0		sta	keystl	; if key held down
0090 021e ad 67		bsr	decode	; decode key pressed
0091 0220 12 e3		bset	1,dflag	; set nineth bit to 1
0092 0222 a6 ff		lda	#\$ff	; send start data
0093 0224 b7 e2		sta	keyst3	; to transmission routine
0094 0226 ad 71		bsr	trnmit	; nine one's
0095 0228 b6 e1	sndagn	lda	keyst2	; send key press message
0096 022a b7 e2		sta	keyst3	; byte
0097 022c 13 e3		bclr	1,dflag	; set nineth bit to 0
0098 022e ad 69		bsr	trnmit	
0099 0230 b6 00		lda	porta	; check if key still pressed
0100 0232 a4 0f		and	#\$0f	; end if no key pressed
0101 0234 26 0f		hne	endtrn	
0102 0236 ad 16		her	keysch	; else check if same
0102 $0230$ dd $10$		lda	kovat 1	; key proggod
0103 $0238$ $b0$ $e0$		Iua	keyst1	/ Key pressed
0104 023a DI EI		lame	Reysiz	· and if not
		Dile 1 des		
0106 0238 ae 68		lax	#\$C8	, delay
0107 0240 5a	tioop	aecx		; before next
0108 0241 26 Id		bne	tioop	; transmission
0109 0243 20 e3		bra	sndagn	
0110 0245 12 e3	endtrn	bset	1,dflag	; send end message
0111 0247 a6 ff		Ida	#ŞII	; of nine ones
0112 0249 b7 e2		sta	keyst3	
0113 024b ad 4c		bsr	trnmit	_
		rti		· ro-ontor stop modo
0114 0240 80		TOT		/ re-encer scop mode
0114 024d 80 0115		101		/ re-enter stop mode
0114 024d 80 0115 0116	******	******	* * * * * * * * * * * * * *	**************************************
0114 0240 80 0115 0116 0117	******* * WHEN	******* A KEY I	****************** S PRESSED THE	DEVICE COMES OUT OF STOP MODE *
0115 0115 0117 0118	******* * WHEN * THE K	******* A KEY I EYBOARD	******************* S PRESSED THE IS SCANNED TO	<pre>&gt; re-enter stop mode ************************************</pre>
0115 0116 0117 0118 0119	******* * WHEN * THE K ******	******* A KEY I EYBOARD ******	****************** S PRESSED THE IS SCANNED TO ****************	**************************************
0114 0248 80 0115 0116 0117 0118 0119 0120	******* * WHEN * THE K ******	******* A KEY I EYBOARD ******	***************** S PRESSED THE IS SCANNED TO **************	<pre>************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD *******	*************** S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terencer stop mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******* jsr lda	**************************************	<pre>/ IE-ERCEI SCOP Mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******** jsr lda sta	**************** S PRESSED THE IS SCANNED TO ******************* datwt porta keyst1	<pre>/ IE-ERCEI Stop Mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******* jsr lda sta and	*************** S PRESSED THE IS SCANNED TO ***************** datwt porta keyst1 #\$0f	<pre>/ Terenter stop mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******** lda sta and beq	************** S PRESSED THE IS SCANNED TO **************** datwt porta keyst1 #\$0f start	<pre>/ Terenter stop mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******** lda sta and beq ldx	**************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Terenter stop mode //  SEE WHICH KEY IS PRESSED //  //  SEE WHICH KEY IS PRESSED //  //  //  //  //  //  //  //  //  //</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f	******* * WHEN * THE K ******* keyscn	X KEY I EYBOARD X KEY I EYBOARD X I S X I da S X A A D A D A C A C A C A C A C A C A C A	S PRESSED THE IS SCANNED TO ****************** datwt porta keystl #\$0f start #\$ef	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // // // // // // // // // // // // //</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0	******* * WHEN * THE K ******* keyscn nxtrow	X KEY I EYBOARD X KEY I Ida sta and beq Idx txa and	**************************************	<pre>/ Terenter stop mode ************************************</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025c b7 e1	******* * WHEN * THE K ****** keyscn nxtrow	<pre>******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta</pre>	****************** S PRESSED THE IS SCANNED TO ***************** datwt porta keyst1 #\$0f start #\$ef keyst1 keyst1 keyst1 keyst1	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // // // // // // // // // // // // //</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0120 026 bf 00	******* * WHEN * THE K ******* keyscn	******* A KEY I EYBOARD ******* Ida sta and beq Idx txa and sta and sta	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop Stop Stop Mode // Stop Stop Stop Stop Stop Stop Stop Stop</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta sta sta	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Terenter stop mode // SEE WHICH KEY IS PRESSED * // SEE WHICH KEY IS PRESSED * // Set one debounce // check if key press // store inputs // mask outputs // mask outputs // store inputs // stop if no key pressed // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // to check in inputs // to check in</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 e4 0	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** lda sta and beq ldx txa and sta sta stx lda	<pre>S PRESSED THE IS SCANNED TO ****************** datwt porta keyst1 #\$of start #\$ef keyst1 keyst1 keyst2 porta porta porta</pre>	<pre>/ Terenter stop mode // Terenter stop m</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025c b4 e0 0129 025c b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 a4 0f	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda and sta	<pre>************************************</pre>	<pre>/ Terenter stop mode // Terenter stop m</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025c b7 e1 0120 026 b7 e1 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta stx lda and beq	<pre>************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // // // // // // // // // // // // //</pre>
0114 024d 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta sta stx lda and beq lslx	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // SEE WHICH KEY IS PRESSED // * // SEE WHICH KEY IS PRESSED // * // SEE which key press // store inputs // mask outputs // stop if no key pressed // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // zero in key-press row clears inputs // check if last row // Cerenter stop // Cerenter stop mode // Terenter stop mode // terent</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta stx lda sta stx lda sta stx lda sta stx lda sta	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // See WHICH KEY IS PRESSED * // See WHICH KEY IS PRESSED * // See WHICH KEY IS PRESSED * // See WHICH KEY IS PRESSED // // // // // // // // // // // // //</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** lda sta and beq ldx txa and sta stx lda and sta stx lda sta stx lda sta stx beq lslx incx bcc	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop mode //  // Stop of the store of the store inputs // stop if no key pressed // store now low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // zero in key-press row clears inputs // check if last row // set lsb to 1 // try portb output if not porta // ***********************************</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025c b7 e1 0130 0260 bf 00 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra	<pre>S PRESSED THE IS SCANNED TO ***************** datwt porta keyst1 #\$of start #\$ef keyst1 keyst2 porta porta #\$of gotit tryb nxtrow</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop mode // // // // // // // // // // // // //</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed 0138	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Terenter stop mode // SEE WHICH KEY IS PRESSED // * // SEE WHICH KEY IS PRESSED // * // SEE WHICH KEY IS PRESSED // * // Set one inputs // mask outputs // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // zero in key-press row clears inputs // check if last row // set lsb to 1 // try portb output if not porta // try next porta output row</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed 0138 0139 026e b6 e0	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta sta sta sta sta lda and beq lslx incx bcc bra lda	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop mode //  // SEE WHICH KEY IS PRESSED //  //  //  //  //  //  //  //  //  //</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed 0138 0139 026e b6 e0 0140 0270 b7 e1	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta stx lda and beq ldx txa incx bcc bra lda sta	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode //  DEVICE COMES OUT OF STOP MODE * //  SEE WHICH KEY IS PRESSED * //  //  //  //  //  //  //  //  //  /</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0127       025b       9f       0128       025c       b4       e0         0129       025e       b7       e1       0130       0260       bf       00         0131       0262       b6       00       0133       0266       27       1c         0133       0266       b7       e1       0133       0266       27       1c         0133       0266       27       1c       0133       0268       58       0135       0269       5c         0136       026a       24       02       0137       026c       20       ed       0138         0139       026e       b6       e0       0141       0270	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra lda sta lalx	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Terenter stop mode // SEE WHICH KEY IS PRESSED // * // SEE WHICH KEY IS PRESSED // * // See which flow free inputs // store inputs // store inputs // store inputs // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // zero in key-press row clears inputs // check if last row // set lsb to 1 // try portb output if not porta // try next porta output row</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0127       025b       9f       0128       025c       b4       e0         0129       025e       b7       e1       0130       0260       bf       00         0131       0262       b6       00       0133       0262       b7       e1         0133       0264       a4       0f       0133       0268       58       0135       0269       5c         0136       026a       24       02       0137       026c       20       ed         0139       026e       b6       e0       0140       0270       b7       e1         0140       0270       b7       e1       0141       0274       bf	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra lda sta lda sta lda sta	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // // // // // // // // // // // // //</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       a7         0126       0259       ae       ef         0127       025b       9f         0128       025c       b4       e0         0129       025e       b7       e1         0130       0260       bf       00         0131       0262       b6       00         0133       0266       27       1c         0134       0268       58       0135       0269         0136       026a       24       02       0137       026c       20       ed         0138       0139       026e       b6       e0       0140       0270       b7 <e1< td="">       0141       0272       ae       f0         0141       0272       ae       f0       0141       027</e1<>	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra lda sta lda sta bcc bra	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop in the store of the store inputs // wait for debounce // check if key press // store inputs // mask outputs // stop if no key pressed // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // zero in key-press row clears inputs // check if last row // set lsb to 1 // try portb output if not porta // try next porta outputs high // set portb 0 output low</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0129       025e       b7       e1         0130       0260       bf       00         0131       0262       b6       00         0133       0266       27       1c         0134       0268       58       0135       0269         0136       026a       24       02       0137       026c       20       ed         0138       0269       5c       0138       0139       026e       b6       e0         0140       0270       b7       e1       0141       0272       ae       f0         0142       0274       bf       00       0141       0272       f0       01	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra lda sta lda sta lda sta lda stx lda stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta sta ldx txa and beq ldx txa and beq ldx txa lda sta stx lda sta stx lda sta stx lda sta stx lda sta sta ldx txa and beq ldx txa and beq ldx txa lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta ldx txa lda sta stx lda sta sta lda sta sta lda sta sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta sta lda sta sta sta lda sta sta sta lda sta lda sta lda sta sta lda sta sta sta sta sta sta sta sta sta st	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // // // // // // // // // // // // //</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0127       025b       9f       0128       0260         0128       025c       b4       e0       0129       025e       b7       e1         0130       0260       bf       00       0131       0262       b6       00         0131       0262       b6       00       0132       0264       a4       0f         0133       0266       27       1c       0134       0268       58       0135       0269       5c         0136       026a       24       02       0137       026c       20       ed         0139       026e       b6       e0       0140       0270       b7       e1      0	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta stx lda sta stx lda sta stx lda sta and beq lslx incx bcc bra ldx sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and bcc bcc bra lda sta sta sta sta sta sta sta sta sta st	S PRESSED THE IS SCANNED TO ************************************	<pre>/ Terenter stop mode // SEE WHICH KEY IS PRESSED // * // Terenter stop mode // tere</pre>
0114       0246       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0127       025b       9f       0128       0260       bf<00	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda sta and beq lslx incx bcc bra ldx sta lslx incx bcc bra ldx sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta beq lslx incx bcc bra lda sta beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta and beq lslx incx bcc bra lda sta sta sta beq lslx incx bcc bra lda sta sta sta bcc bra lda sta sta sta sta sta sta sta sta sta st	<pre>S PRESSED THE IS SCANNED TO ****************** datwt porta keyst1 #\$0f start #\$0f gotit keyst2 porta porta #\$0f gotit tryb nxtrow keyst1 keyst2 #\$f0 porta 0,portb porta #\$0f gotit</pre>	<pre>/ Terenter stop mode // Terenter stop m</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 lc 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed 0138 0139 026e b6 e0 0140 0270 b7 e1 0141 0272 ae f0 0142 0274 bf 00 0143 0276 11 01 0144 0278 b6 00 0145 027a a4 0f 0146 027c 27 06 0147 027e b6 e1	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta lda and beq lslx incx bcc bra lda sta lda sta lda sta lda ata and beq lslx incx bcc bra lda sta sta lda sta lda sta sta lda sta sta lda sta sta lda sta sta sta lda sta sta sta sta sta sta sta sta sta st	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode //  DEVICE COMES OUT OF STOP MODE * //  SEE WHICH KEY IS PRESSED * //  //  //  //  //  //  //  //  //  /</pre>
0114 0246 80 0115 0116 0117 0118 0119 0120 0121 024e cd 02 fc 0122 0251 b6 00 0123 0253 b7 e0 0124 0255 a4 0f 0125 0257 27 a7 0126 0259 ae ef 0127 025b 9f 0128 025c b4 e0 0129 025e b7 e1 0130 0260 bf 00 0131 0262 b6 00 0131 0262 b6 00 0132 0264 a4 0f 0133 0266 27 1c 0134 0268 58 0135 0269 5c 0136 026a 24 02 0137 026c 20 ed 0138 0139 026e b6 e0 0140 0270 b7 e1 0141 0272 ae f0 0142 0274 bf 00 0143 0276 11 01 0144 0278 b6 00 0145 027a a4 0f 0146 027c 27 06 0147 027e b6 e1 0148 0280 a4 3f	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******* lda sta and beq ldx txa and sta lda and beq lslx incx bcc bra lda sta lda sta lda and beq lslx incx bcc bra lda sta and bcd lax ta and beq lax ta and bcc bra and bcc bra and bcc bra and bcc bra and bcc bra and bca and bcc bca bca bca lax and bca and bca bca bca bca bca bca bca bca bca bca	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ IE-Enter stop mode // IE-Enter stop mode // IE-Enter stop mode // IE-Enter stop mode // SEE WHICH KEY IS PRESSED // * // SEE WHICH KEY IS PRESSED // * // IE-Enter stop if no key pressed // stop if no key pressed // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // set lsb to 1 // try portb output if not porta // try next porta output shigh // set portb 0 output low // check for inputs cleared // mask outputs // set all porta outputs high // set portb 0 output low // check for inputs cleared // mask outputs // set individual code since last row // set individual code since last row // Set last row // set individual code since last row // set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since last row // Set last row // Set individual code since las</pre>
0114       024d       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0129       025e       b7       e1         0130       0260       bf       00         0131       0262       b6       00         0133       0266       27       1c         0134       0268       58       0135       0269         0136       026a       24       02       0137       026c       20       ed         0138       0269       5c       0136       0269       5c       0136       0269       5c         0139       026e       b6       e0       0141       0272       ae       f0         0141       0272       ae       f0       0144       0276       f1       01	******* * WHEN * THE K ******* keyscn nxtrow	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda and beq lslx incx bcc bra lda sta lda sta lda and beq lslx incx bcc bra lda sta and sta sta and sta sta and sta sta and sta sta and sta sta and sta sta and sta sta and sta sta and sta sta and sta sta sta and sta sta sta sta sta sta sta sta sta sta	<pre>S PRESSED THE IS SCANNED TO ************************************</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop if no key pressed // Stop if no key pressed // set one row low // read ouput lines // combine with inputs // store key code // to find row which clears inputs // check for inputs cleared // mask outputs // set lsb to 1 // try portb output if not porta // try next porta outputs high // set portb 0 output low // check for inputs cleared // mask outputs // set all porta outputs high // set portb 0 output low // check for inputs cleared // mask outputs // set individual code since last row // store code</pre>
0114       0246       80         0115       0116         0117       0118         0119       0120         0121       024e       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0129       025e       b7       e1         0130       0260       bf       00         0131       0262       b6       00         0132       0264       a4       0f         0133       0266       27       1c         0134       0268       58       0135       0269         0135       0269       5c       0136       0137         0139       026e       b6       e0       0140         0140       0270       b7       e1         0141       0272       ae       f0         0144       0274       bf       00         0143       0276       11       01 <td>******* * WHEN * THE K ******* keyson nxtrow tryb</td> <td>******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta stx lda sta stx lda sta beq lslx incx bcc bra ldx sta ldx txa and sta stx lda sta stx lda sta stx lda sta stx lda sta sta stx lda sta stx lda sta sta sta sta sta sta sta sta sta st</td> <td><pre>S PRESSED THE IS SCANNED TO ****************** datwt porta keyst1 #\$0f start #\$0f gotit tryb nxtrow keyst1 keyst2 #\$0f gotit keyst2 #\$10 porta 0,portb porta #\$0f gotit keyst2 #\$0f gotit keyst2 #\$0f gotit keyst2 #\$10 porta p</pre></td> <td><pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop mode // // // // // // // // // // // // //</pre></td>	******* * WHEN * THE K ******* keyson nxtrow tryb	******* A KEY I EYBOARD ******* jsr lda sta and beq ldx txa and sta stx lda sta stx lda sta beq lslx incx bcc bra ldx sta ldx txa and sta stx lda sta stx lda sta stx lda sta stx lda sta sta stx lda sta stx lda sta sta sta sta sta sta sta sta sta st	<pre>S PRESSED THE IS SCANNED TO ****************** datwt porta keyst1 #\$0f start #\$0f gotit tryb nxtrow keyst1 keyst2 #\$0f gotit keyst2 #\$10 porta 0,portb porta #\$0f gotit keyst2 #\$0f gotit keyst2 #\$0f gotit keyst2 #\$10 porta p</pre>	<pre>/ Terenter stop mode // Terenter stop mode // Terenter stop mode // Stop mode // // // // // // // // // // // // //</pre>
0114       0246       80         0115       0116         0117       0118         0119       0120         0121       0246       cd       02       fc         0122       0251       b6       00         0123       0253       b7       e0         0124       0255       a4       0f         0125       0257       27       a7         0126       0259       ae       ef         0127       025b       9f       0128       0260         0128       025c       b4       e0       0129       025e       b7       e1         0130       0260       bf       00       0131       0266       27       1c         0133       0266       bf       00       0133       0266       27       1c         0133       0266       5c       0136       0269       5c       0136       0269       5c         0136       0266       20       ed       0138       0139       0266       b6       e0         0140       0270       b7       e1       0144       0278       b6       00 <tr< td=""><td>******* * WHEN * THE K ******* keyscn nxtrow tryb</td><td>******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda sta stx lda sta beq lslx incx bcc bra ldx sta lda sta and beq lslx incx bcc bra lda sta sta stx lda sta sta stx lda sta sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta lda sta stx lda sta stx lda sta lda sta stx lda sta lda sta stx lda sta lda sta lda sta lda sta lta</td><td>S PRESSED THE IS SCANNED TO ************************************</td><td><pre>/ leventer stop mode // leventer stop m</pre></td></tr<>	******* * WHEN * THE K ******* keyscn nxtrow tryb	******* A KEY I EYBOARD ******** jsr lda sta and beq ldx txa and sta stx lda sta stx lda sta beq lslx incx bcc bra ldx sta lda sta and beq lslx incx bcc bra lda sta sta stx lda sta sta stx lda sta sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta stx lda sta lda sta stx lda sta stx lda sta lda sta stx lda sta lda sta stx lda sta lda sta lda sta lda sta lta	S PRESSED THE IS SCANNED TO ************************************	<pre>/ leventer stop mode // leventer stop m</pre>



0153	*****	******	* * * * * * * * * * * * * * * * * * *	**********			
0154	* THE I	DECODE RO	OUTINE USES TWO A	RRAYS. IT COMPARES THE KEY *			
0155	* VALUE WITH THE ARRAY KEYDAT AND WHEN A MATCH IS FOUND THE *						
0156	* CORRESPONDING ELEMENT IN THE ARRAY TVDAT BECOMES THE *						
0157	* TRANSMITTED CODE						
0158	IRANGHIIIED CODE.						
0159							
0150 0287 20 18	dogodo	ldr	#010	· data array offect to gave			
0100 0207 ae 18	uecoue	lda	#910	, data array orrset to zero			
0161 0289 d6 03 02	nxtel	Ida	keydat, x	i look at each element of array			
0162 028c bl el		cmp	keyst2	; compare with key read			
0163 028e 27 03		beq	match	; decode if match			
0164 0290 5a		decx		; else try next element			
0165 0291 26 f6		bne	nxtel	; norm if no match found			
0166 0293 d6 03 1a	match	lda	tvdat,x	; get key code			
0167 0296 b7 e1		sta	kevst2	; store code to transmit			
0168 0298 81		rts	- 1				
0169							
0170	* * * * * * *	* * * * * * * * *	* * * * * * * * * * * * * * * * * *	* * * * * * * * * * * * * * * * * * * *			
0171	* ጥህር ግ	PDANCMIC	STON DROTOCOL REO	א יידם_שמת ג יידם_שמת ג אשמדוו			
0171	* DAUG		SION PROIDCOL REQ	UIRES A FRE-BII, A FRE-BII			
0172	* PAUSE	s, a sta	RT BIT AND NINE D	ATA BITS, WHERE THE PRE-BIT *			
0173	* AND 1	THE STAR	I BIT ARE LOGIC '	1'. *			
0174	*****	* * * * * * * * *	* * * * * * * * * * * * * * * * * * *	**********			
0175							
0176 0299 10 e3	trnmit	bset	0,dflag	; initialise for first bit			
0177 029b ad 32		bsr	send1	; send pre-bit			
0178 029d cd 02 fc		jsr	datwt	; pre-bit pause			
0179 02a0 cd 02 fc		isr	datwt	; equalling four half data periods			
0180 02a3 cd 02 fc		isr	datwt	;			
0181 02a6 cd 02 fc		j~- ier	datwt	;			
0101 0200 cd 02 10		bar	aac#t	, and start bit			
0102 $02a9$ au 24		ldu	#400	· transmit 9 data bita			
0103 02aD ae 08		lux	#200 ha at 2	, transmit o data bits			
0184 02ad 34 e2	nxtDit	lsr	keyst3	; get next bit			
0185 02af 25 04		bcs	datal	; send 1 if carry set			
0186 02b1 ad 28		bsr	send0	; send 0 if carry clear			
0187 02b3 20 02		bra	bitsnt				
0188 02b5 ad 18	datal	bsr	sendl				
0189 02b7 5a	bitsnt	decx		; countdown bits sent			
0190 02b8 26 f3		bne	nxtbit	; send next bit if count not zero			
0191 02ba 03 e3 04		brclr	1,dflag,send00	; if flag set			
0192 02bd ad 10		hsr	send1	; send 1 as nineth bit			
0193 $02bf$ 20 02		bra	endend	:			
0195 0281 20 02	aand00	bra	cinacina	, also good 0			
0194 02C1 au 18	sendoud	DSI	#610	, eise send u			
0195 0203 ae 18	endend	Jax	#\$18				
0196 02C5 ad 35	Tooba	bsr	datwt	i delay between successive			
0197 02c7 ad 33		bsr	datwt	; transmissions			
0198 02c9 ad 31		bsr	datwt				
0199 02cb 5a		decx					
0200 02cc 26 f7		bne	loopw				
0201 02ce 81		rts					
0202							
0203							
0204	* * * * * * *	******	* * * * * * * * * * * * * * * * * *	******			
0205	* TO TI	ANSMIT	A LOGIC 11' A 32k	HZ DILLSE TRAIN FOR 512115 TS *			
0206	* EOTT		A 512110 DATISE	*			
0207	гоцц ******	**************************************	JIZUG FAUGE. ****	*****			
0207							
		, .	0 161 2 15				
U2U9 U2CI U1 e3 U4	sendl	prcir	u,dilag,last0	; CNECK II LAST DIT WAS ZERO			
0210 02d2 a6 10		⊥da	#\$10	; burst if last bit was l			
0211 02d4 ad 15		bsr	burst	; 32kHz pulse for 512us			
0212 02d6 ad 24	last0	bsr	datwt	; wait 512us			
0213 02d8 10 e3		bset	0,dflag	; set flag as 1 sent			
0214 02da 81		rts					
0215							



0216								* * * * * * *	* * * * * * * *	*****	******
0217								* TO TR	ANSMIT A	LOGIC '0' A 512	us PAUSE IS FOLLOWED BY A *
0218								* 32kHz	PULSE T	RAIN FOR 512us.	IF A LOGIC '1' FOLLOWS A '0' *
0219								* THE 3	2kHz IS	CONTINUED FOR 10	24us TO AVOID A PROCESSING *
0220								* DELAY			*
0221								******	******	*****	*****
0222											
0223	02db	ad	1f					send0	bsr	datwt	; wait 512us
0223	0244	00		04				bendo	breet	0 kovet3 novt1	; which sizes ; check if next bit is 1
0224	0200	26	10	01					lda	#¢10	; ginglo burgt if 1
0225	0200	20	0.2						hwa	#\$10 dataot	/ Single Duist II I
0220	02e2	20	02						bra	ualsel	, data set
0227	02e4	a6	20					nexti	laa	#\$∠U	; double burst required
0228	0266	aα	03					datset	bsr	burst	; 32KHZ pulse for 512us
0229	02e8	11	e⊰						bclr	0,dflag	; clear flag as 0 sent
0230	02ea	81							rts		
0231											
0232								* * * * * * *	* * * * * * * *	*****	*********
0233								* THE 3	2kHz PUL	SE TRAIN HAS A M	ARK TO SPACE RATIO OF 1 TO 3 *
0234								* * * * * * *	******	*****	*********
0235											
0236	02eb	13	01					burst	bclr	1,portb	; portb 1 low
0237	02ed	21	fe						brn	*	
0238	02ef	12	01						bset	1,portb	; portb 1 high
0239	02f1	21	fe						brn	*	
0240	02£3	13	01						bclr	1,portb	; portb 1 low
0241	02£5	9d							nop		
0242	02£6	4a							deca		; decrement count
0243	02£7	27	02						beq	endbur	; end of burst ?
0244	02£9	20	£0						bra	burst	
0245	02fb	81						endbur	rts		
0246											
0247											
0248	02fc	a6	52					datwt	lda	#\$52	; count
0249	0210 02fe	4a	52					loop	deca	πφ 5 2	; to provide 512us delay
0250	0210	26	fð					TOOD	bne	loon	: after instruction times
0250	0211	01	тu						wt a	1005	/ arter matraction times
0251	0301	01							ILS		
0252	0202	21	£ 1	- 1	.11	le 1	-7 1	1	£ ala	001 061 0a1 041	6H1 671
0255	0302	22	1 I 1 I	e1	40	DT DT	71	Reyual	feb	\$31,\$I1,\$E1,\$U1	, SDI, S/I
0254	0308	34	1 Z	ez	a2	D2	72		1CD field	\$32,\$12,\$e2,\$d2	, \$D2, \$/2
0255	030e	34	14	e4	4	104	74		LCD	\$34,\$14,\$e4,\$04	, \$D4, \$74
0256	0314	38	18	eð	a۶	80	/8		ICD	\$38,\$I8,\$e8,\$a8	, \$D8, \$78
0257	0.01		~	~ ~	1.0				<b>C</b> 1	*** *** ****	
0258	UJLA	11	3e	39	TÜ	17	14	tvdat	ICD	\$11,\$3e,\$39,\$10	,Ş1/,Ş14
0259	0320	12	3d	3b	2c	18	15		fcb	\$12,\$3d,\$3b,\$2c	,\$18,\$15
0260	0326	13	3c	3a	2d	19	16		fcb	\$13,\$3c,\$3a,\$2d	,\$19,\$16
0261	032c	00	0d	0c	07	06	01		fcb	\$00,\$0d,\$0c,\$07	,\$06,\$01
0262											
0263											
0264	0332	80						softin	rti		
0265											
0266	03fa								org	\$3fa	
0267											
0268	03fa	02	18						fdb	presd	; scan keybrd on int
0269	03fc	03	32						fdb	softin	; software interrupt
0270	03fe	02	00						fdb	start	; resett



### How to Reach Us:

Home Page:

www.freescale.com

E-mail: support@freescale.com

#### **USA/Europe or Locations Not Listed:**

Freescale Semiconductor Technical Information Center, CH370 1300 N. Alma School Road Chandler, Arizona 85224 +1-800-521-6274 or +1-480-768-2130 support@freescale.com

### Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH Technical Information Center Schatzbogen 7 81829 Muenchen, Germany +44 1296 380 456 (English) +46 8 52200080 (English) +49 89 92103 559 (German) +33 1 69 35 48 48 (French) support@freescale.com

#### Japan:

Freescale Semiconductor Japan Ltd. Headquarters ARCO Tower 15F 1-8-1, Shimo-Meguro, Meguro-ku, Tokyo 153-0064 Japan 0120 191014 or +81 3 5437 9125 support.japan@freescale.com

#### Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd. Technical Information Center 2 Dai King Street Tai Po Industrial Estate Tai Po, N.T., Hong Kong +800 2666 8080 support.asia@freescale.com

#### For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center P.O. Box 5405 Denver, Colorado 80217 1-800-441-2447 or 303-675-2140 Fax: 303-675-2150 LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



For More Information On This Product, Go to: www.freescale.com