Learning Encoder
(22+2 - Corresponds to HT6P20B)

D/N：HA0080E

Introductions

The HT6P20X series of devices from Holtek are 2^{24} Encoder OTP devices in which users can program their self edited 20~24 bit addresses, more details of which can be found in the HT6P20X datasheet. The Software Intellectual Property (SWIP) within this application can fully emulate the functions of the HT6P20X and uses and uses a HT93LC46 device as memory. Because the HT93LC46 is an EEPROM device (can be erased), this SWIP, in addition to emulating the HT6P20X functions, also incorporates additional functions. The application can change an unlimited number of times the address to be transmitted. The user only has to follow the accompanying circuit, and embed the Learning Encoder Main Program (main.asm) into their own main program as well as including the following 4 files ″HT93LC46.ASM″, ″KEY DEBOUNCE.ASM″, ″TRANSMIT_MODE.ASM″ and ″PGM.ASM″ into their project. The Learning Encoder main.asm file can be consulted for details on the configuration options.

Functional Description

This purpose of this application is to transmit a 22-bit programmable address and 2-bit key encode data to form a 24-bit code.

The main function of the SWIP incorporates 3 parts: Key Debounce, PGM Mode and TRANS Mode, which the program will freely jump between. After power on the program will first initialize the Time base, the function of which is to initiate a TMR interrupt every 332ms. When the program detects an external switch press, the program will enter the transmit mode or enter the program mode, determined by which keys are pressed. When it detects that it should enter the program mode, the time base will be altered to provide a 120μs timing value, after which the program detects a level change on the PGM pin, uses the length of the pulse to determine if the signal is ″1″ or a ″0″. Note here that a ″1″ or a ″0″ is not a simple measurement of the level of the pulse, but rather as shown below:

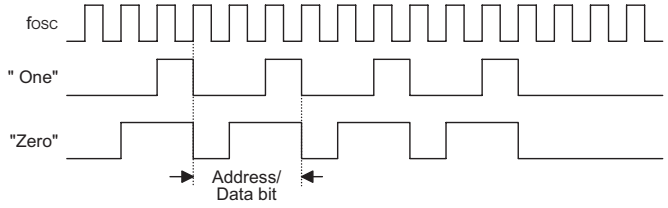


Fig. 1

After receiving all of the 22-bit address, the time base will return to its original value of 332μs, with the received address being placed into the EEPROM. In this way when power is removed the address information will still be retained. When power is re-applied the original programmed address can be again utilized to implement a code transmit operation. After entering the transmit function, the program will transmit the received address bits in the order of the lowest bit first and ending with the highest bit according to the protocol of Fig.1. The logic state of the external pins will then be transmitted in the order of the highest bit first and ending with the lowest bit, after which an ″anti-code″ period will be transmitted. This is shown in Fig.2

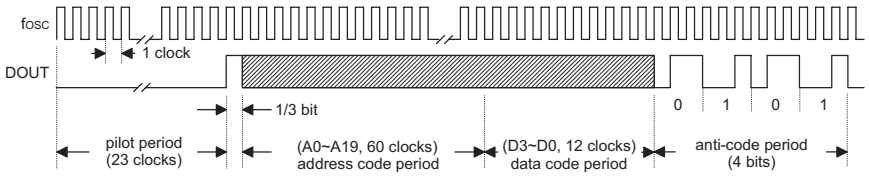


Fig. 2

Pin Function

HT48R06A-1

MCU	SWIP	Description
PA.0~PA.1	D0~D1	Data input/Transmission enable pins
PA.4	PGM	Program mode control pin, active low
PA.5	SIO	Programming address/control code input
PA.6	DOUT	Data serial output pin
PA.7	DISPLAY Pin	To Display program is over
PB.0	CS	Chip select pin (EEPROM)
PB.1	SK	Serial clock output pin for EEPROM
PB.2	DI/DO Pin	Data input and output pin for EEPROM

Instructions for Use

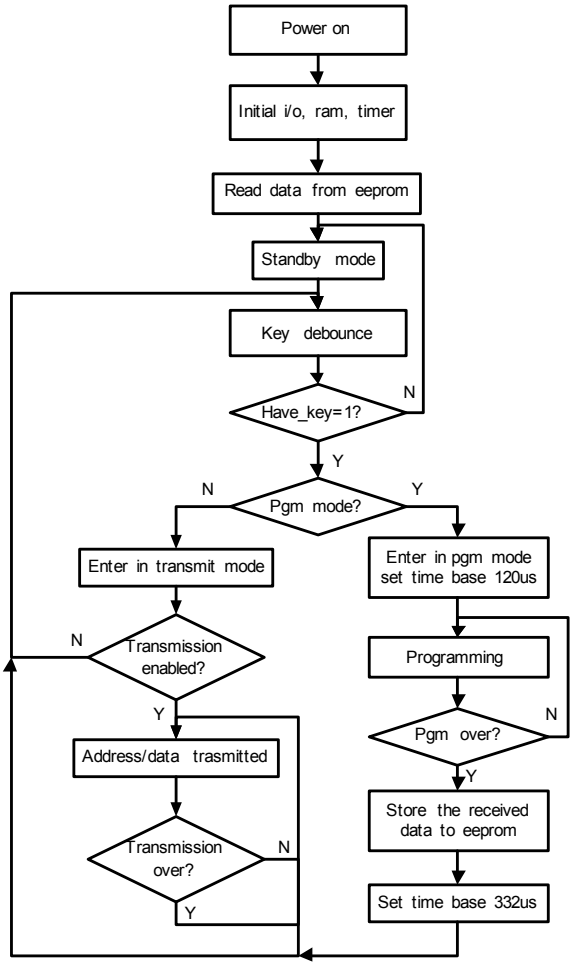
Software IP: SWIP_HT6P20B

The Learning Encoder (22+2) SWIP contains 2 internal SWIPs one known as transmit_mode and the other known as pgm_mode. As their names suggest, the transmit_mode section is used when the transmit mode is used and the pgm_mode section is used when the program mode is used. The following table shows an explanation of the two modes:

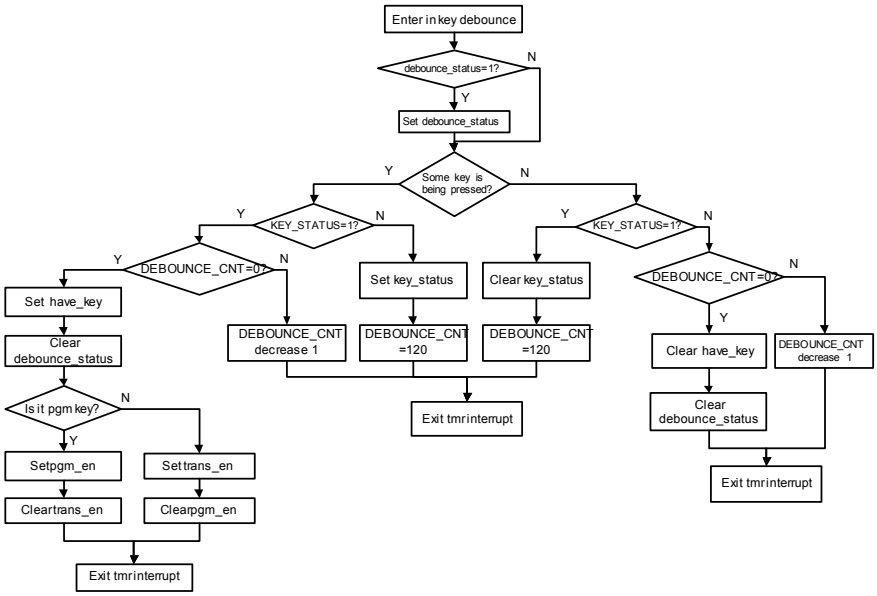
IP Name (Label)	System Resources	Function Descriptions
Transmit mode	Function	Encode and transmit 24 bits of information (2 status)
	MCU	HT48R06A-1
	ROM	119 words
	RAM	12 Bytes: ″trans_bit_cnt″, ″trans_add_cnt″, ″trans_data_cnt″, ″pilotcnt″, ″trans_buffer″, ″data_res″, ″add_buf0~add_buf02″, ″trans_add_buf0~trans_add_buf 2″ 7 Bits: ″trans_en″, ″transmitting″, ″trans_over″, ″transmitting_add″, ″transmitting_data_anticode″ ″oneaddbitover″, ″onedatabitover″
	Stack	1 level used
	Subroutine/Macro	Subroutine
	I/O Lines	1 I/O lines Pa.6: Data output pin (I/O), pull-high
	f_{SYS}	4MHz
	Other MCU Resources	TMR interrupt
	User Interface	1. User should include ″transmit_mode.asm″ in the project, and extern transmit_initial:near, transmit_mode:near 2. Call ″transmit_initial″ in the main program′s initialisation 3. Set time base 332μs, and enable TMR interrupt 4. Prepare add_buf0~prepare add_buf2, trans_add_buf0~trans_add_buf 2 and data_res 5. Set ″trans_en″ to transmit or clear it to be unable

IP Name (Label)	System Resources	Function Descriptions
PGM mode	Function	Receive and decode the encoder′s code (2 status)
	MCU	HT48R06A-1
	ROM	66 words
	RAM	8 bytes: ″hi_count″, ″lo_count″, ″hi_count_save″, ″lo_count_save″, ″bitcounter″, ″pgm_code0~ pgm_code2″ 3 bits: ″pgm_en″, ″pgm_ok″, ″sio_status″
	Stack	1 level used
	Subroutine/Macro	Subroutine
	I/O Lines	1 I/O lines Pa.5: Data input pin (I/O), pull-high
	f_{SYS}	4 MHz
	Other MCU Resources	TMR interrupt
	User Interface	1. User should include ″pgm.asm″ in the project, and extern pgm_initial:near, pgm_mode:near 2. Call ″pgm_initial″ in the main program′s initial 3. Set time base 120μs, and enable TMR interrupt 4. Set ″pgm_en″ to enable pgm_mode or clear it to be unable 5. The receive code is in the pgm_code0~pgm_code2

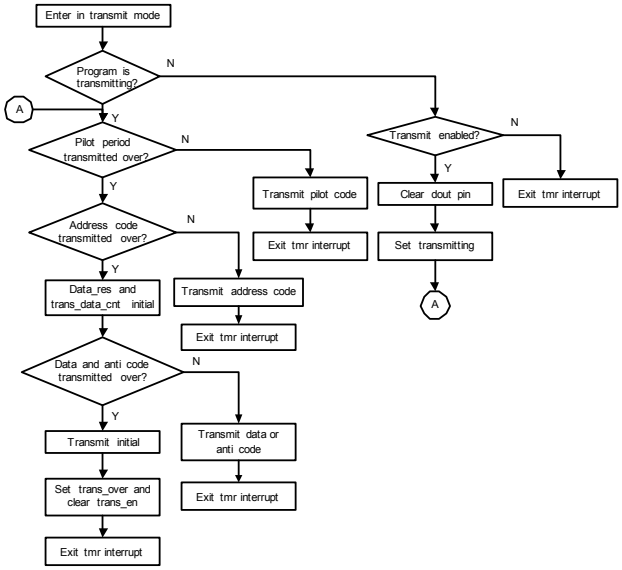
Program Flowchart



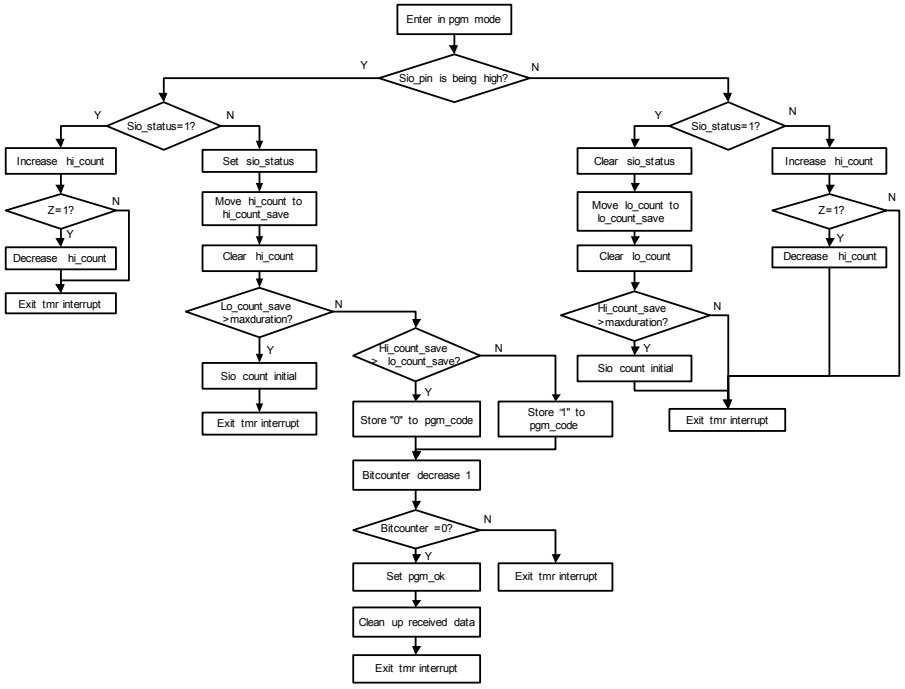
Main program Flowchart of Learning Encoder



Key Debounce Flow chart of Learning Encoder



Flowchart of transmit mode



Flowchart of pgm mode

Application Circuit

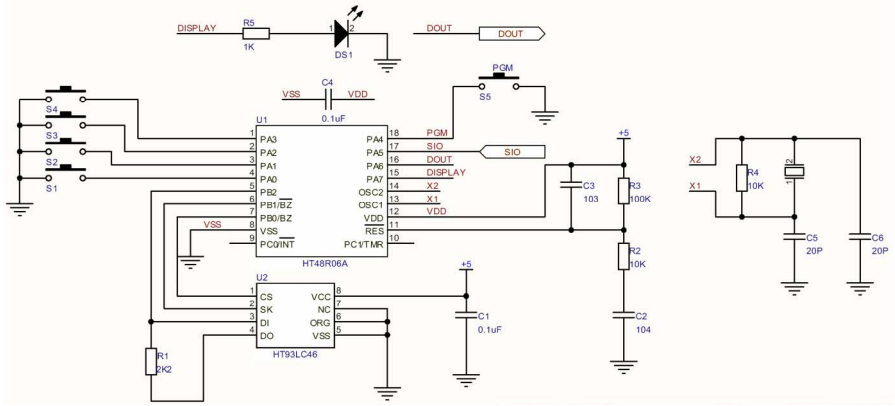


Fig.3 Learning Encoder (22add+2data) Demo Kit

Demo Kit User Instructions

Programmable

After power on, if the user wants to enter the program mode, the PGM switch should be pressed once. The LED will illuminate to indicate that the user can now execute a program operation. After the programming is complete the LED will switch off, indicating that a correct programming procedure has been conducted. When this happens the user can at any time press the DATA keys to transmit the correct code. Because the memory used is an EEPROM type, the user can at any time after power on, conduct a program operation. This program operation can be conducted for an unlimited number of times.

Transmit

After power on, the user can at any time press any of the two DATA keys to initiate a transmit code operation. The Demo Kit has a low power mode, which means if the MCU is not in the program mode or transmitting a code will enter a power down mode, ensuring that it consumes a minimum of standby current.