Product Overview

Address Spaces

Addressing Modes

Memory Map

SMCS51 Instruction Set

1 KS51840

OVERVIEW

KS51840, a 4-bit single-chip CMOS microcontroller, consists of the reliable SMCS-51 CPU core with on-chip ROM and RAM. Eight input pins and 11 output pins provide the flexibility for various I/O requirements. Auto reset circuit generates reset pulse every certain period, and every halt mode termination time. The KS51840 microcontroller has been designed for use in small system control applications that require a low-power, cost - sensitive design solution. In addition, the KS51840 has been optimized for remote control transmitter.

FEATURES

ROM Size

- 1,024 bytes

RAM Size

- 32 nibbles

Instruction Set

- 39 instructions

Instruction Cycle Time

- 13.2 μsec at fxx = 455 kHz

Input Ports

- Two 4-bit ports (24 pins)/one 4-bit port, five1-bit ports (20 pins)

Output Ports

- One 4-bit port, seven 1-bit ports (24 pins)/one 4-bit port, five1-bit ports (20 pins)

Built-in Oscillator

- Crystal/ceramic resonator

Built-in Reset Circuit

- Built-in Power-on reset and auto reset circuit for generating reset pulse every 13.1072/fxx (288 ms at fxx = 455 kHz)

Four Transmission Frequencies

- fxx/12 (1/4 duty), fxx/12 (1/3 duty), fxx/8 (1/2 duty), and no-carrier frequency

Supply Voltage

- 1.8 V-3.6 V (250 kHz ≤ f_{OSC} ≤ 3.9 MHz)
  2.2 V-3.6 V (3.9 MHz < f_{OSC} ≤ 6 MHz)

Power Consumption

- Halt mode: 1 μA (maxium)
- Normal mode: 0.5 mA (typical)

Operating temperature

- - 20 °C to 85 °C

Package Type

- 24 SOP, 20 DIP, 20 SOP

Oscillator Frequency divide select

- Mask Option = f_{OSC} or f_{OSC}/8

BLOCK DIAGRAM

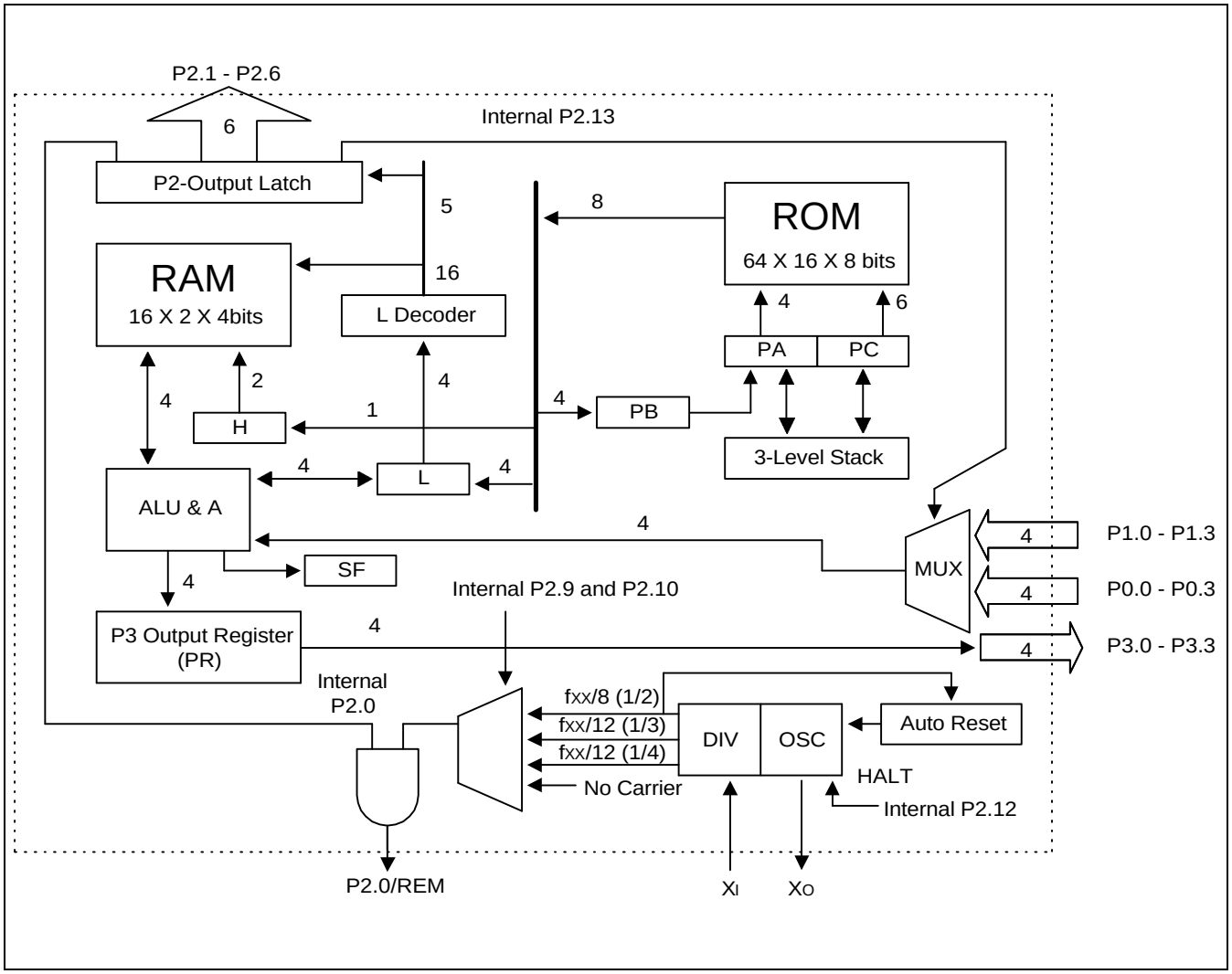


Figure 1-1. Block diagram

PIN CONFIGURATION (24 SOP)

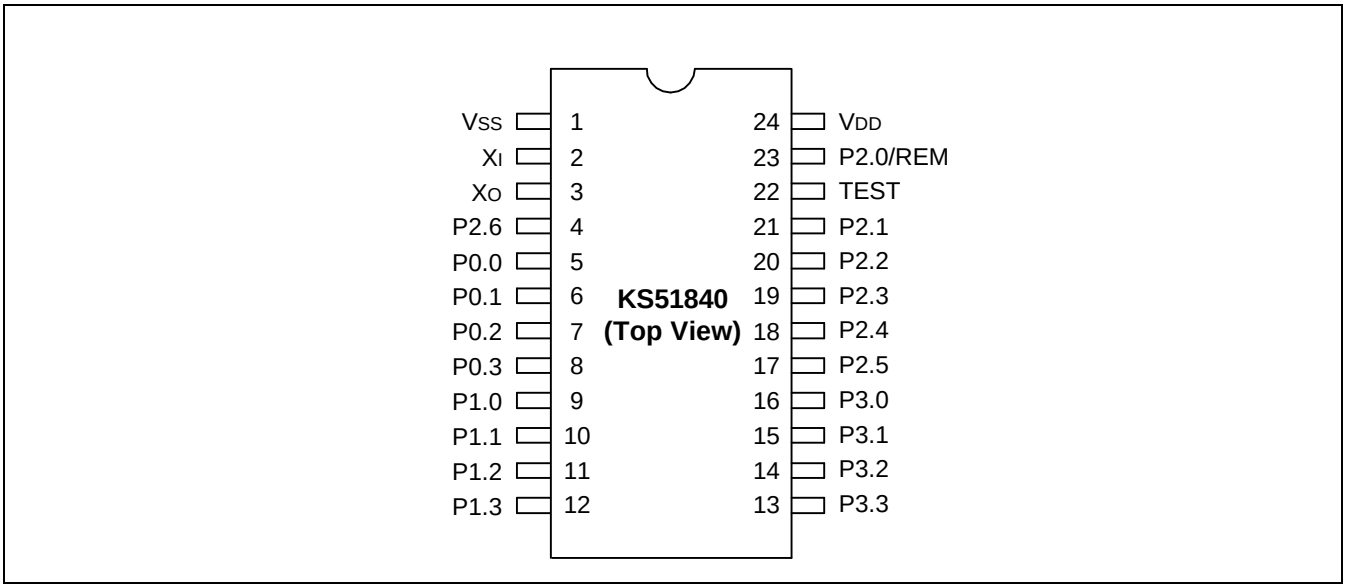


Figure 1-2. Pin Configuration (24 SOP)

Table 1-1. PIN Description for 24 PINS

Pin Name	Pin Number	Pin Type	Description	I/O Circuit Type
P0.0-P0.3	5, 6, 7, 8	Input	4-bit input port when P2.13 is low	A
P1.0-P1.3	9, 10, 11, 12	Input	4-bit input port when P2.13 is high	A
P2.0 REM	23	Output	1-bit individual output for remote carrier frequency (1)	B
P2.2-P2.5	20, 19, 18, 17	Output	1-bit individual output port	C
P2.1, P2.6	21, 4			D
P3.0-P3.3	16, 15, 14, 13	Output	4-bit parallel output port	C
TEST	22	Input	Input pin for test (Normally connected to V_{SS})	–
X_I	2	Input	Oscillation clock input	–
X_O	3	Output	Oscillation clock output	–
V_{DD}	24	–	Power supply	–
V_{SS}	1	–	Ground	–

NOTES:

1. The carrier can be selected by software as fxx/12 (1/3 duty), fxx/12 (1/4 duty), fxx/8 (1/2 duty), or no-carrier frequency.
2. Package type can be selected as 24 SOP in the ordering sheet.

PIN CONFIGURATION (20 DIP, 20 SOP)

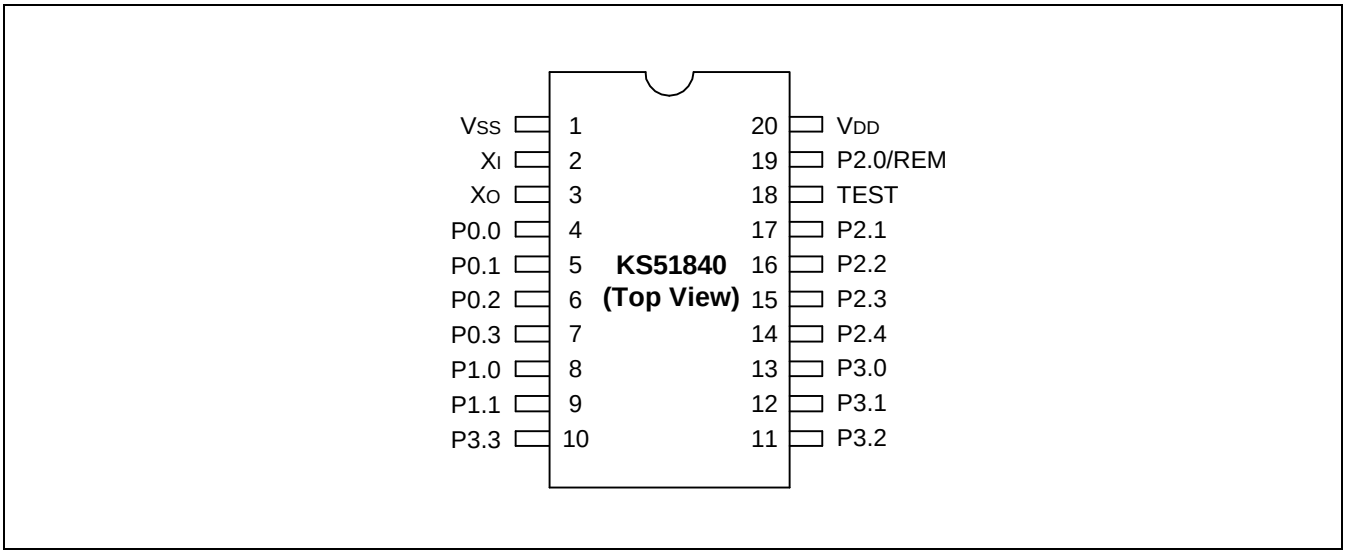


Figure 1-3. Pin Configuration (20 DIP, 20 SOP)

Table 1-2. Pin Description for 20 Pins

Pin Name	Pin Number	Pin Type	Description	I/O Circuit Type
P0.0-P0.3	4, 5, 6, 7	Input	4-bit input port when P2.13 is low	A
P1.0-P1.1	8, 9	Input	2-bit input port when P2.13 is high	A
P2.0/REM	19	Output	1-bit individual output for remote carrier frequency (1)	B
P2.2-P2.4	16, 15, 14	Output	1-bit individual output port	C
P2.1	17			D
P3.0-P3.3	13, 12, 11, 10	Output	4-bit parallel output port	C
TEST	18	Input	Input pin for test (Normally connected to V_{SS})	–
X_I	2	Input	Oscillation clock input	–
X_O	3	Output	Oscillation clock output	–
V_{DD}	20	–	Power supply	–
V_{SS}	1	–	Ground	–

NOTES:

1. The carrier can be selected by software as fxx/12 (1/3 duty), fxx/12 (1/4 duty), fxx/8 (1/2 duty), or no-carrier frequency.
2 Package type can be selected as 20 DIP, or 20 SOP in the ordering sheet.

I/O CIRCUIT SCHEMATICS

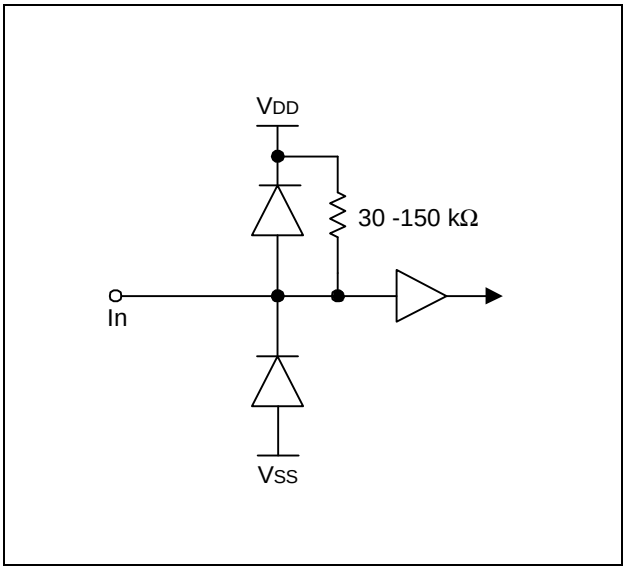
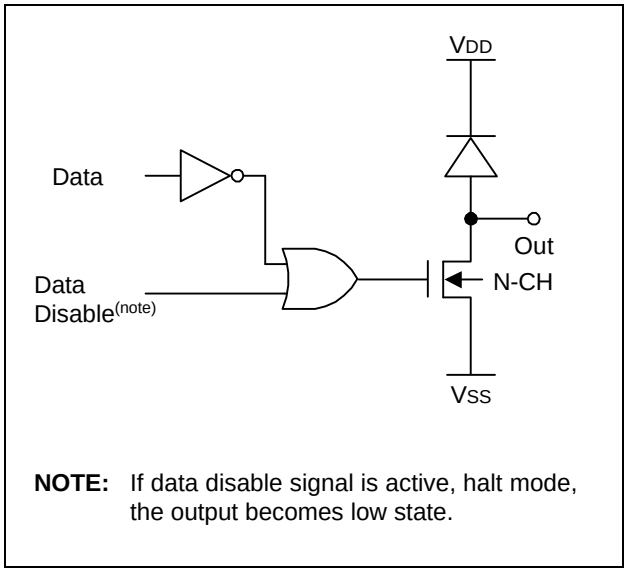
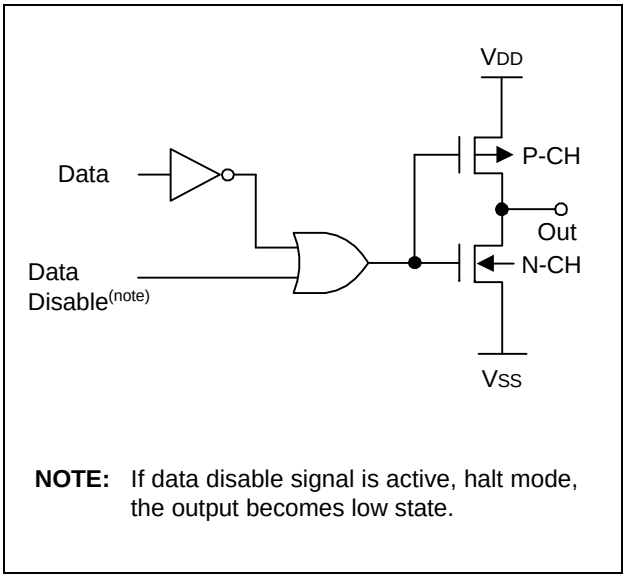


Figure 1-4. I/O Circuit Type A



NOTE: If data disable signal is active, halt mode, the output becomes low state.

Figure 1-6. I/O Circuit Type C



NOTE: If data disable signal is active, halt mode, the output becomes low state.

Figure 1-5. I/O Circuit Type B

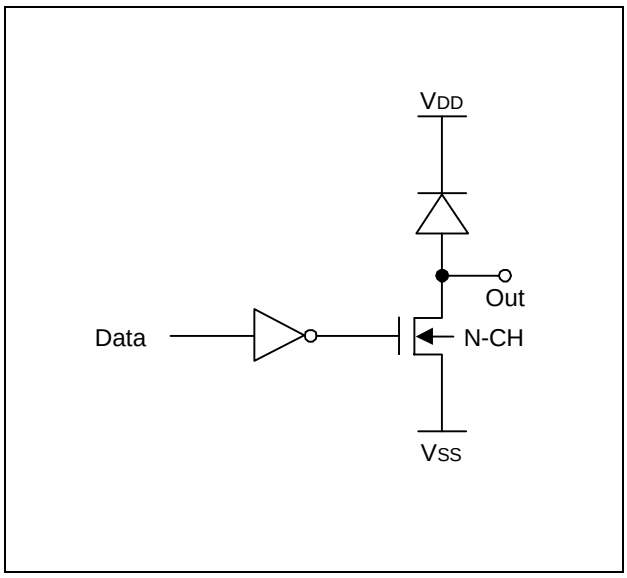


Figure 1-7. I/O Circuit Type D

Table 1-3. Absolute Maximum Ratings

Parameters	Symbols	Ratings	Units
Supply Voltage	V_{DD}	- 0.3 to 6	V
Input Voltage	V_I	- 0.3 to V_{DD} + 0.3	V
Output Voltage	V_O	- 0.3 to V_{DD} + 0.3	V
Soldering Temperature	T_{SLD}	260 (10 sec)	°C
Storage Temperature	T_{STG}	- 55 to 125	°C

Table 1-4. DC Characteristics (V_{DD} = 3 V, TA = 25°c)

Parameters		Symbols	Test Conditions	Min	Typ	Max	Units
Supply Voltage		V_{DD}	250kHz≤ f_{OSC}≤3.9MHz	1.8	3.0	3.6	V
			3.9MHz< f_{OSC} ≤6MHz	2.2	3.0	3.6	
Operating Temperature		T_A	–	-20	–	85	°C
High-Level Input Voltage		V_{IH1}	All input pins except X_{IN}	0.7 V_{DD}	–	V_{DD}	V
		V_{IH2}	X_{IN}	V_{DD}-0.3	–	V_{DD}	V
Low-Level Input Voltage		V_{IL1}	All input pins except X_{IN}	0	–	0.3 V_{DD}	V
		V_{IL2}	X_{IN}	0	–	0.3	V
High-Level Output Current P2.0		I_{OH1}	V_O = 2.0 V	-6.0	-9	-1.4	mA
Low-Level Output Current P2.0		I_{OL1}	V_O = 0.4 V	1.5	3.0	4.5	mA
Low-Level	P3 Output	I_{OL2}	V_O = 0.4 V	0.5	1.0	2.0	mA
Output	P2.1-P2.3			1.5	3.0	4.5	
Current	P2.4-P2.6			0.5	1.0	2.0	

Table 1-4. DC Characteristics (V_{DD} = 3 V, TA = 25°c) (Continued)

Parameters	Symbols	Test Conditions	Min	Typ	Max	Units
High-Level Input Leakage Current	I_{LIH1}	$V_I = V_{DD}$ All input pins except X_{IN}	–	–	3	uA
	I_{LIH2}	X_{IN}	–	3	10	
Low-level Input Leakage Current	I_{LIL1}	X_{IN}	-0.6	-3	-10	
High-level Output Leakage Current	I_{LOH}	$V_O = V_{DD}$ All output pins except P2.0	–	–	3	uA
Pull-up Resistance of Input Port	R	V_{DD} = 3 V V_I = 0 V	30	70	150	KΩ
Average Supply Current	I_{DD}	V_{DD} = 3 V Crystal/resonator Non-divide option f_{OSC} = 1 MHz Dvide-8 option f_{OSC} = 6 MHz	–	0.5	1.0	mA
HALT Current	I_{DDH}	f_{OSC} = 0	–	–	1.0	uA
Clock Frequency	fxx	Crystal/ceramic	250	–	1000	kHz
Oscillator Frequency	f_{OSC}	Crystal/ceramic Non-divide option	250	–	1000	
		Crystal/ceramic Divide-8 option	2000		6000	

FUCTIONAL DESCRIPTION

Program Memory (ROM)

The KS51840's program memory consists of a 1024-byte ROM, organized in 16 pages. Each page is 64 bytes long. (See Figure 1-8).

ROM addressing is supported by a 10-bit register made up of two sub-registers: a 4-bit Page Address register (PA), and a 6-bit Program Counter (PC).

Pages 0 through 15 (FH) can each access 64 (3FH) bytes.

ROM addressing occurs as follows: The 10-bit register selects one of the ROM's 1024-bytes. A new address is then loaded into the PC register during each instruction cycle.
Unless a transfer-of -control instruction such as JP,CALL, or RET is encountered, the PC is loaded with the next sequential 6-bit address in the page, PC + 1. In this case, the next address of 3FH would be 00H.

Only the PAGE instruction can change the Page Buffer (PB) to a specified value.

When a JP or CALL instruction is executed, and if the Status Flag is set to "1", the contents of the PB are loaded into the PA register. If the Status Flag is "0", however, the JP or CALL is executed like NOP instruction in an instruction cycle and the Status Flag is set to "1". After that, program execution proceeds.

Page-In Addressing

All instructions, including, JP and CALL, can be executed by page. (See Figure 1-9). When the Status Flag is "1", a JP or CALL causes a program to branch to its address (operand) in a page.

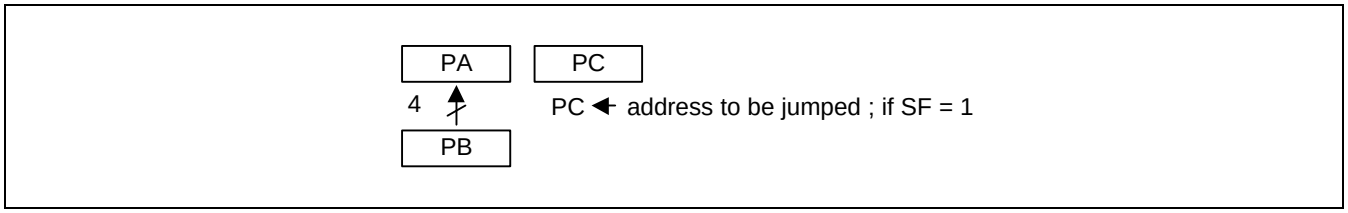


Figure 1-8. Page-In Addressing

Page-To-Page Addressing

When a PAGE instruction occurs, and if the Status Flag is "1", a JP or CALL instruction will cause a program to branch to its address (operand) across the page (See Figure 1-10).

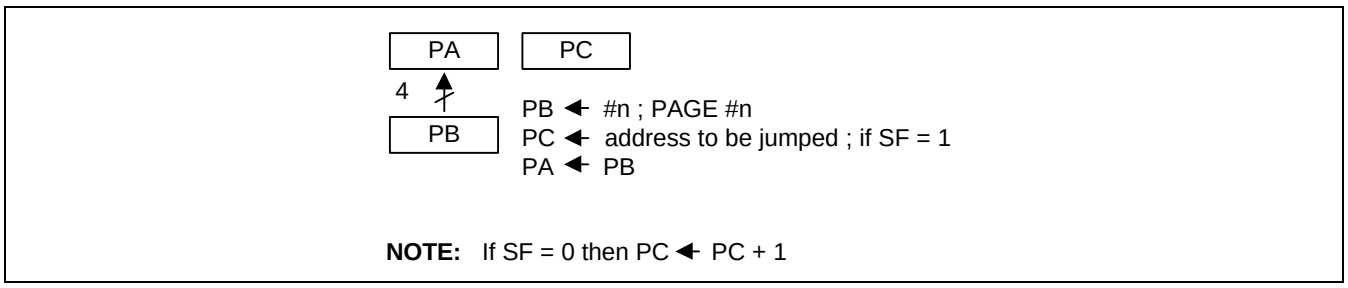
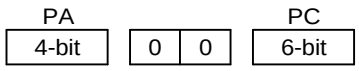


Figure 1-9. Page-to-Page Addressing



The 10-bit register points one of 1024 bytes at addresses 0000H to 0F3FH.
After reset, it points to 0FXXH for execution in the first instruction cycle. it then becomes 0F00H in th
next instruction cycle.

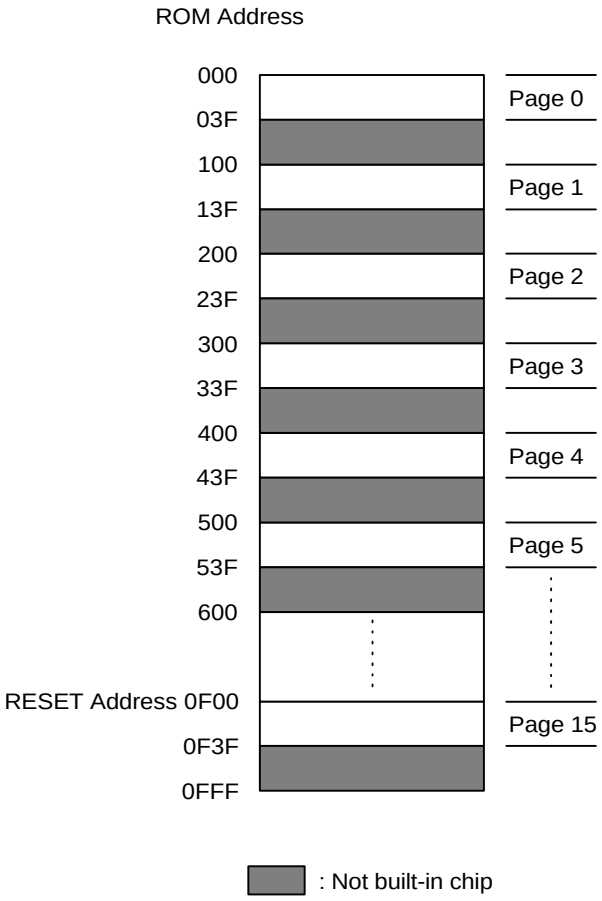


Figure 1-10. KS51840 Program Memory Map

DATA MEMORY (RAM)

The KS51840's data memory consists of a 32-nibble RAM which is organized into two files of 16 nibbles each (See Figure 1-12).
RAM addressing is implemented by a 7-bit register, HL.
It's upper 3-bit register (H) selects one of two files and its lower 4-bit register (L) selects one of 16 nibbles in the selected file.

Instructions which manipulate the H and L registers are as follow:

Select a file :

```
MOV     H,#n        ;  H ← #n, where n must be 0,4
NOT     H           ;  Complement MSB of H register
```

Select a nibble in a selected file :

```
MOV     L,A         ;  L ← A
MOV     L,A,@HL     ;  L ← M (H,L)
MOV     L,#N        ;  L ← #n, where 0 ≤ n ≤ 0FH
INCS    L           ;  L ← L + 1
DECS    L           ;  L ← L - 1
```

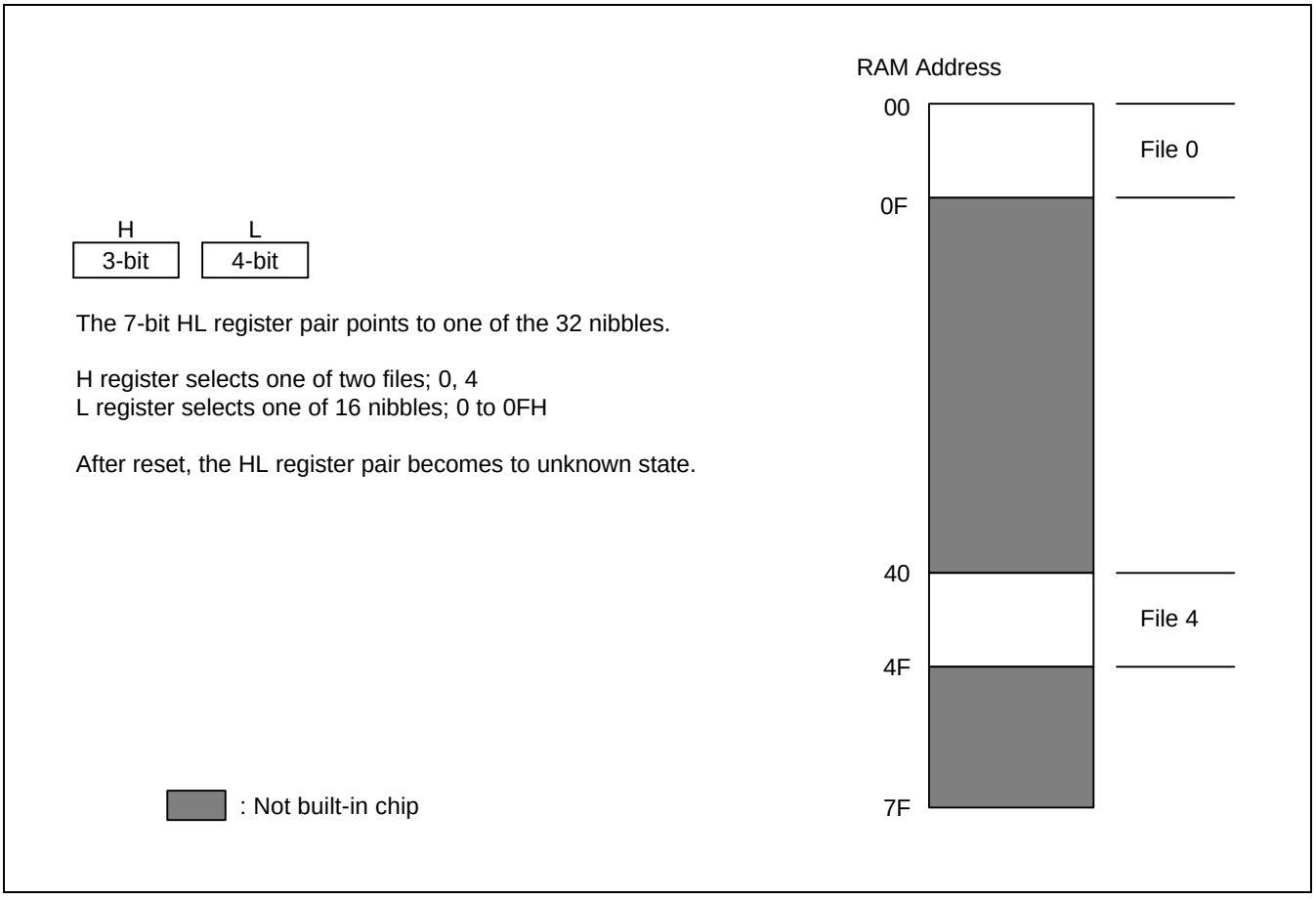


Figure 1-11. KS51840 Data Memory Map

REGISTER DESCRIPTIONS

Stack Register (SR)

Three levels of subroutine nesting are supported by a three-level stack as shown in Figure 1-13.

Each subroutine call (CALL) pushes the next PA and PC address into the stack. The latest stack to be stored will be overwritten and lost. Each return instruction (RET) pops the stack back into the PA and PC registers.

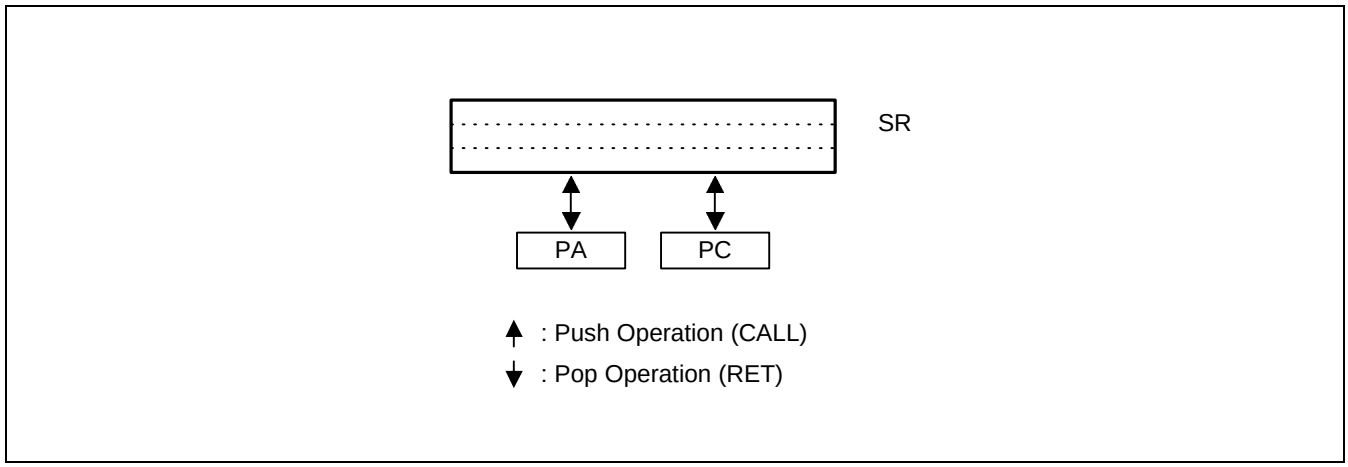


Figure 1-12. Stack Operations

Page Address Register (PA), Page Buffer Register (PB)

The Page Address Register (PA) and Page Buffer Register (PB) are 4-bit registers. The PA always specifies the current page.

A page select instruction (PAGE #n) loads the value “n” into the PB. When JP or CALL instruction is executed, and if the Status Flag (SF) is set to 1, the contents of PB are loaded into PA. If SF is “0”, however, the JP or CALL is executed like NOP instruction and SF is set to “1”. The contents of PB don’t be loaded. Figure 1-14 illustrates this concept.

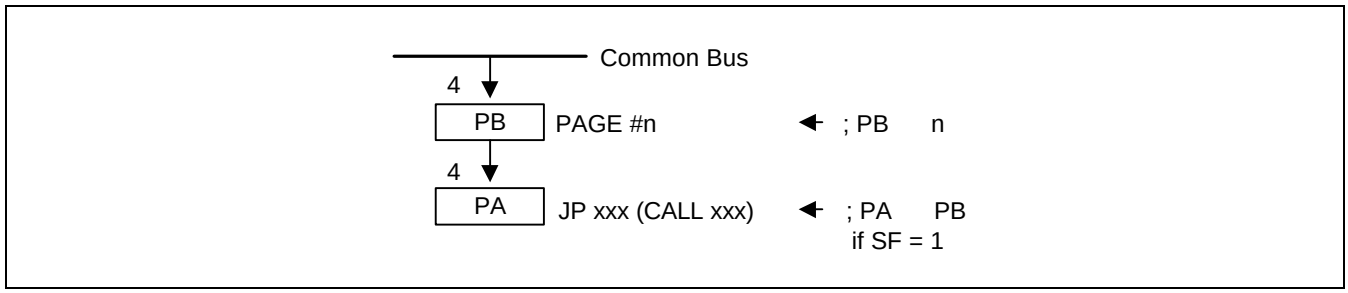


Figure 1-13. PA, PB Operations

Arithmetic Logic Unit (ALU), Accumulator (A)

The SMCS-51 CPU contains an ALU and its own 4-bit register (accumulator) which is the source and destination register for most I/O, arithmetic, logic, and data memory access operations.

Arithmetic functions and logical operations will set the status flag (SF) to “0” or “1”

Status Latch (SL)

The Status latch (SL) flag is an 1-bit flip-flop register. Only the “CPNE L,A” instruction can change the value of SL.

If the result of a “CPNE L,A” instruction is true, the SL is set to “1”; If not true, to “0”.

Status Flag : SF

The Status Flag (SF) is a 1-bit flip-flop register which enables programs to conditionally skip an instruction. All instructions, including JP and CALL, are executed when SF is “1”.

But if SF is “0”, the program executes NOP instruction instead of JP or CALL and resets SF to “1”. Then, program execution proceeds. The following instructions set the SF to “0”:

- **Arithmetic Instructions**

```
ADDS    A, #n       ;  if no carry
ADDS    A,@HL       ;  if no carry
INCS    A,2HL       ;  if no carry
INCS    A           ;  if no carry
INCS    L           ;  if no carry
SUBS    A,@HL       ;  if no carry
DECS    A,@HL       ;  if no carry
DECS    A           ;  if no carry
DECS    L           ;  if no carry
```

- **Compare Instructions**

```
CPNE    @HL,A       ;  if M(H,L) = (A)
CPNZ    @HL         ;  if M(H,L) = 0
CPNE    L,#n        ;  if (L) = #
CPNE    L,A         ;  if (L) = (A)
CPNE    A,@HL       ;  if (A) > M(H,L)
CPNZ    P0          ;  if (P0) = 0
CPBT    @HL,b       ;  if M (H,L,b) ≠ 1
```

- **Data Transfer Instructions**

```
MOV     @HL+,A      ;  if no carry
MOV     @HL_,A      ;  if borrow
```

- **Logical Instructions**

```
NOTI    A           ;  if (A) ≠ 0 after operation
```

INPUT PORTS : P0, P1

The P0 and P1 input ports have internal pull-up 30-150 kΩ resistors, (See I/O circuit type A), each multiplexed to a common bus (See Figure 1-15). If the P2.13 pin is programmed to low, then port 0 is selected as the input port. Otherwise, if the P2.13 pin high, port 1 is selected.

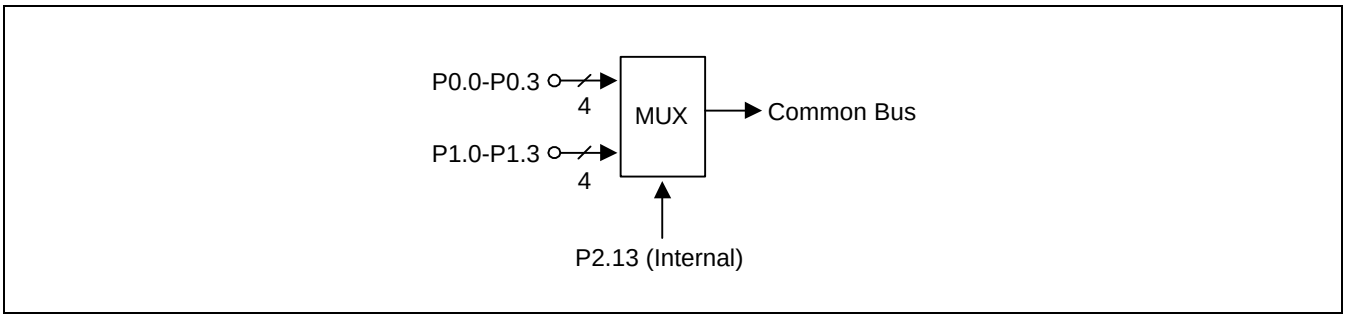


Figure 1-14. KS51840 Input Port

OUTPUT PORTS : P2, P3

The P2 and P3 output ports can be configured as push-pull (P2.0/REM only) and open drain (P2.1-P2.6, P3.0-P3.3) as follows:

- Standard push-pull : A COMS push-pull buffer with N-channel transistor to ground in conjunction with a P-channel transistor to V_{DD}, compatible with COMS and TTL. (See I/O Circuit Type B).
- N-channel open drain : An N-channel transistor to ground, compatible with CMOS and TTL. (see I/O Circuit Type C and D).

P2.0, P2.2-P2.5 and P3.0-P3.3 pins become low state in halt mode.
The L register specifies P2 output pins (P2.0/REM-P2.6, P2.9-P2.10, P2.12, and P2.13) individually as follows:

- SETB P2.(L) : Set port 2 bits to correspond to L-register contents.
- CLRB P2.(L) : Clear port 2 bits to correspond to L-register contents.

P3 output pins P3.0-P3.3 are parallel output pins.

For the KS51840, only the 4-bit accumulator outputs its value to the P3 port by the output instruction “OUT P3, @SL+ A” (the value of the Status Latch (SL) does not matter).

TRANSMISSION CARRIER FREQUENCY

One of four carrier frequencies can be selected and transmitted through the P2.0/REM pin by programming the internal P2.9, P2.10 and P2.0 pins (See Table 1-5). Figure 1-16 shows a simplified diagram of the various transmission circuits.

Table 1-5. Carrier Frequency Selection Table

P2.10	P2.9	Carrier Frequency of P2.0/REM Pin
0	0	fxx/12, 1/3 duty
0	1	fxx/8, 1/2 duty
1	0	fxx/12, 1/4 duty
1	1	No carrier

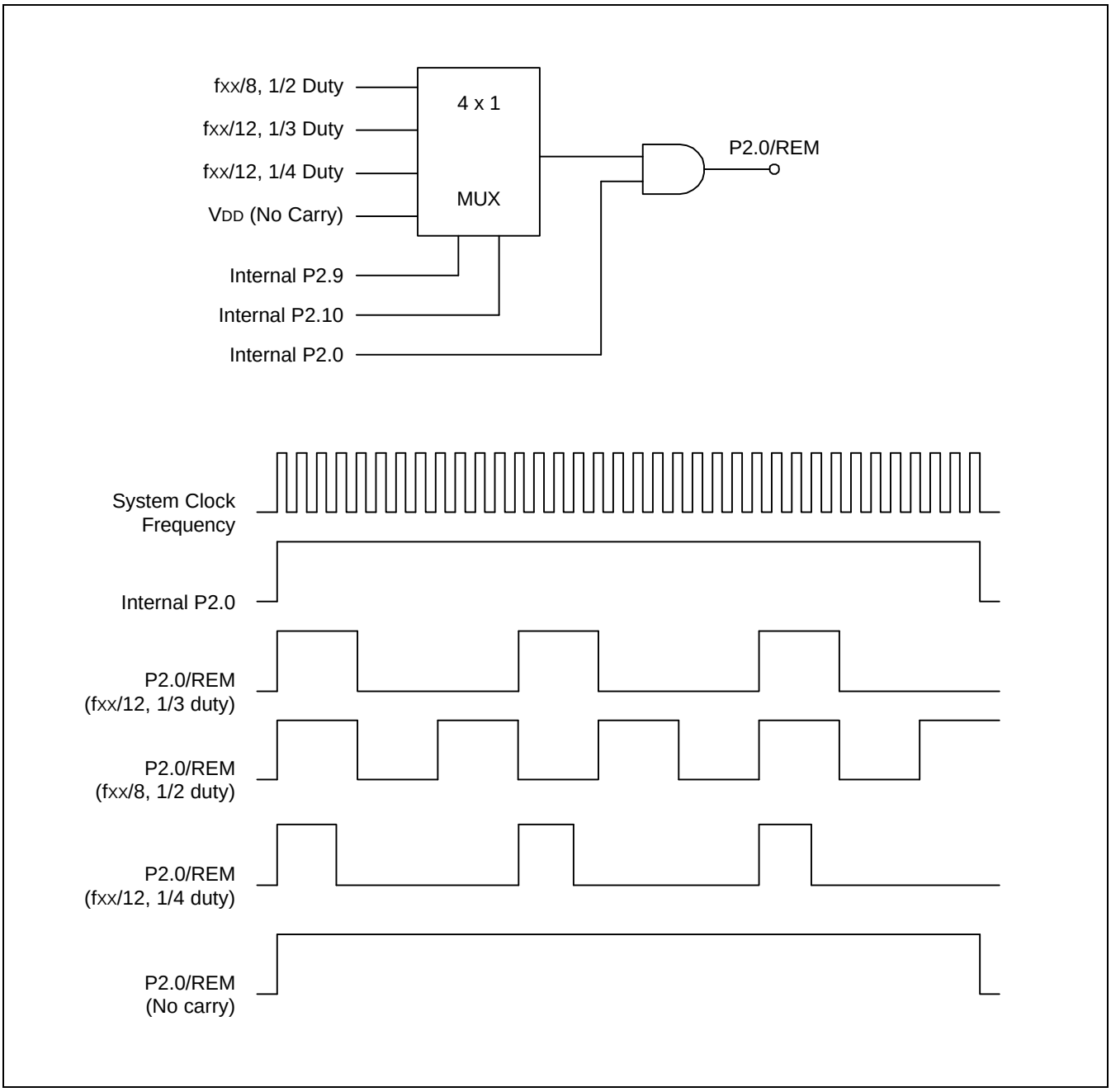


Figure 1-15. Diagram of Transmission Circuits

HALT MODE

The HALT mode is used to reduce power consumption by stopping the clock and holding the states of all internal operations fixed. This mode is very useful in battery-powered instruments. It also holds the controller in wait status for external stimulus to start some event. The KS51840 can be halted by programming the P2.12 pin high, and by forcing P0 input pins (P0.0-P0.3) to high and P1 input pins (P1.0-P1.3) to high, concurrently (See Figure 1-17). When in HALT mode, the internal circuitry does not receive any clock signal, and all P2, P3 output pins become low states. However, P2.1 and P2.6 pins retain their programmed values until the device is re-started as follows:

- Forcing any P0 and P1 input pins to low : system reset occurs and it continues to operate from the reset address.

An oscillation stabilization time of 13 msec in fxx = 455 kHz crystal oscillation is needed for stability (See Figure 1-18).

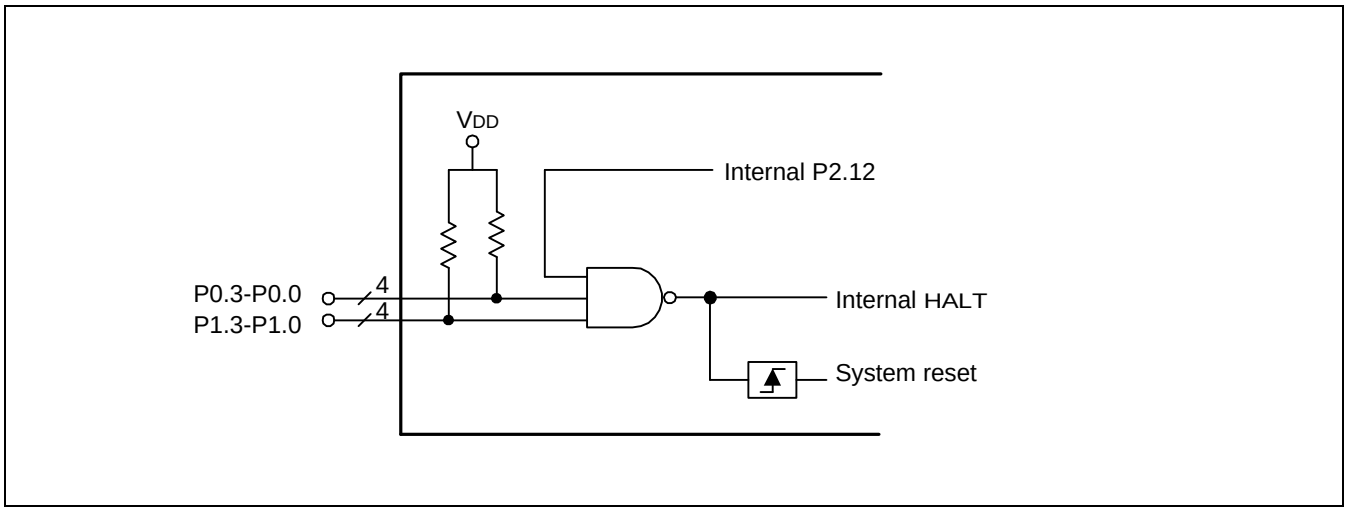


Figure 1-16. Block Diagram of HALT Logic

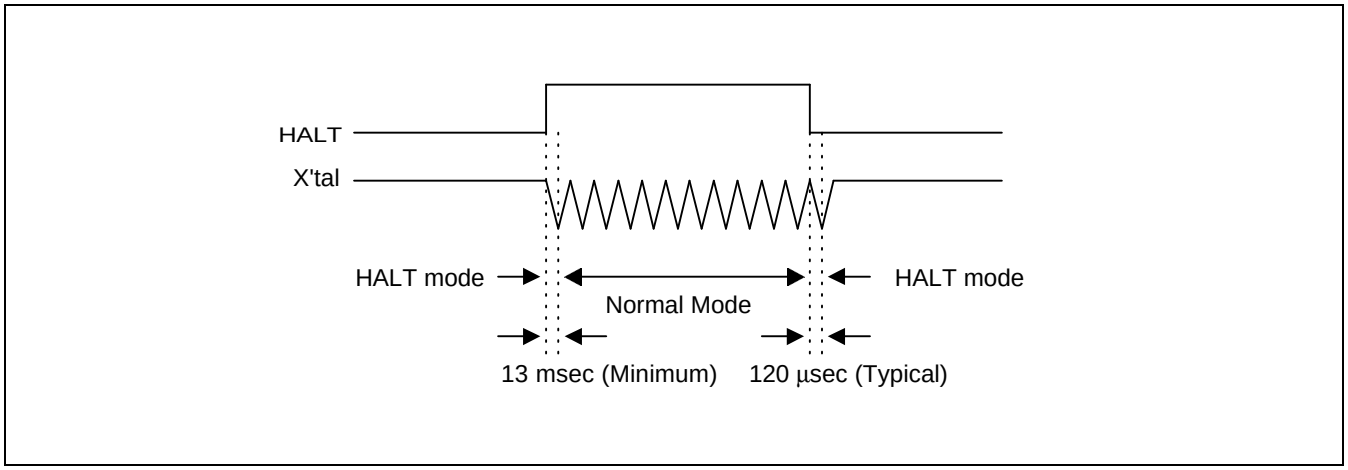


Figure 1-17. Release Timing for HALT or RESET to Normal Mode in Crystal Oscillation

RESET

All reset operations are internal in the KS51840. It has an internal power-on reset circuit consisting of a 7 pF capacitor and a 1 MΩ resistor (See Figure 1-19). The controller also contains an auto-reset circuit that resets the chip every 131,072 oscillator clock cycles (288 ms at a fxx = 455 kHz clock frequency). The auto-reset counter is cleared by the rising edge of a internal P2.0 pin, by HALT, or by the power-on reset pulse (See Figure 1-20). Therefore, no clocks are sent to the counter and the time-out is suspended in HALT mode. When a reset occurs during program execution, a transient condition occurs. The PA register is immediately initialized to 0FH. The PC, however, is not reset to 0H until one instruction cycle later. For example, if PC is 1AH when a reset pulse is generated, the instruction at 0F1AH is executed, followed by the instruction at 0F00H.

After a reset, approximately 13 msec is needed before program execution proceeds (assuming fxx = 455 kHz ceramic oscillation).

Upon initialization, registers are set as follows:

- PC register to 0 in next instruction cycle
- PA and PB registers to 0FH (15th page)
- SF and SL registers to 1
- HL registers to unknown state
- All internal/external output pins (P3.0-P3.3, P2.0/REM-P2.6, P2.9, P2.10, P2.12 and P2.13) to low.

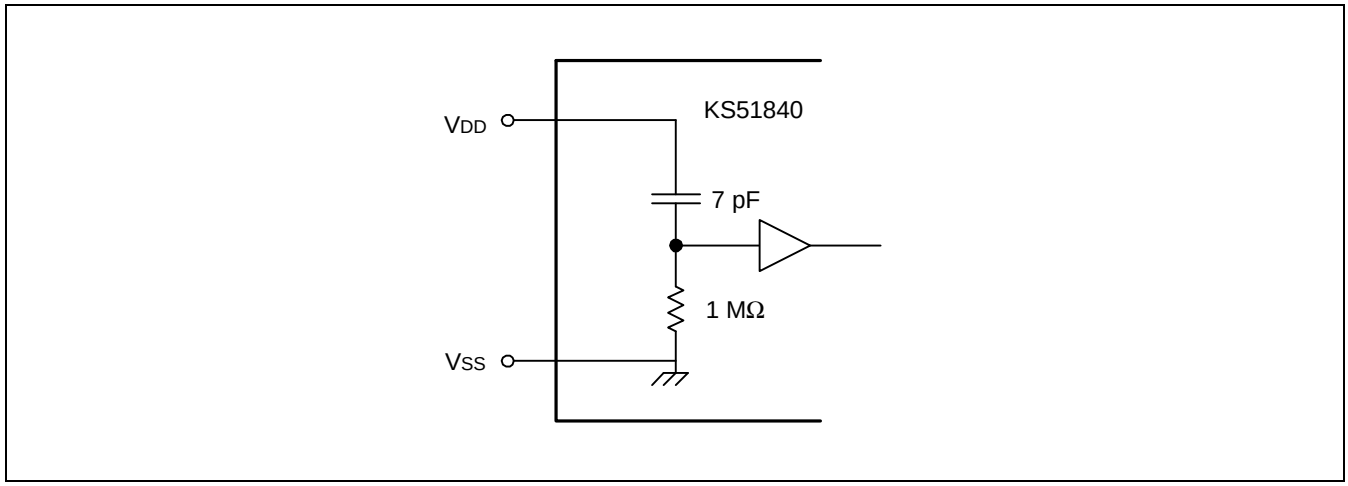
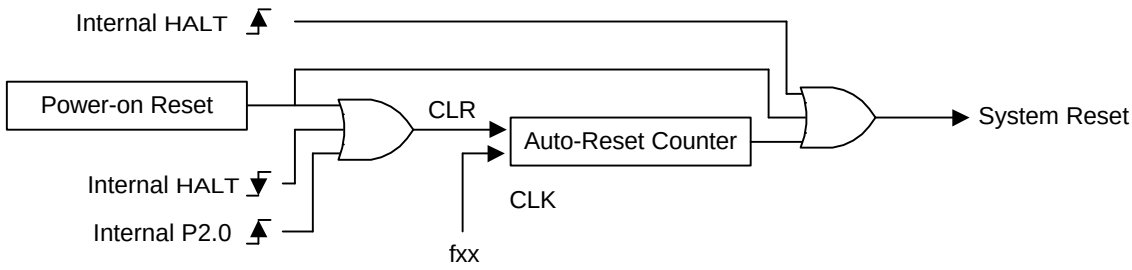


Figure 1-18. KS51840's Power-on Reset Circuit



The Auto-Reset Counter is cleared every 131,072/fxx (288 msec if fxx is 455 KHz).

Figure 1-19. Auto Reset Block Diagram

OSC DIVIDE OPTION CIRCUIT

The OSC Divide Option Circuit provides a maximum 1MHz fxx system clock. f_{OSC} which is generated in oscillation circuit is divided eight or non-divide in this circuit to produce fxx. This dividing ratio will be chosen by mask option. (See Figure 1-21)

f_{OSC} : Oscillator clock

fxx : System clock (f_{OSC} or f_{OSC} /8)

f_{CPU}: CPU clock (f_{CPU} = fxx/6)

1 instruction cycle clock

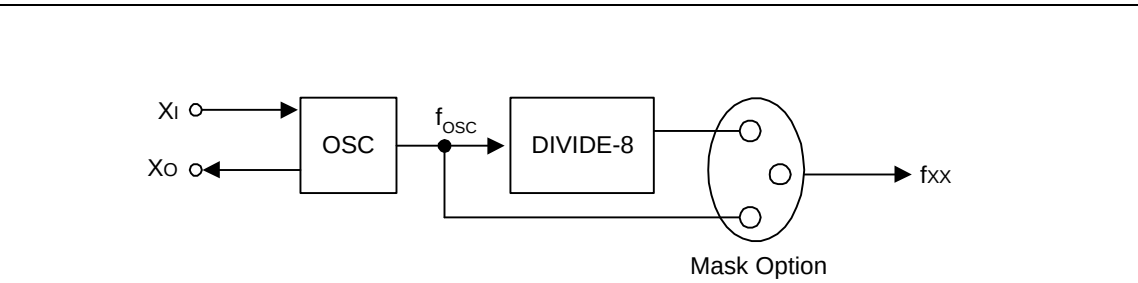
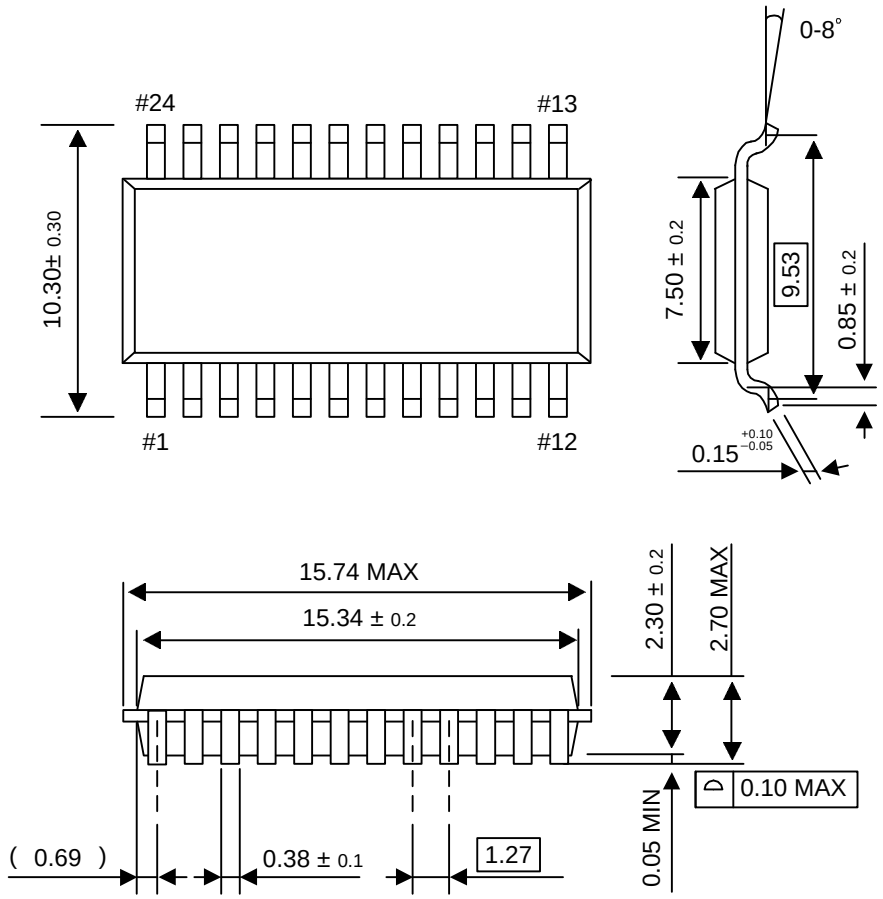


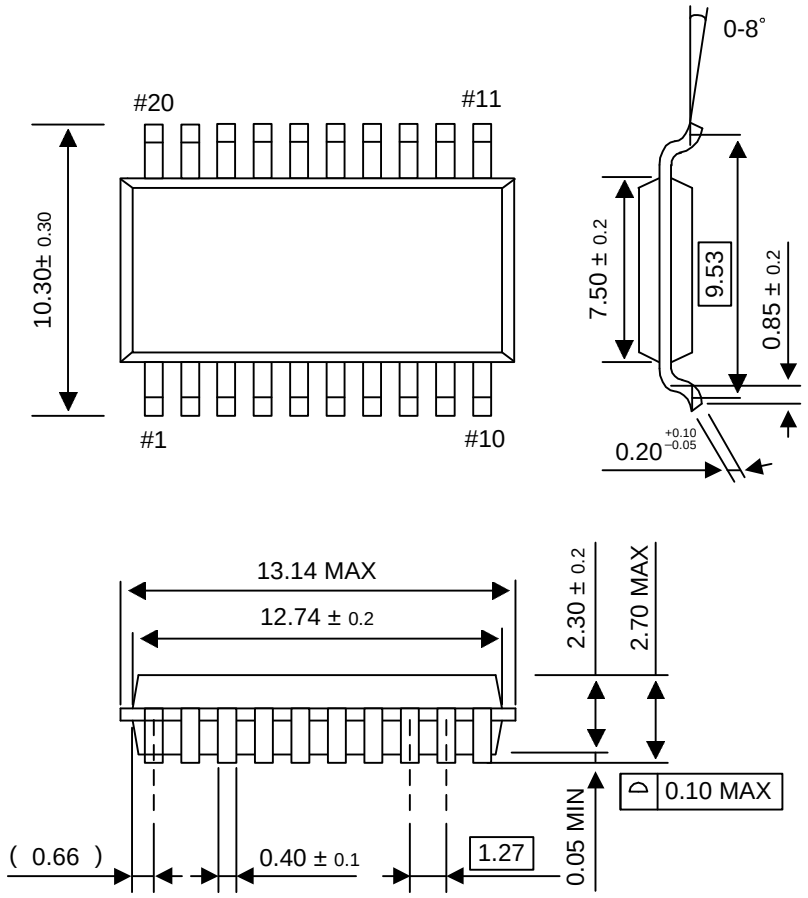
Figure 1-20. KS51840 OSC Divide Option Circuit

PACKAGE DIMENSIONS



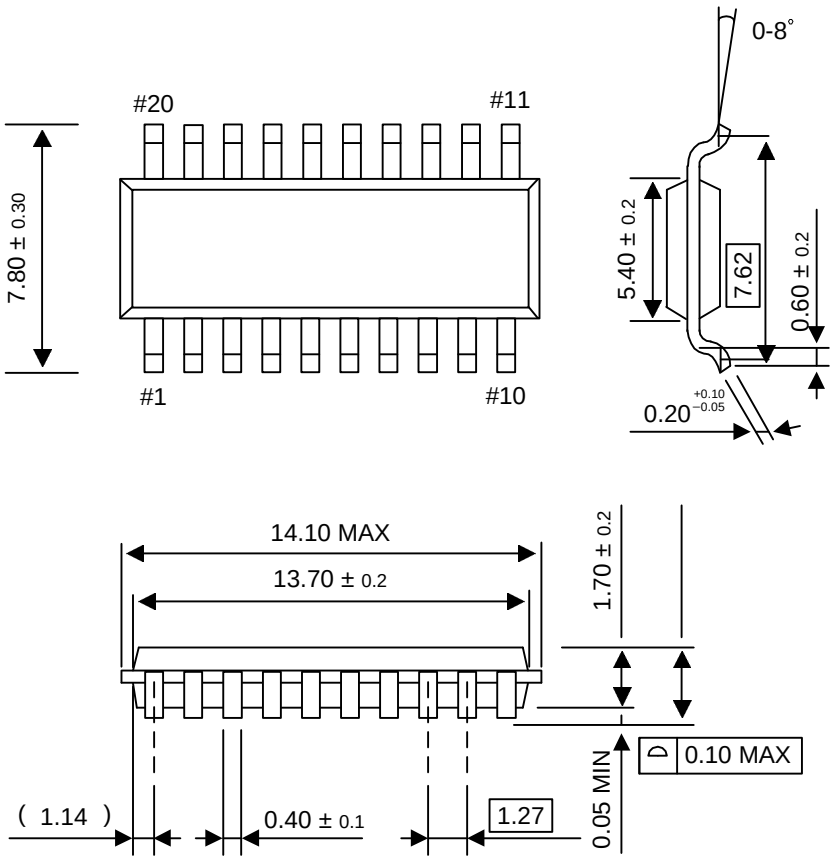
NOTE: Dimensions are in millimeters.

Figure 1-21. 24 SOP-375



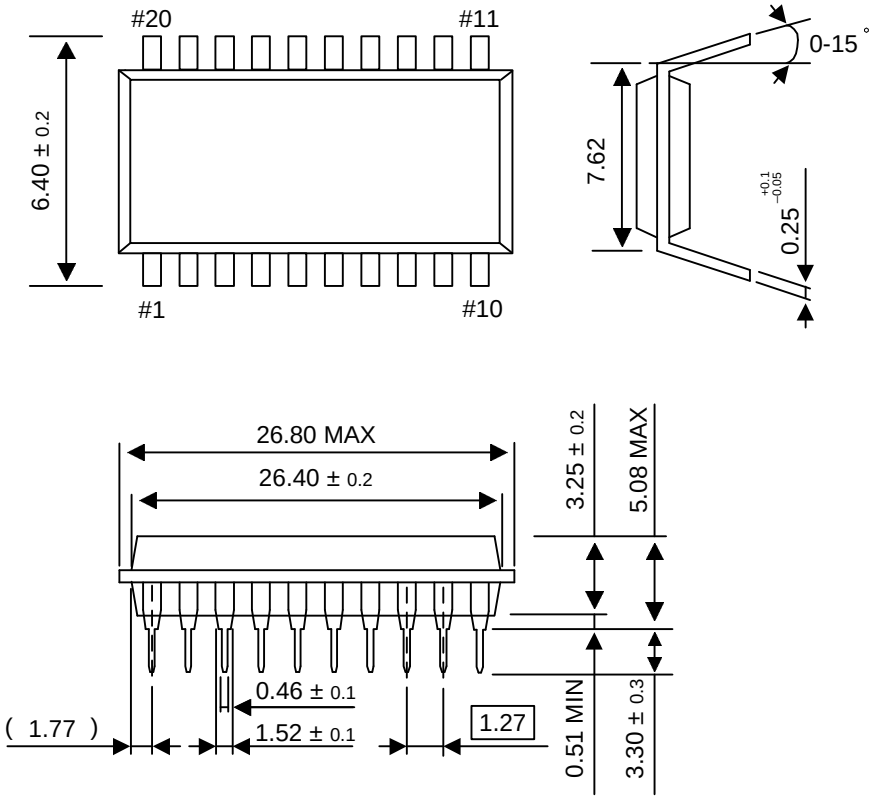
NOTE: Dimensions are in millimeters.

Figure 1-22. 20 SOP-375



NOTE: Dimensions are in millimeters.

Figure 1-23. 20 SOP-300



NOTE: Dimensions are in millimeters.

Figure 1-24. 20 DIP-300A

3 INSTRUCTION SET

OVERVIEW

Abbreviations and symbols table specifies internal architecture, instruction operand and operational symbols.

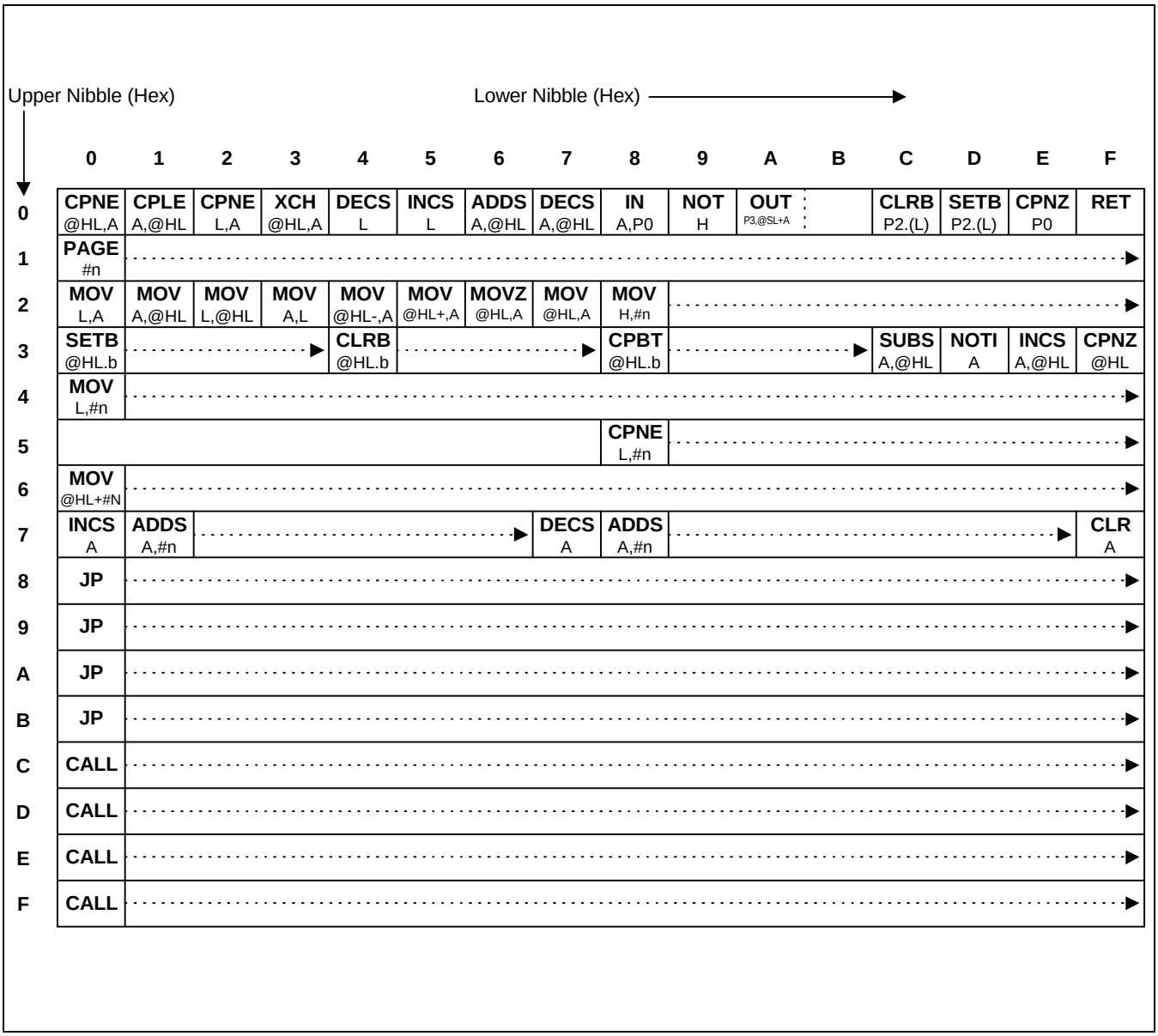
As mentioned before, JP and CALL instructions are executed normally only when SF is high. If SF is low, the program executes NOP instruction instead of them and sets SF to high. And then, the program executes a next instruction. In addition, JPL and CALL are long jump and long call instructions which consists of PAGE and JP/CALL instructions.

Table 3-1. Abbreviations and Symbols

Symbol	Description	Symbol	Description
L	L register (4 bits)	SF	Status Flag
A	Accumulator (4 bits)	P3	P3-output
(L)	The contents of the L register	P0	P0 input (4 bits)
(A)	The contents of the accumulator	D	Any binary number
SL	Status latch (1 bit)	DST	Destination operand
PB	Page buffer register (4 bits)	C	Carry Flag
PA	Page address register (4bits)	SRC	Source operand
P2	P2-output	REG	Register
PC	Program counter	←	Transfer
SR	Stack register	+	Addition or increment by 1
H	H register	≤	Equal or less than
M	RAM addressed by H and L registers	( )	The complement of the contents
(H)	The contents of the H register	@	Indirect register address prefix
M (H,L)	The contents of the RAM addressed by H,L	#n	Constant n (immediate 3or 4-bit data)
b	Bit address of the RAM [(H,L)] addressed by H,L	↔	Is exchanged with
≠	Not equal to	–	Subtract or decrement by 1

Table 3-2. Instruction Set Summary

Mnemonic	Operand	Description
MOV Instructions		
MOV	L,A	Move A to register L
MOV	A,L	Move L register to A
MOV	@HL,A	Move A to indirect data memory
MOV	A,@HL	Move indirect data memory to A
MOV	L,@HL	Move indirect data memory to register L
MOV	@HL+,A	Move A to indirect data memory and increment register L
MOV	@HL-,A	Move A to indirect data memory and decrement register L
MOV	L,#n	Move immediate data to register L
MOV	H,#n	Move immediate data to register H
MOV	@HL+,#n	Move immediate data to indirect data memory and increment register L
MOVZ	@HL,A	Move A to indirect data memory and clear A
XCH	@HL,A	Exchange A with indirect data memory
PAGE	#n	Set PB register to n
Program Control Instructions		
CPNE	@HL,A	Compare A to indirect data memory and set SF if not equal
CPNZ	@HL	Set SF if indirect data memory
CPNE	L,A	Compare A to register L, set SF and SL if not equal
CPNE	L,#n	Compare immediate data to register L and set SF if not equal
CPLE	A,@HL	Set SF if A is less than or equal to indirect data memory
CPNZ	P0	Set SF if A is less than or equal to indirect data memory
CPBT	@HL,b	Test indirect data memory bit and set SF if indirect bit is one
JP	dst	Jump if SF flag is set
CALL	dst	Call subroutine if SF is set
RET		Return from subroutine
I/O Instructions		
SETB	P2.(L)	Set bit
CLRB	P2.(L)	Clear bit
IN	A,P0	Input P0 to A
OUT	P3,@SL+A	Output A to P3-PLA output port
Logical Instructions		
NOTI	A	Complement A and increment A
NOT	H	Complement MSB of H register
CLR	A	Clear
Arithmetic Instructions		
ADDS	A,@HL	Add indirect data memory to A
ADDS	A,#n	Add immediate data to A
SUBS	A,@HL	Subtract A from indirect data memory
INCS	A,@HL	Increment indirect data memory and load the result in A
INCS	L	Increment register L
INCS	A	Increment A
DECS	A	Decrement A
DECS	A,@HL	Decrement indirect data memory and load the result in A
DECS	L	Decrement register L
Bit Manipulation Instruction		
SETB	@HL.b	Set indirect data memory bit
CLRB	@HL.b	Clear indirect data memory bit



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	CPNE @HL,A	CPLE A,@HL	CPNE L,A	XCH @HL,A	DECS L	INCS L	ADDS A,@HL	DECS A,@HL	IN A,P0	NOT H	OUT P3,@SL+A		CLRB P2.(L)	SETB P2.(L)	CPNZ P0	RET
1	PAGE #n															
2	MOV L,A	MOV A,@HL	MOV L,@HL	MOV A,L	MOV @HL-,A	MOV @HL+,A	MOVZ @HL,A	MOV @HL,A	MOV H,#n							
3	SETB @HL.b				CLRB @HL.b				CPBT @HL.b				SUBS A,@HL	NOTI A	INCS A,@HL	CPNZ @HL
4	MOV L,#n															
5									CPNE L,#n							
6	MOV @HL+#N															
7	INCS A	ADDS A,#n						DECS A	ADDS A,#n							CLR A
8	JP															
9	JP															
A	JP															
B	JP															
C	CALL															
D	CALL															
E	CALL															
F	CALL															

Figure3-1. KS51 Opcode Map

MOV L,A

Binary Code:	0 0 1 0	0 0 0 0

Description: The contents of the accumulator are moved to register L.
The contents of the source operand are not affected.

Operation: (L) ← (A)

Flags: SF : Set to one
SL : Unaffected

Example:

```
CLR     A       ; Clear the contents of A
MOV     L,A     ; Move 0H to REG L
```

MOV A,L

Binary Code:	0 0 1 0	0 0 1 1

Description: The contents of register L are moved to the accumulator.
The contents of the source operand are not affected.

Operation: (A) ← (L)

Flags: SF : Set to one
SL : Unaffected

Example:

```
MOV     L,#3H   ; Move 3H to REG L
MOV     A,L     ; Move 0H to A
```

MOV @HL,A

Binary Code:	0 0 1 0	0 1 1 1

Description: The contents of the accumulator are moved to the data memory whose address is specified by registers H and L.
The contents of the source operand are not affected.

Operation: M [(H,L)] ← (A)

Flags: SF : Set to one
SL : Unaffected

Example:

```
CLR     A       ; Clear the contents of A
MOV     H,#0H   ; Move 0H to REG H
MOV     L,#3H   ; Move 3H to REG L
MOV     @HL,A   ; Move 0H to RAM address 03H
```

MOV A,@HL

Binary Code:	0 0 1 0 | 0 0 0 1
Description:	The contents of the data memory addressed by registers H and L are moved to accumulator. The contents of the source operand are not affected.
Operation:	(A) ← M [(H,L)]
Flags:	SF : Set to one SL : Unaffected

Example:

```
                        Assume HL contains 04H
MOV     A,@HL           ; Move contents of RAM addressed 04H to A
```

MOV L,@HL

Binary Code:	0 0 1 0 | 0 0 1 0
Description:	The contents of the data memory addressed by registers H and L are moved to register L. The contents of the source operand are not affected.
Operation:	(L) ← M [(H,L)]
Flags:	SF : Set to one SL : Unaffected

Example:

```
                        Assume Hl contains 04H
MOV     L,@HL           ; Move contents of RAM address 4H to REG L
CPNE    L,#5H           ; Compare 5H to REG L values
JP      XX              ; jump to XX if REG L value is not 5H
JP      YY              ; Jump to YY if REG L value is 5H
```

MOV @HL+,A

Binary Code:	0 0 1 0 | 0 1 0 1
Description:	The contents of the accumulator are moved to the data memory addressed by registers H,L; L register contents are incremented by one. The contents of the source operand are not affected.
Operation:	M [(H,L)] ← (A), L ← L + 1
Flags:	SF : Set if carry occurs; cleared otherwise SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#0FH
CLR     A
MOV     @HL+A           ; Move 0H to RAM address 0FH and increment REG L value
                        by one
JP      PRT             ; jump to PRT, since there is a carry from increment
```

MOV @HL-A

Binary Code:	0 0 1 0	0 1 0 0

Description: The contents of accumulator are moved to the data memory addressed by registers H,L; L register contents are decremented by one.
The contents of the source operand are not affected.

Operation: M [(H,L)] ← (A), L ← L - 1

Flags: SF : Set if no borrow; cleared otherwise
SL : Unaffected

Example:
```
MOV     H,#0H
MOV     L,#3H
CLR     A
MOV     @HL-,A
JP      ABC
```

MOV L,#N

Binary Code:	0 1 0 0	d d d d

Description: The 4-bit value specified by n (data) is loaded into register L.
The contents of the source operand are not affected.

Operation: (L) ← #n

Flags: SF : Set to one
SL : Unaffected

Example:
```
MOV     L,#8H           ; 8H is moved to REG L
```

MOV H,#n

Binary Code:	0 0 1 0	1 d d d

Description: The 3-bit value specified by n (data) is moved to register H.
The contents of the source operand are not affected.

Operation: (H) ← #n

Flags: SF : Set to one
SL : Unaffected

Example:
```
MOV     H,#4H           ; 4H is moved into REG H
```

MOV @HL+,#n

Binary Code:	0 1 1 0	d d d d

Description: The 4-bit value specified by n (data) is moved to data memory addressed by registers H,L; L register contents are incremented by one.
The contents of the source operand are not affected.

Operation: M [(H,L)] ← #n, L ← L + 1

Flags: SF : Set to one
SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#7H
MOV     @HL+,#9H    ; Move 9H to RAM address 07H and increment REG L value
                    by one, then REG L contains 8H
```

MOVZ @HL,A

Binary Code:	0 0 1 0	0 1 1 0

Description: The contents of the accumulator are moved to the data memory addressed by registers H,L;
accumulator contents are cleared to zero.

Operation: M [(H,L)] ← (A), (A) ← 0

Flags: SF : Set to one
SL : Unaffected

Example:

```
MOV     L,#3H
MOV     A,L
MOVZ    @HL,A       ; Move 3H to indirect RAM and clear A to zero
MOV     L,A         ; Move 0H to REG L
SETB    P2.(L)      ; Set P2.0 to 1
```

XCH @HL,A

Binary Code:	0 0 0 0	0 0 1 1

Description: This instruction exchanges the contents of the data memory addressed by registers H and L with the accumulator contents.

Operation: M [(H,L)] ↔ (A)

Flags: SF : Set to one
SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#6H
CLR     A           ; Clear A to zero
ADDS    A,#5H       ; Add 5H to A
XCH     @HL,A       ; Exchange 5H with  contents of RAM address 06H
```

PAGE #n

Binary Code:	0 0 0 1 | d d d d		
Description:	The immediate 4-bit value specified by n (data) is loaded into the PB register.		
Operation:	(PB) ← #n		
Flags:	SF : Set to one SL : Unaffected		
Example:	PAGE	#3H	; Move 3H to page buffer
	JP	AN	; Jump to label AN located at page 3 if SF is one; otherwise, it is skipped

CPNE @HL,A

Binary Code:	0 0 0 0 | 0 0 0 0		
Description:	The contents of accumulator are compared to the contents of indirect data memory; an appropriate flag is set if their values are not equal. The contents of both operands are unaffected by the comparison.		
Operation:	M [(H,L)] ≠ (A)		
Flags:	SF : Set if not equal, cleared otherwise SL : Unaffected		
Example:	CLR	A	
	ADDS	A,#3H	
	MOV	H,#0H	
	MOV	L,#6H	
	CPNE	@HL,A	; Acc value 3H is compared to contents of RAM address 06H
	JP	OA	; Jump to OA if values of RAM address 06H are not 3h
	JP	OB	; Jump to OB if values of RAM address 06H are 3H

CPNZ @HL

Binary Code: | 0 0 1 1 | 1 1 1 1 |

Description: This instruction compares the magnitude of indirect data memory with zero, and the appropriate flag is set if their values are not equal, i.e., if the contents of indirect data memory are not zero.
The contents of operand are unaffected by the comparison.

Operation: M [(H,L)] ≠ 0

Flags: SF : Set if not zero, cleared otherwise
SL : Unaffected

Example:

```
                        Assume the contents of RAM address are 4H
CPNZ    @HL             ; Compare 4H with zero
JP      EQ              ; Jump to EQ because the result is not equal
JP      WAIT
```

CPNE L,A

Binary Code: | 0 0 0 0 | 0 0 1 0 |

Description: The contents of the accumulator are compared to the contents of register L; the appropriate flags are set if their values are not equal.
The contents of both operands are unaffected by the comparison.

Operation: (L) ≠ (A)

Flags: SF : Set if not equal, cleared otherwise
SL : Set if not equal, cleared otherwise

Example:

```
                        Assume REG L contains 5H, A contains 4H
CPNE    L,A             ; Compare A to REG L values
JP      K1              ; Jump to K1 because the result is not equal
JP      K2
```

CPNE L,#n

Binary Code:	0 1 0 1	d d d d

Description: This instruction compare the immediate 4 bit data n with the contents of register L, and sets an appropriate flag if their values are not equal.
The contents of both operands are unaffected by the comparison.

Operation: (L) ≠ #n

Flags: SF : Set if not equal, cleared otherwise
SL : Unaffected

Example:

```
CLR     A
ADDS    A,#4H
MOV     L,A
CPNE    L,#5H       ; Compare immediate data 5H to REG L values
JP      K3          ; Jump to K3 because the result is not equal
```

CPNE A,@HL

Binary Code:	0 0 0 0	0 0 0 1

Description: The contents of indirect data memory are compared to the contents of the accumulator. Appropriate flags are set if the contents of the accumulator are less than or equal to the contents of indirect data memory.
The contents of both operands are unaffected by the comparison.

Operation: (A) ≤ M [(H,L)]

Flags: SF : Set if less than or equal to, cleared otherwise
SL : Unaffected

Example:

```
                            Assume RAM address holds 8H
CPLE    A,@HL       ; Compare 8H to A values
JP      MAR         ; Jump to MAR if 0H ≤ A ≤ 8H
JP      BPR         ; Jump to BPR if 9H ≤ A ≤ 0FH
```

CPNZ P0

Binary Code: | 0 0 0 0 | 1 1 1 0 |

Description: The instruction compares the contents of Port 0 with zero. Appropriate flags are set if their values are not equal, i.e., if the contents of Port 0 are not zero.
The contents of the operand are unaffected by the comparison.

Operation: (P0) ≠ 0

Flags: SF : Set if not zero, cleared otherwise
SL : Unaffected

Example:

```
MOV     L,#0DH
CLRB    P2.(L)      ; Clear P2.13, i.e., select P0 input
CPNZ    P0          ; Compare P0 to zero
JP      KEYIN       ; Jump to KEYIN if P0 ≠ 0
JP      NOKEY       ; Jump to NOKEY if P0 = 0
```

CPBT @HL,b

Binary Code: | 0 0 1 1 | 1 0 d d |

Description: CPBT tests indirect data memory bit and sets appropriate flags if the bit value is one.
The contents of operand are unaffected by the test.

Operation: M [(H,L)] = 1

Flags: SF : Set if one, cleared otherwise
SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#0BH
CPBT    @HL,3       ; Test RAM address 0BH bit 3
JP      Q1          ; Jump to Q1 if RAM address bit 3 is 1
JP      Q2          ; Jump to Q2 if RAM address bit 3 is 0
```

JP dst

Binary Code:	1 0 d d	d d d d
Description:	The JP transfers program control to the destination address if the SF is one. The conditional jump replaces the contents of the program counter with the address indicated and transfers control to that location. Had the SF flag not been set, control would have proceeded with the next instruction.	
Operation:	If SF = 1 ; PC ← (W), PA ← PB	
Flags:	SF : Set to one SL : Unaffected	
Example:	JP SUTIN1	; This instruction will cause program execution to branch to the instruction at label SUTIN; SUTIN1 must be within the current page

---

CALL dst

Binary Code:	1 1 d d	d d d d
Description:	If the SF flag is set to 1, this instruction calls a subroutine located at the indicated address, and then pushes the current contents of the program counter to the top of the stack. The program counter value used is the address of the first instruction following the CALL ins. The specified destination address is then loaded into the program counter and points to the first instruction of a procedure. At the end of the procedure, the return (RET) instruction can be used to return to the original program flow.	
Operation:	If SF = 1 ; SRi ← PC + 1, PSRi ← PA PC ← I (W), PA ← PB	
Flags:	SF : Set to one SL : Unaffected	
Example:	CALL ACD1	; CALL subroutine located at the label ACD1 where ACD1 must be within the current page

---

RET

Binary Code:	0 0 0 0	1 1 1 1
Description:	This instruction is normally used to return to the previously executing procedure at the end of a procedure entered by a CALL instruction. The contents of the location addressed by the stack pointer are popped into the program counter. The next statement executed is that addressed by the new contents of the program counter.	
Operation:	PC ← Sri, PB ← PSRi PA ← PB	
Flags:	SF : Set to one SL : Unaffected	
Example:	RET	; Return from subroutine

SETB P2.(L)

Binary Code:	0 0 0 0	1 1 0 1

Description: This instruction sets the Port 2 bit addressed by register L without affecting any other bits in the destination.

Operation: P2.(L) ← 1

Flags: SF : Set to one
SL : Unaffected

Example:
```
MOV     L,#0H
SETB    P2.(L)      ; Set P2.0 to 1
```

CLRB P2.(L)

Binary Code:	0 0 0 0	1 1 0 0

Description: This instruction clears the Port 2 bit addressed by register L without affecting any other bits in the destination.

Operation: P2.(L) ← 0

Flags: SF : Set to one
SL : Unaffected

Example:
```
MOV     L,#0H
CLRB    P2.(L)      ; Clear P2.0 to 0
```

IN A,P0

Binary Code:	0 0 0 0	1 0 0 0

Description: Data present on Port n is transferred (read) to the accumulator.

Operation: (A) ← (Pn) (n = 0,1)

Flags: SF : Set to one
SL : Unaffected

Example:
```
IN      A,P0        ; Input port 0 data to Acc
MOV     L,A
CPNE    L,#3H
JP      OX          ; Jump to OX  if port 0 data ≠ 3H
JP      QP          ; Jump to QP if port 0 data = 3H
```

OUT P3,@SL+A

Binary Code:	0 0 0 0	1 0 1 0

Description: The contents of the accumulator and SL are transferred to the P3 Output register.

Operation: (P3 Output register) ← (A) + (SL)

Flags: SF : Set to one
SL : Unaffected

Example:

```
CLR     A
OUT     P3,@SL+A        ; Zero output on port 3
```

NOTI A

Binary Code:	0 0 1 1	1 1 0 1

Description: The contents of the accumulator are complemented; all 1 bits are changed to 0, and vice-versa, and then incremented by one.

Operation: (A) ← (A), (A) ← (A) +1

Flags: SF : Set if the result is zero, cleared otherwise
SL : Unaffected

Example:

```
CLR     A
ADDS    A,#7H
NOTI    A       ; Complement 7H (0111B) and increment the result by one;
                ; the instruction NOTI A then leaves 9H (1001B) in A
```

NOT H

Binary Code:	0 0 0 0 | 1 0 0 1
Description:	The MSB of register H is complemented,
Operation:	(H) ← (H)
Flags:	SF : Set to one SL : Unaffected

Example:

```
MOV     H,#4H
NOT     H       ; Complement 4H (100B), then it leaves 00H (000B) in REG H
```

CLR A

Binary Code:	0 1 1 1 | 1 1 1 1
Description:	The contents of the accumulator are cleared to zero (all bits set on zero).
Operation:	(A) ← 0
Flags:	SF : Set to one SL : Unaffected

Example:

```
CLR     A       ; A value are cleared to zero
```

ADDS A,@HL

Binary Code:	0 0 0 0 | 0 1 1 0
Description:	ADDS adds the contents of indirect data memory to accumulator, leaving the result in the accumulator. The contents of the source operand are unaffected.
Operation:	(A) ← M [(H,L)] + (A)
Flags:	SF : Set if a carry occurred, cleared otherwise SL : Unaffected

Example:

```
                        Assume RAM address holds 5H
CLR     A       ; Clear A to zero
ADDS    A,@HL   ; This instruction will leaves 5H in A
```

ADDS A,#n

Binary Code:	0 1 1 1 | d d d d		
Description:	The specified 4-bit data n is added to the accumulator and the sum is stored in the accumulator.		
Operation:	(A) ← (A) + #n		
Flags:	SF : Set if a carry occurred, cleared otherwise SL : Unaffected		
Example:	CLR	A	; Clear A to zero
	ADDS	A,#4H	; Add 4H to A, it leaves 4H in A

SUBS A,@HL

Binary Code:	0 0 1 1 | 1 1 0 0		
Description:	SUBS subtracts the contents of accumulator from the contents of indirect data memory, leaving the result in the accumulator. The contents of source operand are unaffected.		
Operation:	(A) ← M [(H,L)] - (A)		
Flags:	SF : Set if no borrow occurred, cleared otherwise SL : Unaffected		
Example:			Assume RAM address holds 0CH
	MOV	L,#8H	
	MOV	A,L	
	SUBS	A,@HL	; Subtract A from 0CH; it will leave 4H in A

INCS A,@HL

Binary Code:	0 0 1 1 | 1 1 1 0		
Description:	The contents of indirect data memory are incremented by one and the result is loaded into the accumulator. The contents of indirect data memory are unaffected.		
Operation:	(A) ← M [(H,L)] + 1		
Flags:	SF : Set if a carry occurred, cleared otherwise SL : Unaffected		
Example:			Assume RAM address holds 6H
	CLR	A	; Clear A to zero
	INCS	A,@HL	; Increment 6H by one and leave 7H in A

INCS L

Binary Code:	0 0 0 0	0 1 0 1
Description:	The contents of the L register are incremented by one.	
Operation:	(L) ← (L) + 1	
Flags:	SF : Set if a carry occurred, cleared otherwise SL : Unaffected	
Example:	MOV L,#5H	
	INCS L	; Increment REG L value 5H by one

INCS A

Binary Code:	0 1 1 1	0 0 0 0
Description:	The contents of the accumulator are incremented by one.	
Operation:	(A) ← (A) + 1	
Flags:	SF : Set if no borrow occurred, cleared otherwise SL : Unaffected	
Example:	MOV L,#5H	
	MOV A,L	
	INCS A	; Increment 5H by one

DECS A

Binary Code:	0 1 1 1	0 1 1 1
Description:	The contents of the accumulator are decremented by one.	
Operation:	(A) ← (A) - 1	
Flags:	SF : Set if a carry occurred, cleared otherwise SL : Unaffected	
Example:	MOV L,#0BH	
	MOV A,L	
	DECS A	; The instruction leaves the value 0AH in A

DECS A,@HL

Binary Code:	0 0 0 0	0 1 1 1

Description: The contents of the data memory addressed by the H and L registers are decremented by one and the result is loaded in the accumulator.
But the contents of data memory are not affected.

Operation: (A) ← M [(H,L)] - 1

Flags: SF : Set if a carry occurred, cleared otherwise
SL : Unaffected

Example:

```
                        Assume RAM address holds 5h
MOV     L,#0AH
MOV     A,L
DECS    A,@HL           ; Decrement the value 5H by one, and the result value 4H is
                          loaded in A
```

DECS L

Binary Code:	0 0 0 0	0 1 0 0

Description: The contents of the L register are decremented by one.

Operation: (L) ← (L) - 1

Flags: SF : Set if no borrow occurred, cleared otherwise
SL : Unaffected

Example:

```
MOV     L,#3H
DECS    L               ; This instruction leaves the value 2H in REG L
```

SETB @HL,b

Binary Code:	0 0 1 1	0 0 d d

Description: This instruction sets indirect data memory bit addressed by registers H and L without affecting any other bits in the destination.

Operation: b ← 1 (b = 0,1,2,3)

Flags: SF : Set to one
SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#5H
SETB    @HL.2           ; Set RAM address 05H bit 2 to 1
```

DECS A,@HL

Binary Code:	0 0 1 1 | 0 1 d d
Description:	This instruction clears the indirect data memory bit addressed by registers H and L without affecting any other bits in the destination.
Operation:	b ← 1 (b = 0,1,2,3)
Flags:	SF : Set to one SL : Unaffected

Example:

```
MOV     H,#0H
MOV     L,#5H
CLRB    @HL.3       ; Clear RAM address 05H bit 3 to zero
```

4 DEVELOPMENT TOOLS

SMDS

The Samsung Microcontroller Development System, SMDS ,is a complete PC-based development environment for KS51840 microcontroller. The SMDS is powerful, reliable, and portable. The SMDS tool set includes a versatile debugging utility, trace with built-in logic analyzer, and performance measurement applications.

Its window-oriented program development structure makes SMDS easy to use. SMDS has three components:

— IBM PC- compatible SMDS software, all device-specific development files, and the SAMA assembler.
— Development system kit including main board, personality board, SMDS manual, and target board adapter, if required.
— Device-specific target board.

SMDS PRODUCT VERSIONS

As of the date of this publication, two versions of the SMDS are being supported:

— SMDS Version 4.8 (S/W) and SMDS Version 3.6 (H/W); last release: January, 1994.
— SMDS2 Version 5.3 (S/W) and SMDS2 Version 1.3 (H/W); last release: November, 1995.

The new SMDS2 Version 1.3 is intended to replace the older Version 3.6 SMDS. The SMDS2 contains many enhancements to both hardware and software. These development systems are also supported by the personality boards of Samsung's microcontroller series: KS56, KS57, and KS88.

SAMA ASSEMBLER

The Samsung Arrangeable Microcontroller (SAM) Assembler, SAMA, is a universal assembler, and generates object code in standard hexadecimal format.

Compiled program code includes the object code that is used for ROM data and required SMDS program control data. To compile programs, SAMA requires a source file and an auxiliary definition (DEF) file with device-specific information.

TARGET BOARDS AND PIGGYBACKS

Target boards are available for KS51840/51850 microcontroller. All required target system cables and adapters are included with the device-specific target board.

Piggyback chips are provided to customers in limited quantities for KS51840/51850 microcontroller. The KS51840/51850 piggyback chips, PB51840/51850-20 and PB51850-24 are now available.

PB51840/51850-20 is 20 DIP piggyback chip for 20 DIP, 20 SOP package device of KS51840/51850 microcontroller.
PB51840/51850-24 is 24DIP piggyback chip for 24 SOP package device of KS51850 microcontroller.

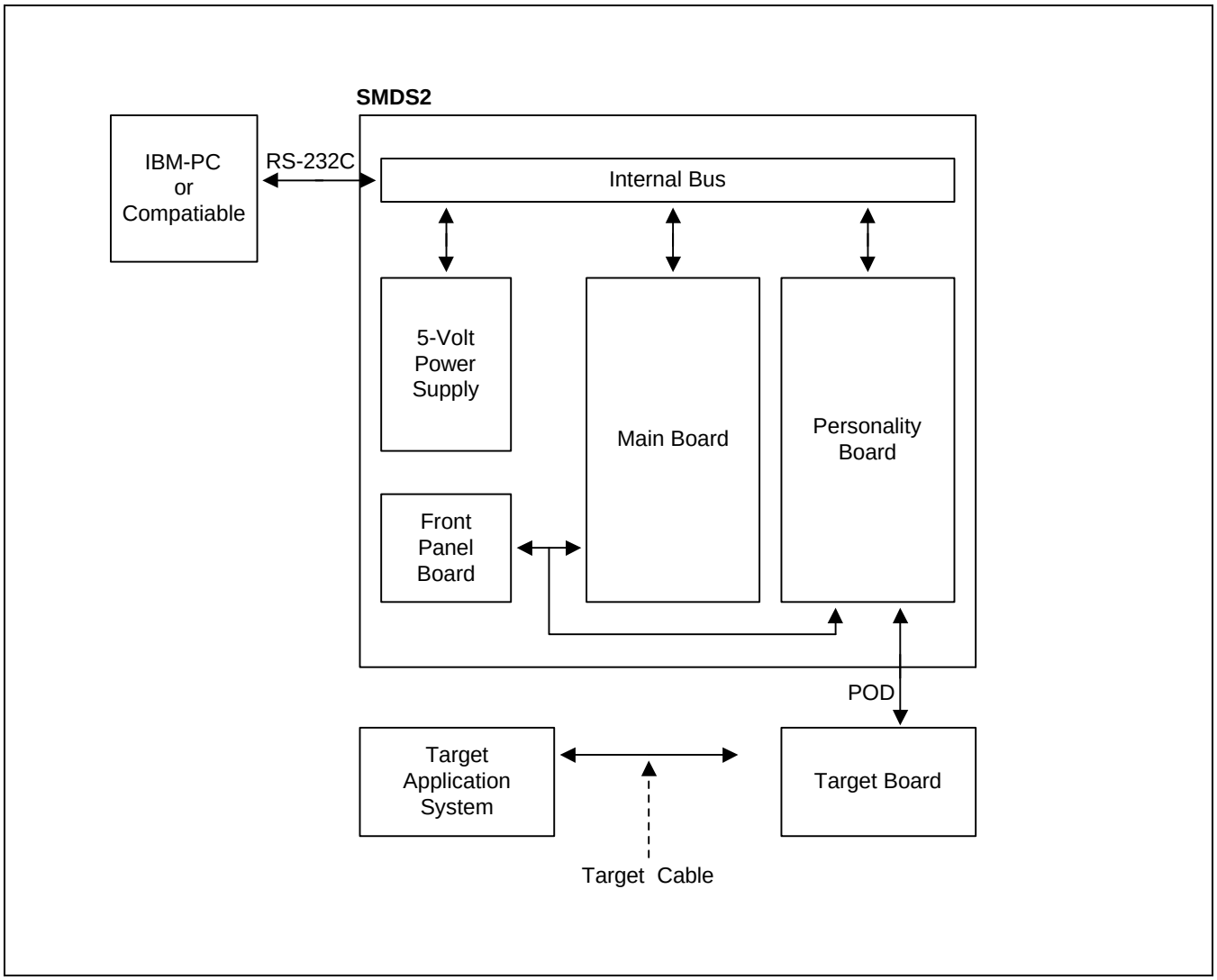


Figure 4-1. SMDS Product Configuration (SMDS2)

TB51840/51850A TARGET BOARD

The TB51840/51850A target board is used for the KS51840/51850 microcontroller. It is supported by the SMDS2 development system only.

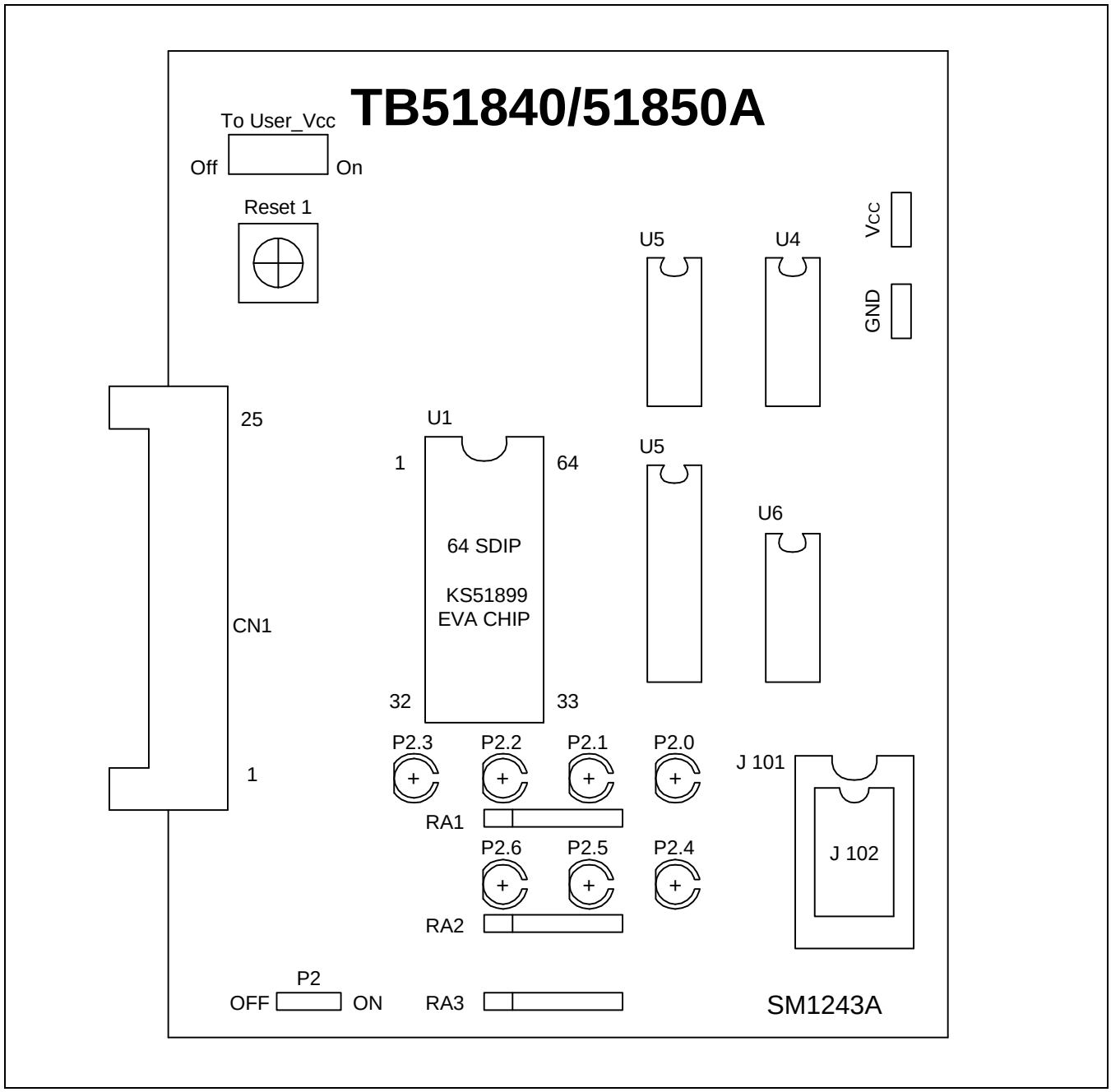
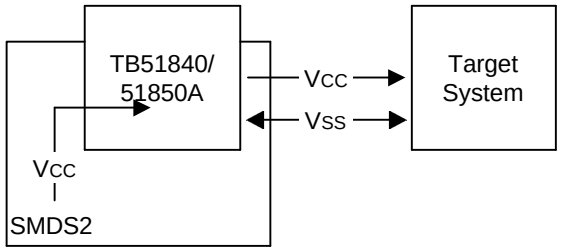
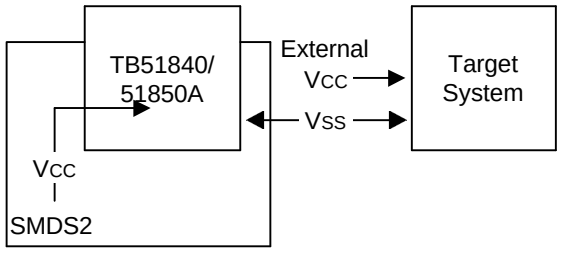


Figure 4-2. TB51840/51850A Target Board Configuration

Table 4-1. Power Selection Settings for TB51840/51950A

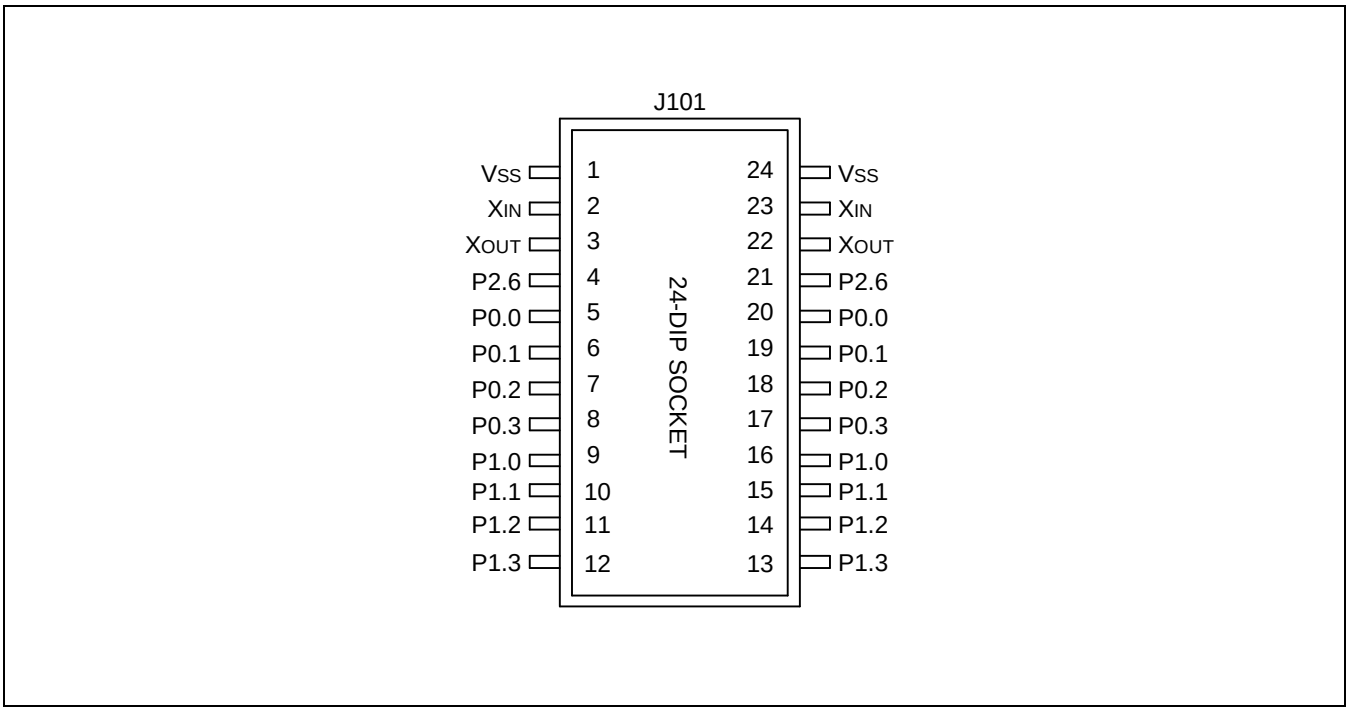
'To User_Vcc' Settings	Operating Mode	Comments
To User_Vcc OFF ON		The SMDS2 supplies V_{CC} to the target board (evaluation chip) and the target system.
To User_Vcc OFF ON		The SMDS2 supplies V_{CC} only to the target board (evaluation chip). The target system must have its own power supply.

LED 2.0-LED 2.6:

These LEDs are used to display value of the P2.0-P2.6. It will be turn on, if the value is Low.

P2 OPTION SWITCH:

Switch ON: You can see the port value using the LED display.
Switch OFF: You can't see the port value. That is, the LED won't be turn ON by the port value.



Figue 4-3. 24 DIP Socket for TB51840 (KS51840, 24 SOP)

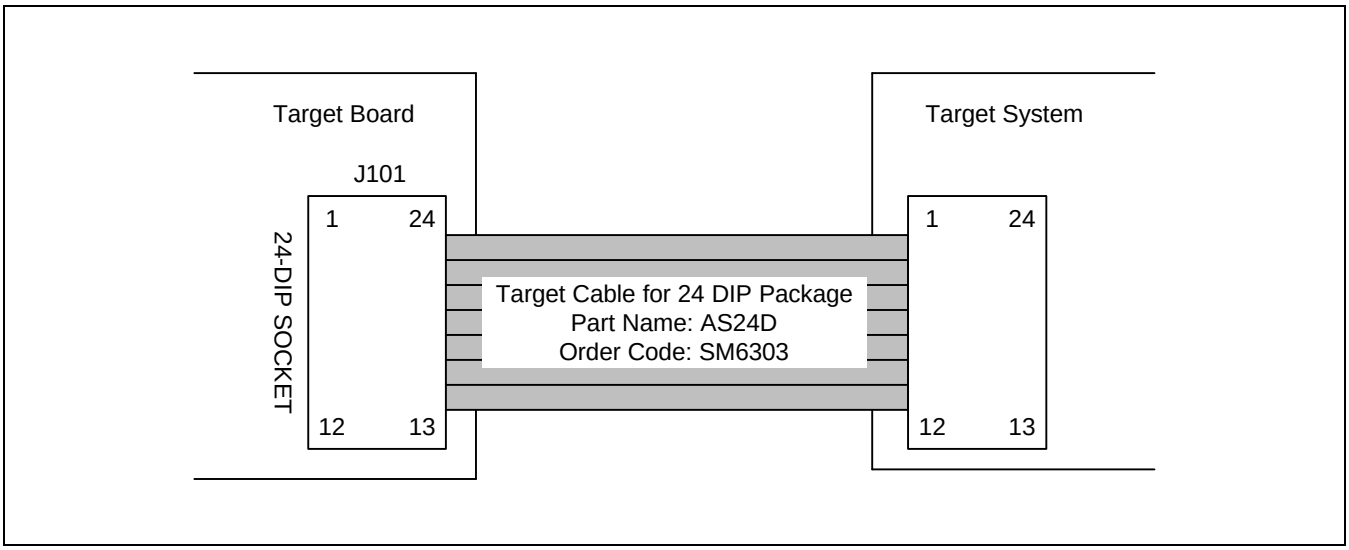


Figure 4-4. TB51840/51850A Cable for 24 DIP Package

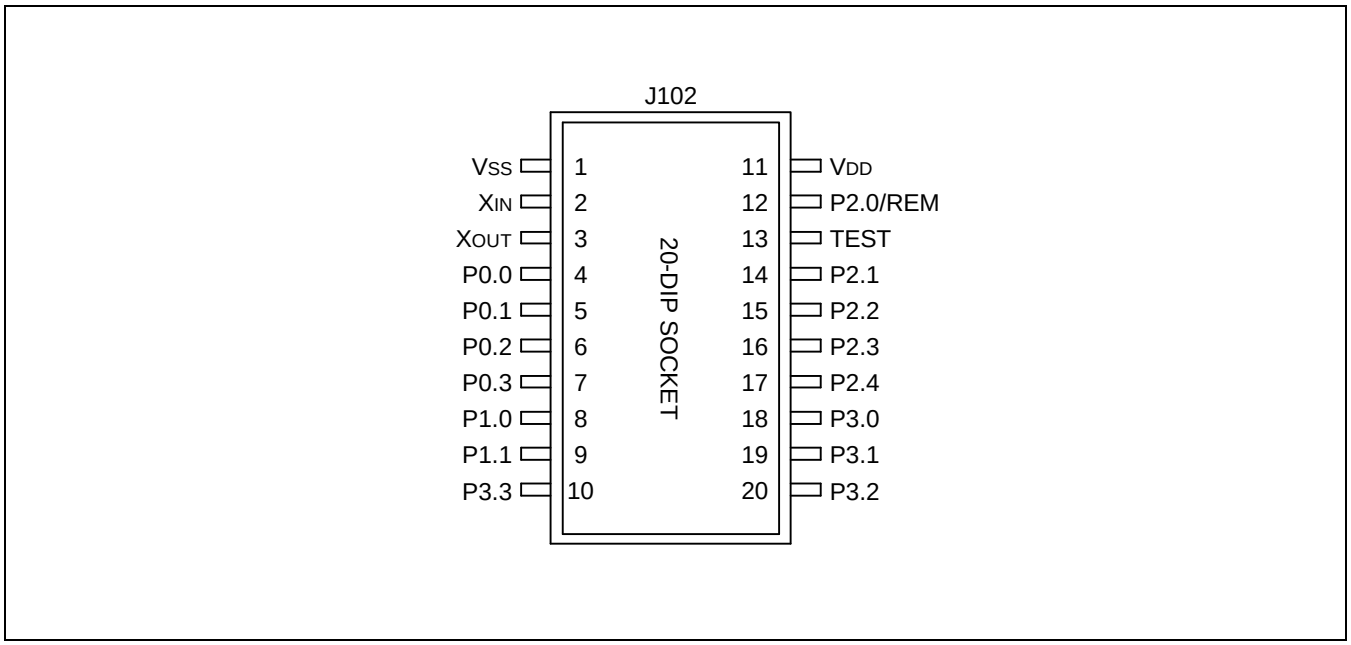


Figure 4-5. 20 DIP Socket for TB51840/51850A (KS51840/51850, 20 DIP, 20 SOP)

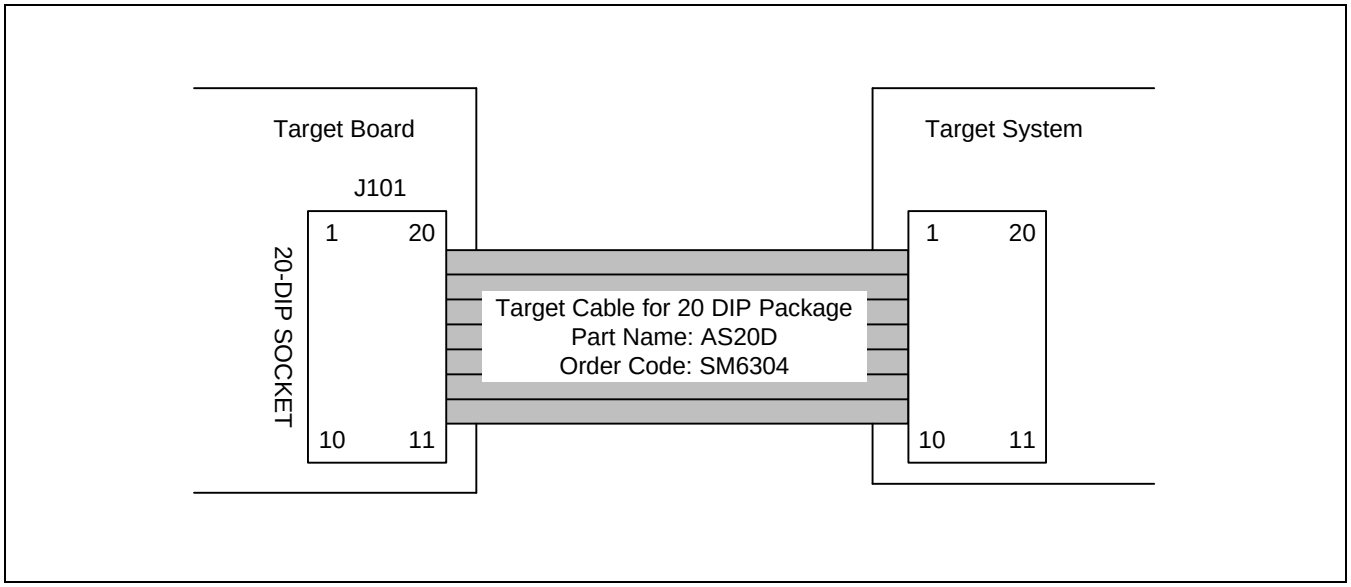


Figure 4-6. TB51840/51850A Cable for 20 DIP Package

5 REMOTE CONTROL TX.

OVERVIEW

The KS51840/51850 4-bit single-chip CMOS microcontroller is designed using the reliable SMCS-51 CPU core with on-chip ROM and RAM. An auto-reset circuit generates a RESET pulse in regular intervals, and can be used to initiate a Halt mode release. The KS51840/51850 microcontroller is intended for use in small system control applications that require a low-power and cost-sensitive design solution. In addition, the KS51840/51850 has been optimized for remote control transmitters. A difference between the KS51840 and KS51850 is that KS51850 has N-channel transistor for I.R.LED drive.

FEATURES

Table 5-1. KS51840/51850 Features

Feature	KS51840	KS51850
ROM	1024 bytes	1024 bytes
RAM	32 x 4 bits	32 x 4 bits
Carrier frequency	fxx /12, fxx /8, no carrier	fxx /12, fxx /8, no carrier
Operating voltage	250 kHz ≤ f_{OSC}≤ 3.9 MHz 1.8 V to 3.6 V, 3.9 MHz < f_{OSC} < 6 MHz 2.2 V to 3.6 V	250 kHz ≤ f_{OSC} ≤ 3.9 MHz 1.8 V to 3.6 V, 3.9 MHz < f_{OSC} < 6 MHz 2.2 V to 3.6 V
Package	24 SOP, 20 SOP/DIP	24 SOP, 20 SOP/DIP
Tr. for I.R.LED drive	x	Built-in

Table 5-2. KS51840/51850 Package Types

Item	24 pins	20 pins
Package	24 SOP-375	20 DIP-300A 20 SOP-300 20 SOP-375

Table 5-3. KS51840/51850 Functions

	Description
Automatic reset by Halt mode release	When Halt mode is released, the chip is reset after an oscillator stabilization interval of 9 ms. (fxx = 455 kHz)
Output pin state retention function	When the system enters Halt Mode, P3.0-P3.3, P2.0, and P2.2-P2.5 go low level in 24 pins. P3.0-P3.3, P2.0, and P2.2-P2.4 go low level in 20 pins. But the P2.0 is floating state in KS51850.
Auto-reset	With oscillation on and with no change to the IP2.0 output pin, a reset is activated every 288 ms at fxx = 455 kHz.
Osc. Stabilization time	CPU instructions are executed after oscillation stabilization time has elapsed.
Power-on reset circuit	resister: 1 MΩ, capacitor value: 7 pF.
Other functions	Carrier frequency generator. Halt wake-up function.

RESET

The KS51840/51850 has three kinds of reset operations:

— POR (Power-On Reset)

— Auto-reset

— Automatic reset by Halt release

Power-On Reset Circuits

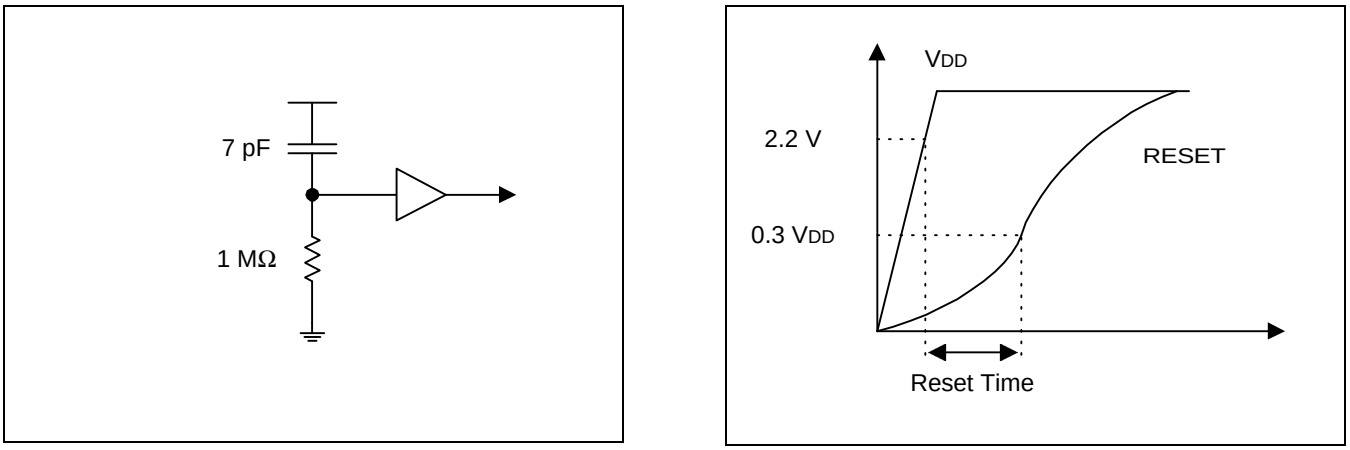


Figure 5-1. Power-On Reset Circuits

Auto-Reset

The auto-reset function resets the CPV every 131,072 oscillator cycles (288 ms at fxx = 455 kHz). The auto-reset counter is cleared when a rising edge is detected at IP2.0, or by a HALT or RESET pulse.

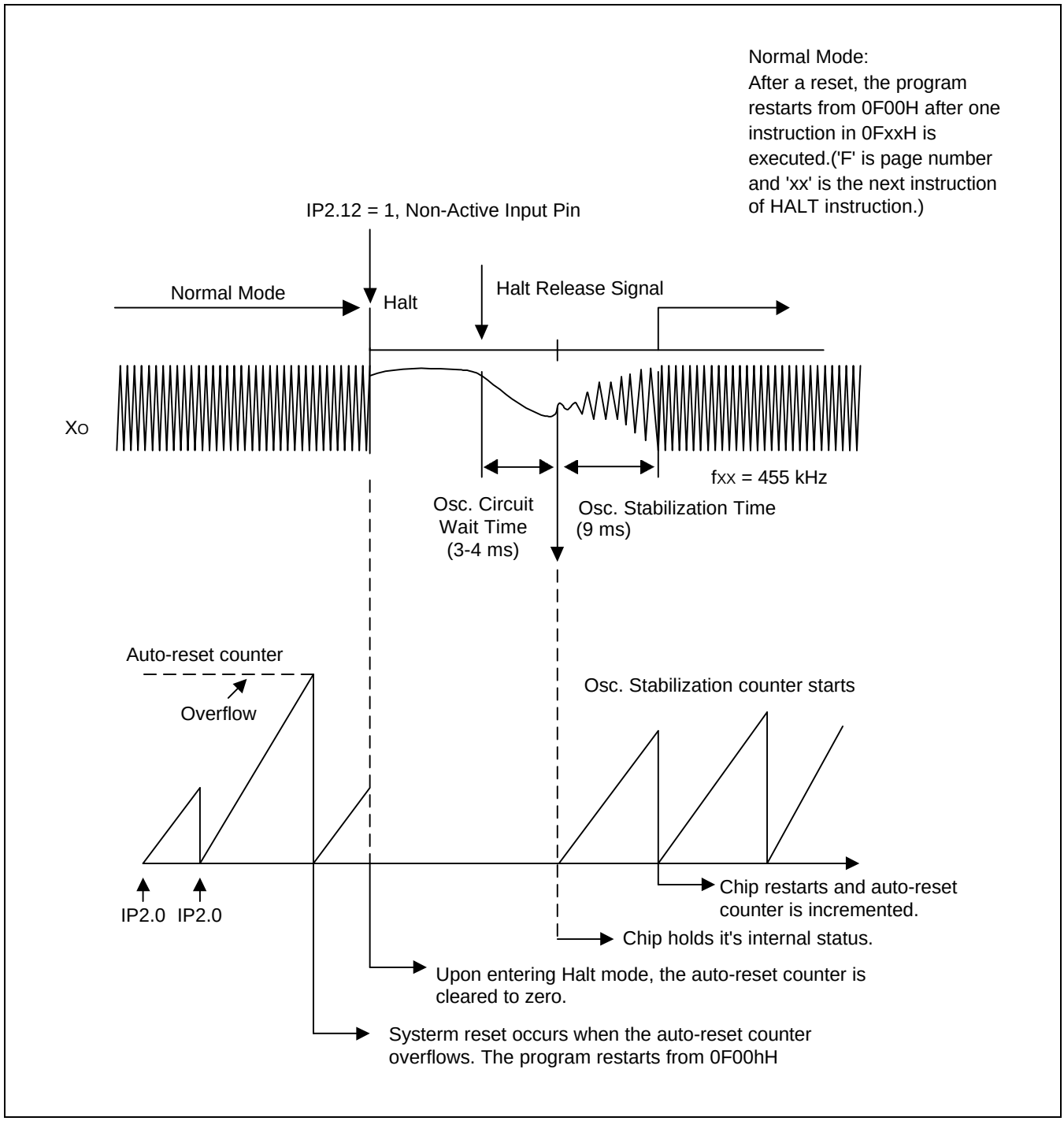


Figure 5-2. Auto-Reset Counter Function

Automatic Reset by Halt Mode Release

This function resets the CPV by releasing Halt mode. The CPV is reset to its initial operating status and program execution starts from the reset address.

Halt Mode and Automatic Reset by Halt Release

Halt mode is used to reduce power consumption by stopping the oscillation and holding the internal state. Halt mode can be entered by forcing IP2.12 to high level (remaining input pins are non-active).

Before entering Halt mode, programmer should pre-set all key strobe output pins to active state even though Halt mode causes some pins to remain active.
For the 24 pins, P3.0-P3.3, P2.0, P2.2- P2.4, and P2.5 are sent low and for 20 pins, P3.0-P3.3, P2.0, P2.2-P2.4 are sent low. Forcing any key input port to active state causes the clock oscillation logic to start system initialization.
At this time, the system is reset after the oscillation stabilization time elapses. A system reset causes program execution to start from address 0F00H.

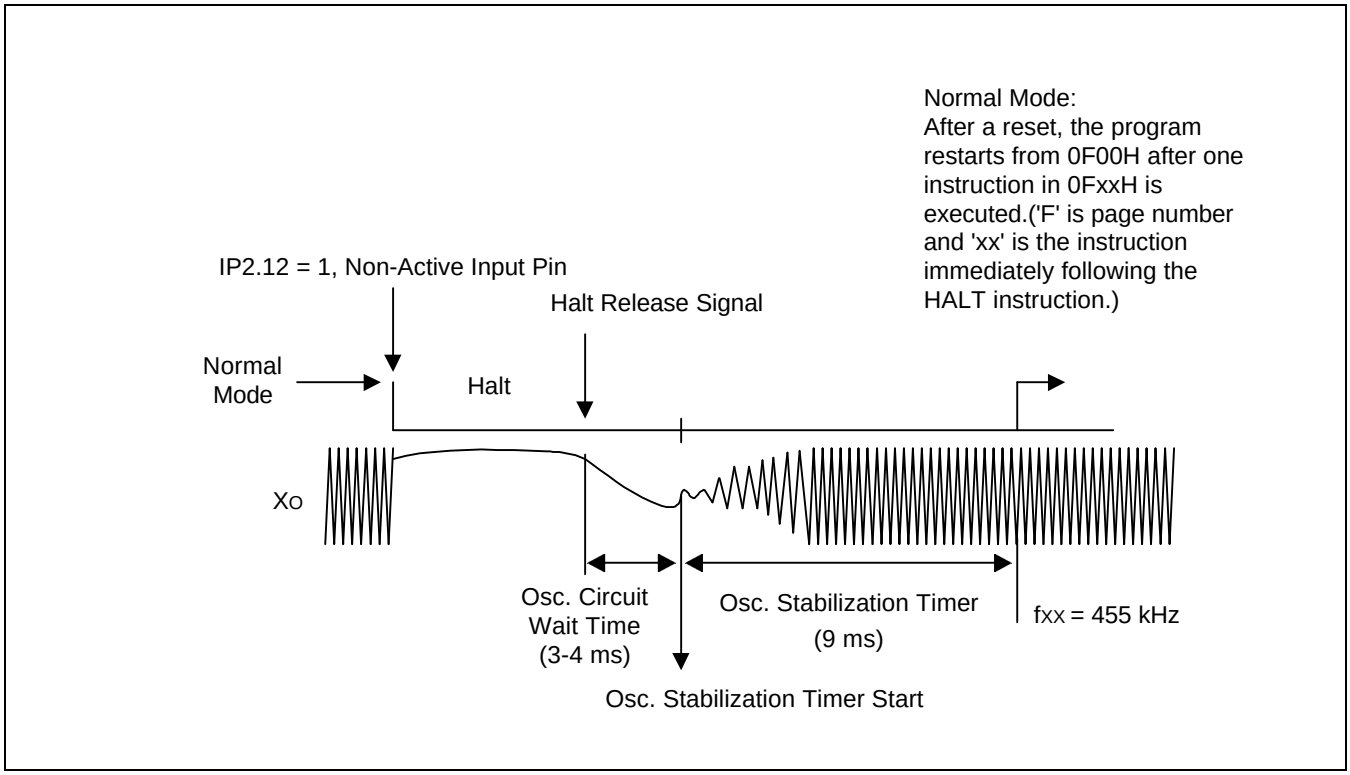


Figure 5-3. Reset Timing Diagram

HALT mode programming

The KS51840/51850 can enter Halt mode by setting the IP2.12 pin to high level and forcing P0 and P1 input to a normal state. If IP 2.12 is high and any input is active, the chip cannot enter Halt mode. Therefore, the next instruction is executed, which must be a clear command for IP2.12.

```
        MOV     L,#5
KEYOLO  CLRB    P2.(L)          ;  P2.5,4,3,2, ← Low
        DECS    L
        CPNE    L,#1
        JP      KEYOLO
        CLR     A               ;  Acc. ← #0h
        OUT     P3,@SL+A        ;  P3.0,1,2,3, ← Low
        MOV     L,#0DH
        CLRB    P2.(L)          ;  Select the P0 input
        IN      A,P0
        INCS    A               ;  P0 input check
        JP      .+2
        JP      KEYCHK          ;  If any key pressed in P0, jump to KEYCHK routine
        SETB    P2.(L)          ;  Select the P1 input
        IN      A,P0
        INCS    A               ;  P1 input check
        JP      + 2
        JP      KEYCHK          ;  If any key pressed in P1, jump to KEYCHK routine
        MOV     L,#0CH          ;  No key pressed
        SETB    P2.(L)          ;  Halt mode
```

; When no key is pressed, the chip enters Halt mode. Pressing any key while in Halt mode causes the chip to be initialized and restarted from the reset address.
; If any key is pressed between timea and timeb, the following instruction is executed.

```
        MOV     L,#0CH          ;  These two instructions remove the condition of
                                   re-entering
        CLRB    P2,(L)          ;  Halt mode.
```

RESET and HALT Logic Diagram

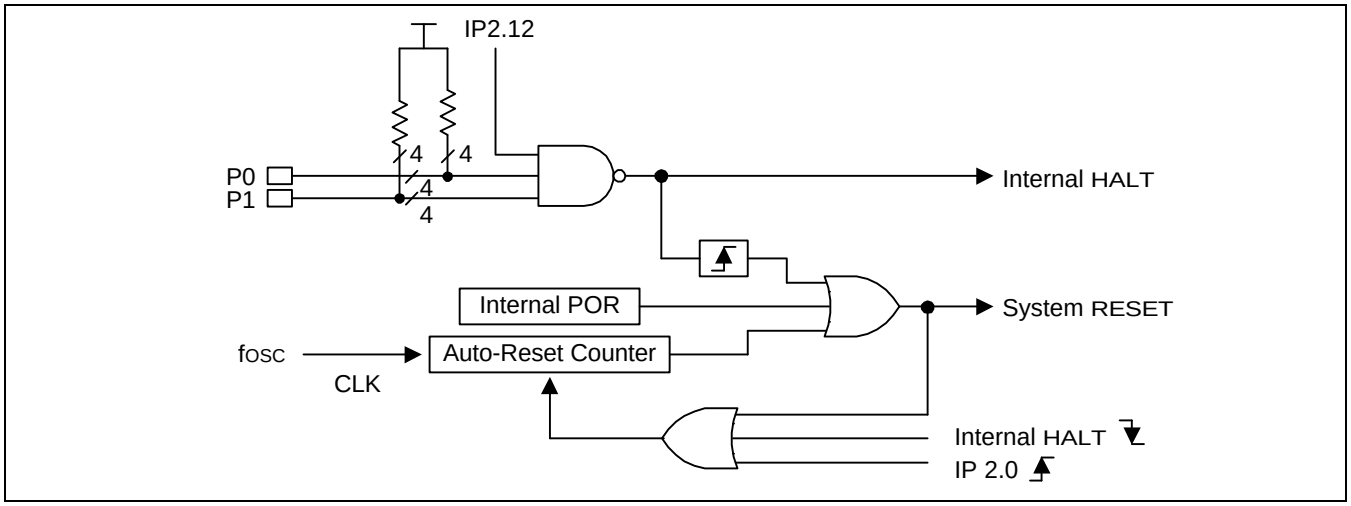


Figure 5-4. RESET and HALT Logic Diagram

OUTPUT PIN DESCRIPTION

Indicator LED Drive Output

To drive the indicator LDE, the programmer should use P2.1 of the KS51840/51850 (which have higher current drive capability than other pins) in order to retain the pre-programmed status during Halt mode. Be careful to turn on the LED when a reset signal is generated. Because a reset signal sends all of the internal and external output pins to low level, the programmer must set LED output P2.1 to high state using a reset subroutine.

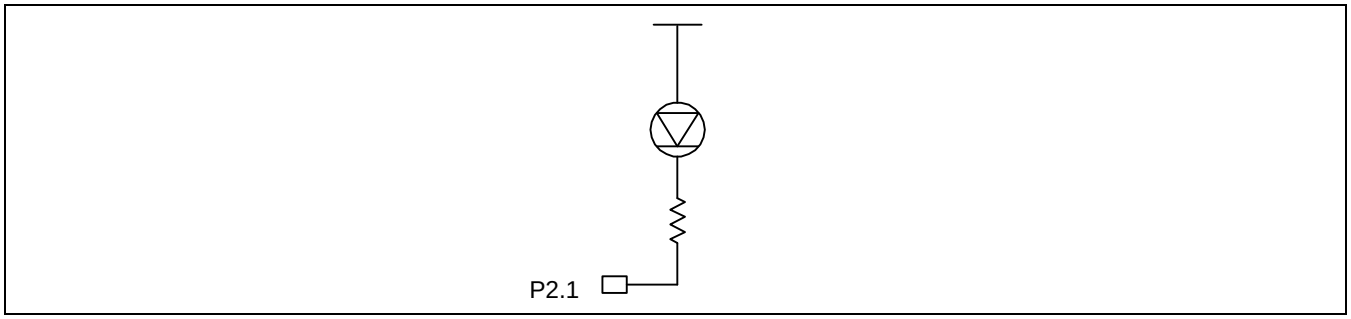


Figure 5-5. LED Drive Output Circuit

Strobe Output Option

To active the optional strobe output function for TV and VCR remocon applications, the programmer must use the option selection strobe output pin (P2.6).
This pin has lower current drive capability than other pins and retain the pre-programmed status while in Halt mode. Be careful to turn on the option strobe output pin when a reset signal is generated. Because the reset sends all internal and external output pins to low level, the option strobe output pin should always be non-active state
(H-Z). The pin should be active only when you are checking option status to reduce current consumption.

Table 5-4. Strobe Output Option

Pin usage	Key Output	LED Drive	Option Selection
P3.0-P3.3 P2.2-P2.5	00	X	X
P2.1	0	00	0
P2.6	0	0	00

NOTE: X = not allowed
0 = good
00 = better

Output Pin Circuit Type

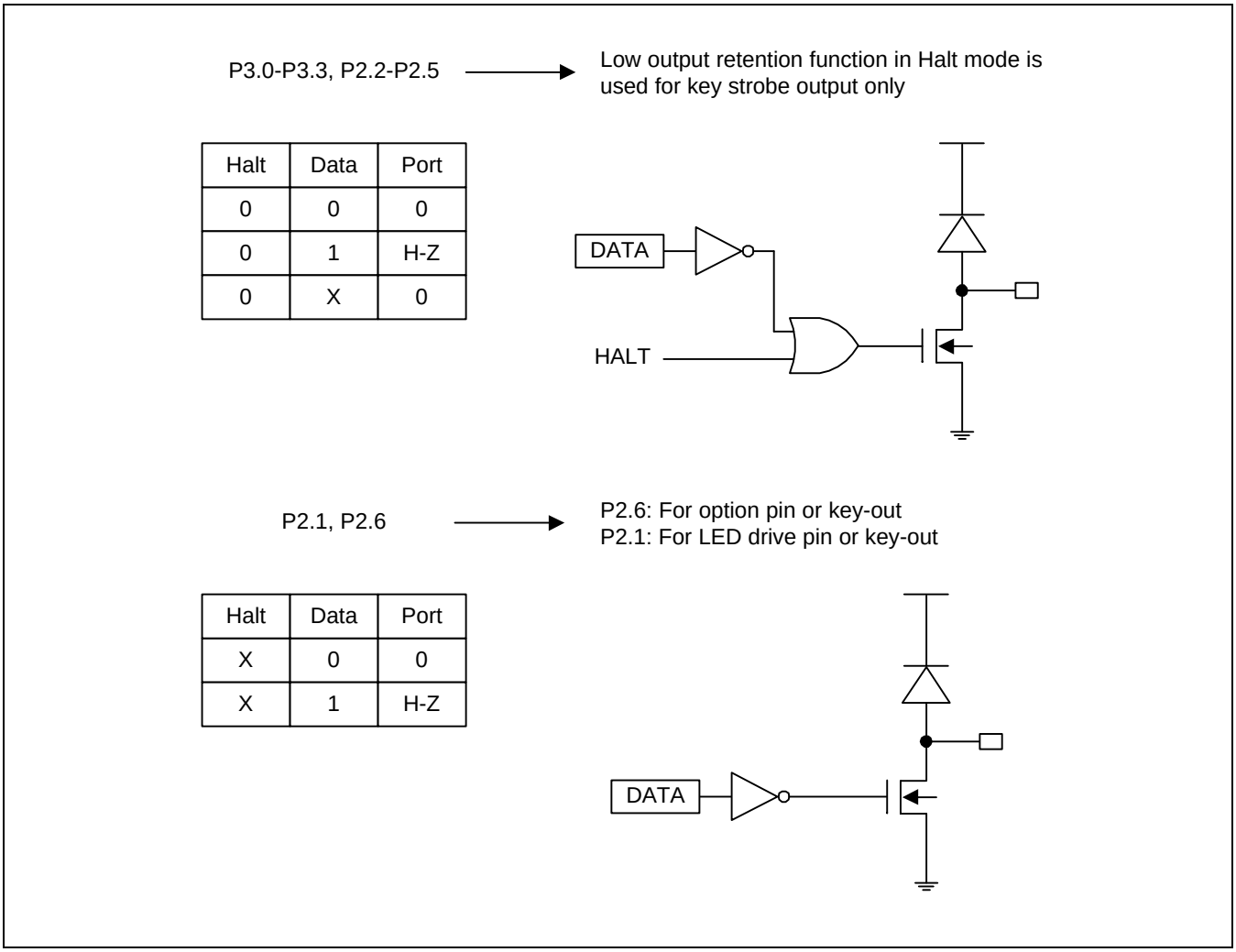


Figure 5-6. Output Pin Circuits

Soft Ware Delay Routine

To obtain a constant time value, the KS51840/51850 use a software delay routine (there is not an internal timer interrupt). One instruction cycle is six oscillator clocks. Using a ceramic resonator with a constant frequency, you can calculate the time delay as follows:

t = 6/fxx Number of Instructions

Where t: Elapsed time and fxx: System clock.

Programming Tip

To program a 1-ms delay: 1 ms = 6/455 kHz x n, where fxx = 455 kHz
Therefore, n = 75.8 = 76 instructions

```
DLY1MS    CLR     A
          ADDS    A,#0BH          ;  Two instructions
DLY       MOV     H,#0            ;  Dummy instruction
          MOV     H,#0            ;  Dummy instruction
          MOV     H,#0            ;  Dummy instruction
          MOV     H,#0            ;  Dummy instruction
          DECS    A
          JP      DLY             ;  DLY loop: 6 instructions
;2 + (ACC + 1) x instructions in loop = 2 + (11 + 1) x 6 = 74
          CLR     A
          CLR     A               ; Two instructions.
; Total number of instructions for DLY1MS is 76.
```

NOTE

In order to lengthen the delay time, you can use an arithmetic instruction combination of L register and Accumulator. The L register causes the address lower pointer to access RAM spce and the output port pointer to control the P2 (individual/serial output) port status.

— RAM manipulation instruction: RAM address pointer.

```
MOV       A,@HL CPNE   @HL,A
ADDS      A,@HL SETB   @HL.b
```

— P2 output control instruction: P2 pointer.

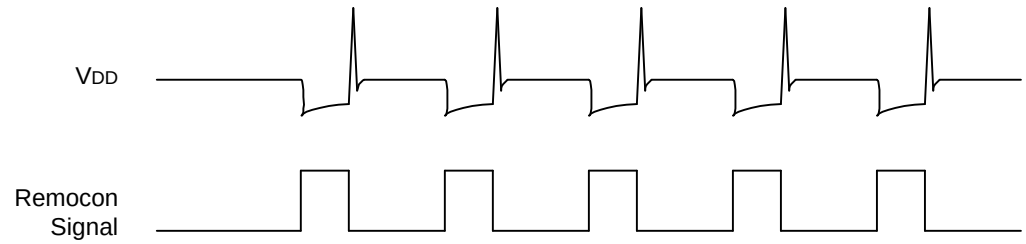
```
SETB   P2.(L)   CLRB   P2.(L)
```

PROGRAMMING GUIDELINES

When programming KS51840/51850 microcontroller, please follow the guidelines presented in this subsection.

PCB Artwork

For remote control applications, turning the I.R.LED on and off may cause variations in transmission current ranging from a few hundred μA to a few hundred mA. This current variation generates overshoot and undershoot noise on the power line, causing a system malfunction.



To reduce noise and to stabilize the chip's operation, we recommend that the application designer reduce overshooting of the I.R.LED drive current and design PCB for the remote controller as follows: (The noise level should be limited to around 0.5 Vp-p, where Vp-p is the peak-to-peak voltage)

— Oscillation circuit should be located as near as possible to the chip.

— PCB pattern for V_{DD}/V_{SS} should be as wide and short as possible.

— I.R.LED drive TR and I.R.LED should be located as far as possible from the chip.

— Power supply battery and power capacitor should be located as near as possible to the chip.

— The ground pattern of the TEST pin (TEST pin I.R.LED drive TR) and V_{SS} pin should be separated and connected directly with the battery terminal.

— The ceramic capacitor (0.1 uFor 0.01uF) is recommended to use noise filter.

SMDS

When a breakpoint or single-step instruction is executed in area of PAGE and JP or CALL instruction, the JP or CALL may jump to the wrong address, We therefore recommend using a JPL or CALL instruction (instead or PAGE and JP or PAGE and CALL) to avoid this problems. Note that JP and CALL are 2-byte instructions.

Programming Guidelines for Reset Subroutine

1. We recommend that you initialize a H register to either “0” or “4”
2. Do not write the instructions CALLL (PAGE + CALL) or JPL (PAGE + JP) to the reset address 0F00H. In other words, do not use a PAGE instruction at 0F00H.
3. Turn off the LED output pin.
4. To reduce current consumption, do not set the option output pin to active state.
5. Pre-set the remocon carrier frequency (to fxx/12, fxx /8, and so on) before remocon signal transmission.
6. Because the program is initialized by an auto-reset or Halt mode release, even in normal operating state, do not pre-set all RAM data. If necessary, pre-set only the RAM area you need.
7. Be careful to control output pin status because some pins are automatically changed to active state.
8. To enter Halt mode, the internal port, IP2.12, should be set to high level and all of the input pins should be set to normal state.
9. To release Halt mode, an active level signal is supplied to input pins. If pulse width is less than 9 ms at fxx = 455 kHz, nothing happens and program re-enters Halt mode. That is, the external circuit should maintain the input pulse over a 9-ms interval in order to release Halt mode. After Halt mode is released, the hardware is reset. The hardware reset sends all internal and external output pins low (except P2.0 in KS51850) and clears the stack to zero. However, H,L and A registers retain their previous status.
10. If a rising edge is not generated at IP2.0, reset signal occurs every 288 ms at fxx = 455 kHz. To prevent an auto-reset, IP2.0 should be forced low and then high at regular intervals (within 288 ms at fxx = 455 kHz).

KS51840 Application Circuit Example

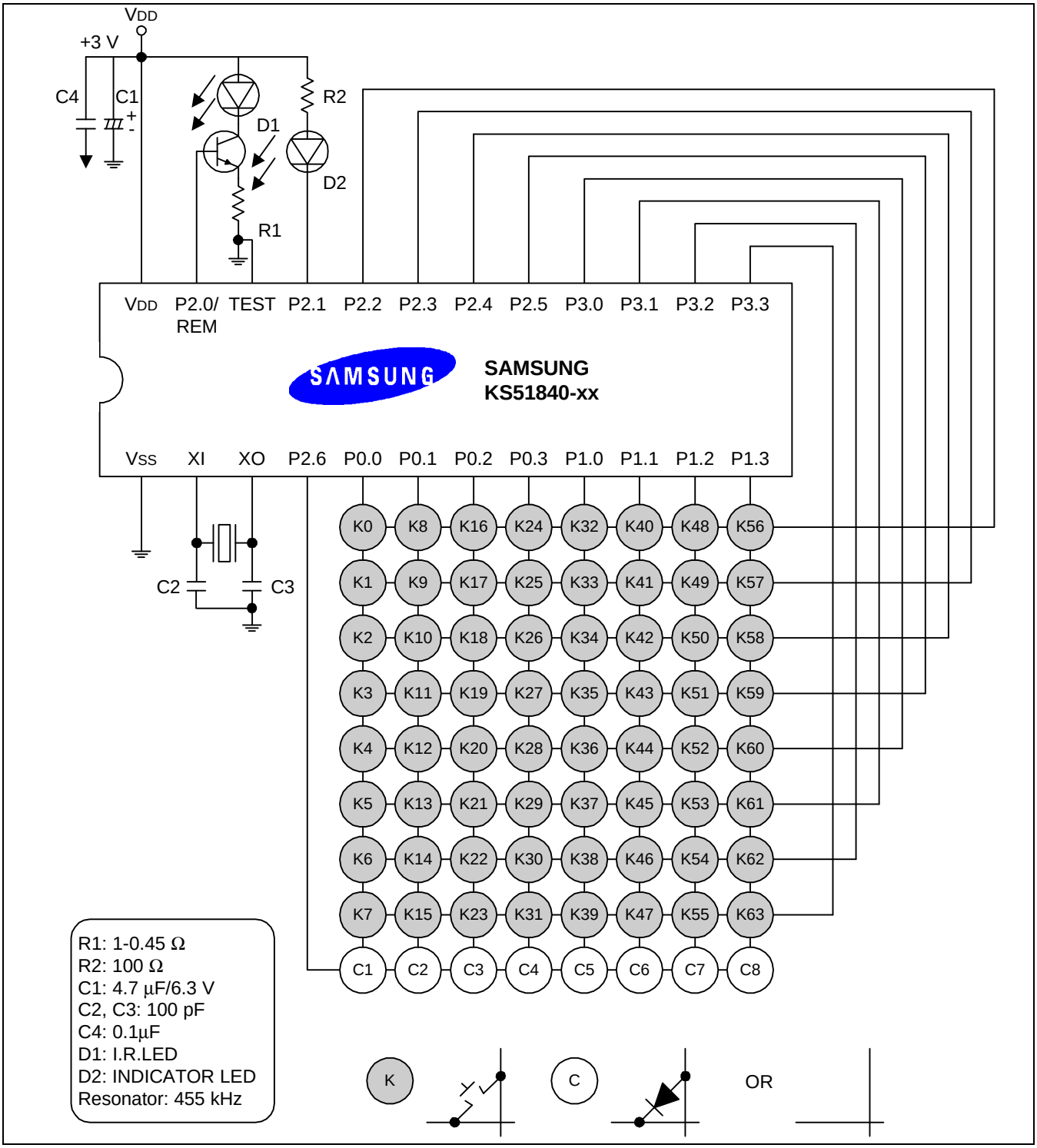


Figure 5-7. KS51840 Applicatrion Circuit Example

KS51850 Application Circuit Example

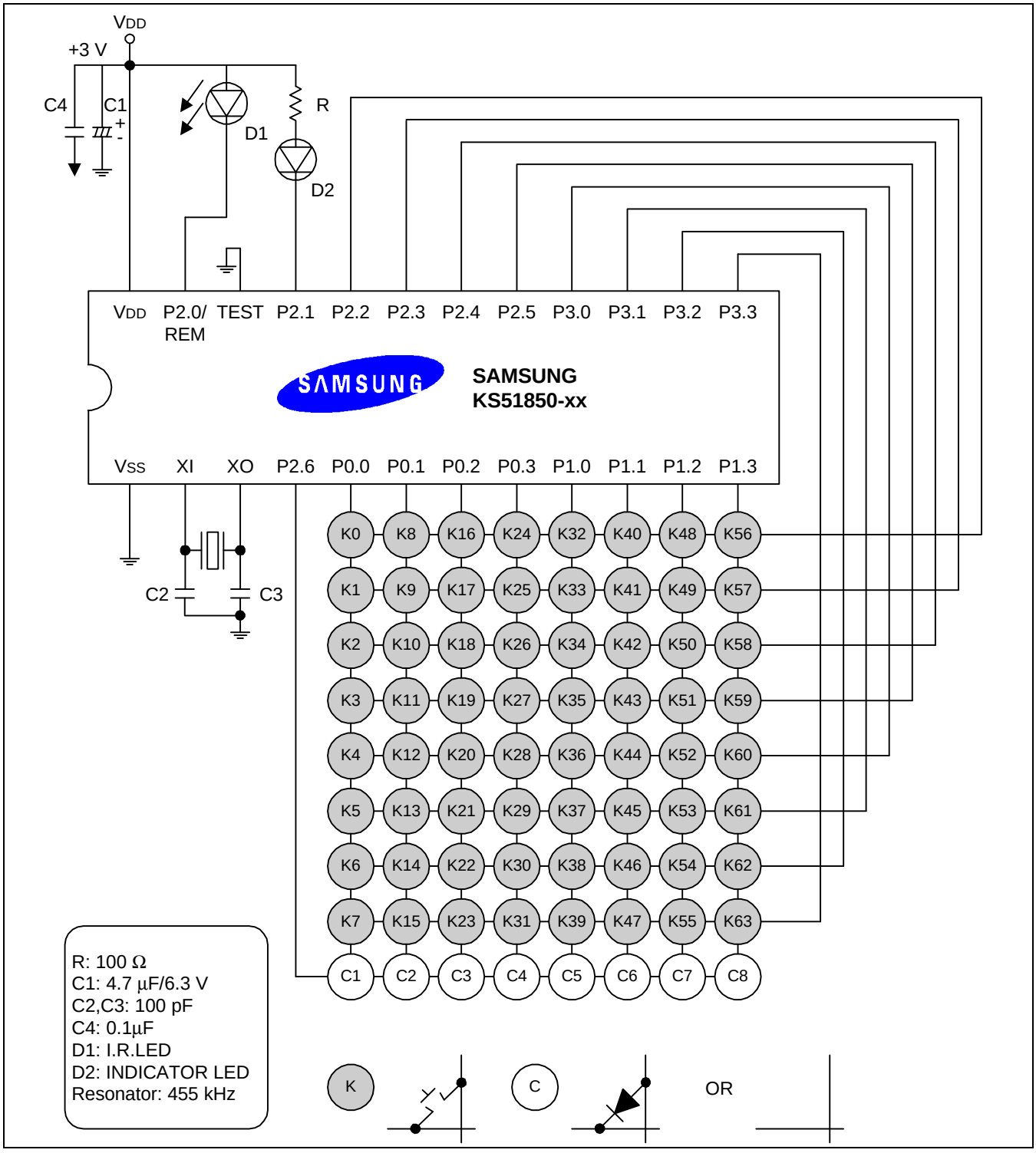


Figure 5-8. KS51850 Application Circuit Example

Program Flowchart

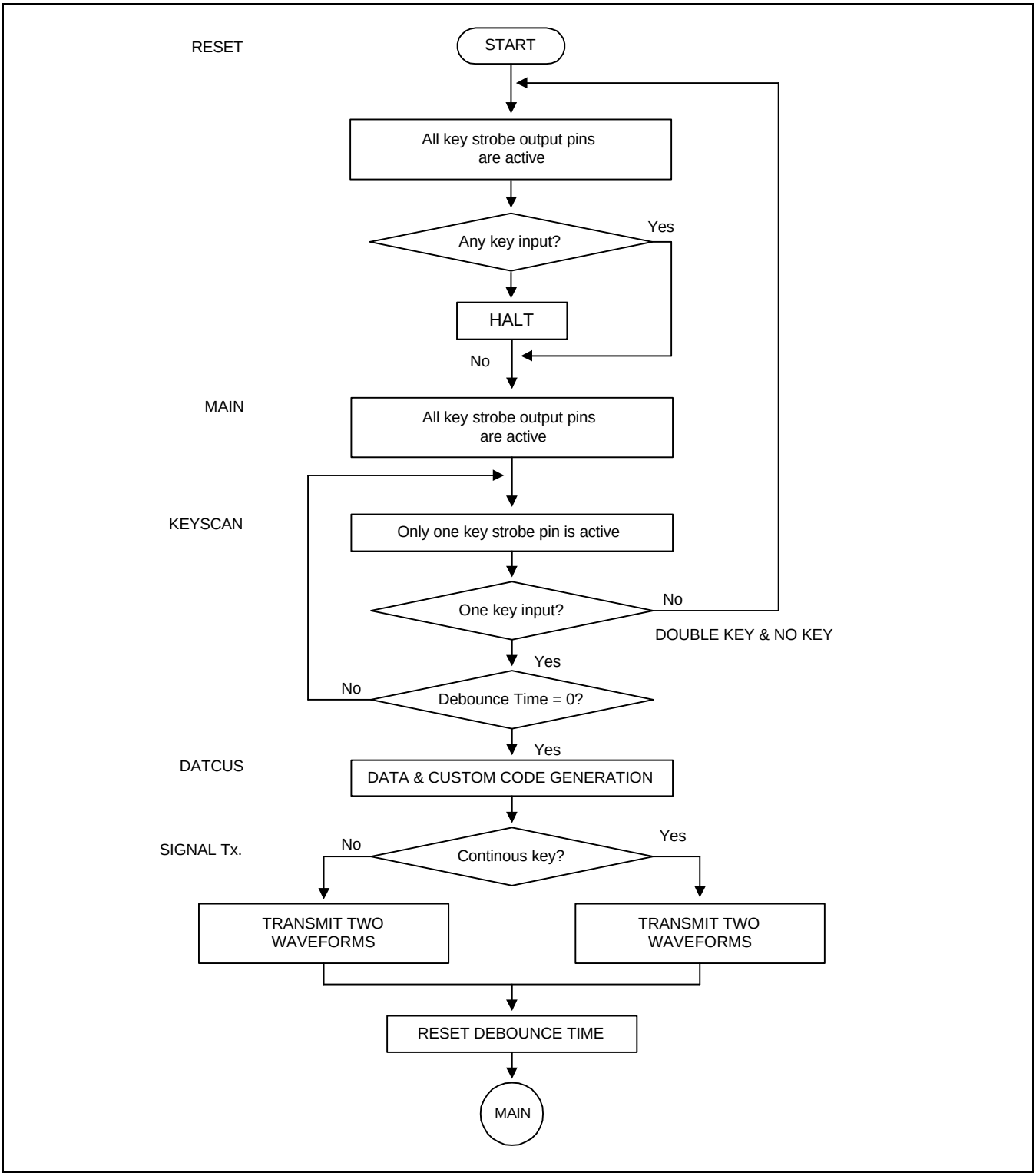


Figure 5-9. Program Flowchart 1

KS51840/51850 KEYSCAN FUNCTION

Description

This program has an 8 x 9 key matrix, which consists of input P0 and P1 and output P2 and P3. Because pull-up resistors are connected, the normal state for all input pins is high level. The operating method for the keyscan function is as follows:

— All output pins remain active state (= low).

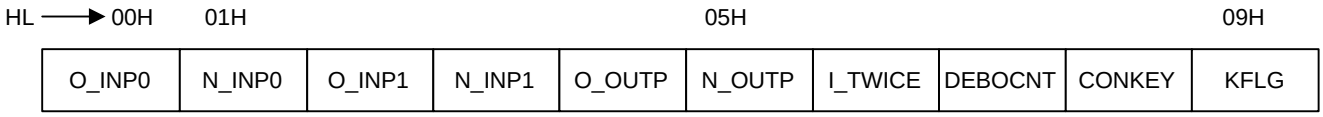
— If key is pressed, set all output pins to non-active state and rotate the pins to set only a pin to active state during debounce time.

— If key is pressed more than one or if no key is pressed, go to reset label.

— IF a new key is pressed, reset debounce time, continous flag, and key-in flag.

RAM Assignment

H register selects #0



O_INP0: he old value of P0
N_INP0: The new value of P0
O_INP1: The old value of P1
N_INP1: The new value of P1
O_OUTP: The old value of output port
N_OUTP: The new value of output port
I_TWICE: Double number increment
DEBOCNT: Debounce time
CONKEY: Continous key flag
KFLG: key input flag

Program Flowchart

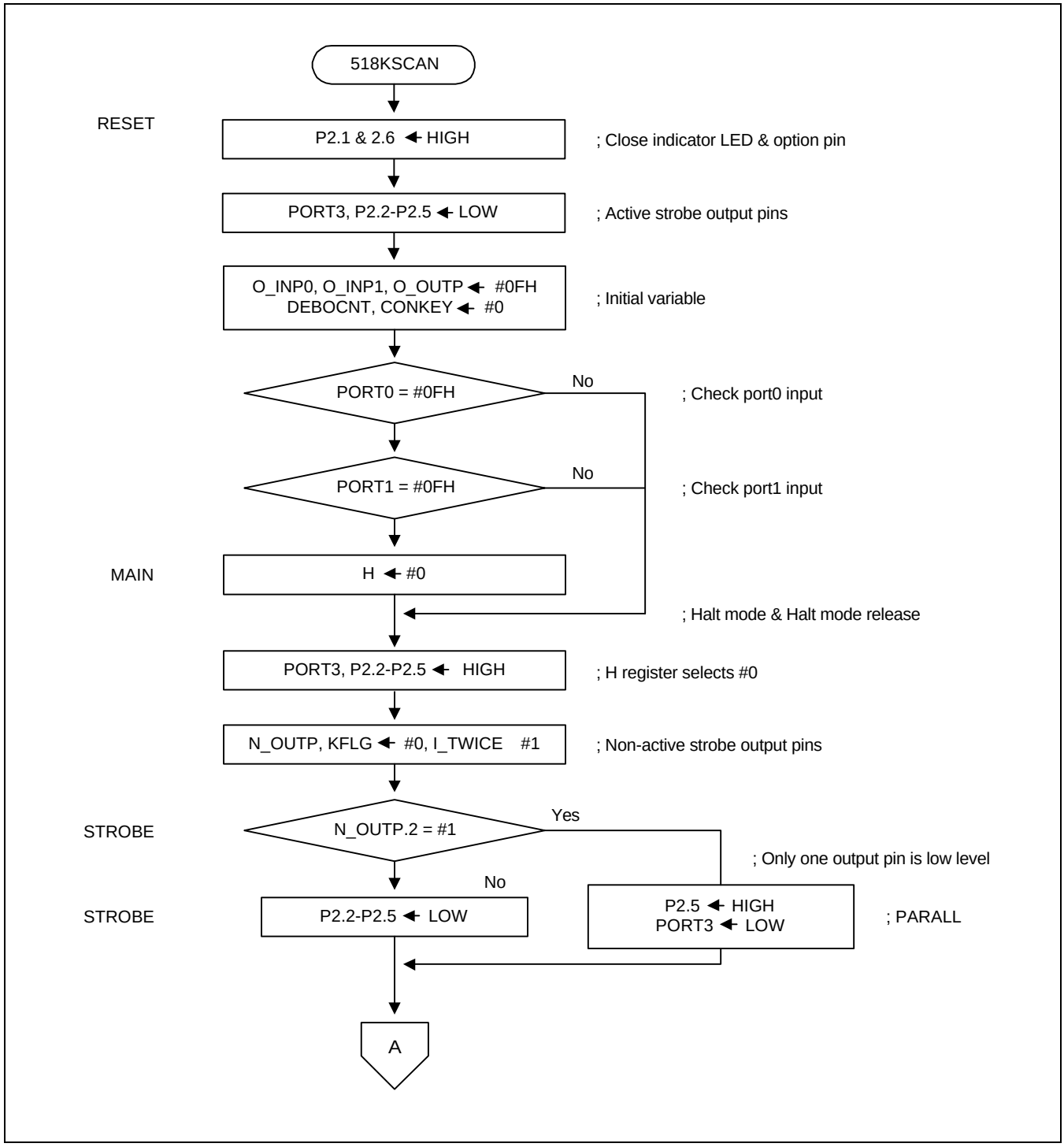


Figure 5-10. Program Flowchart 2

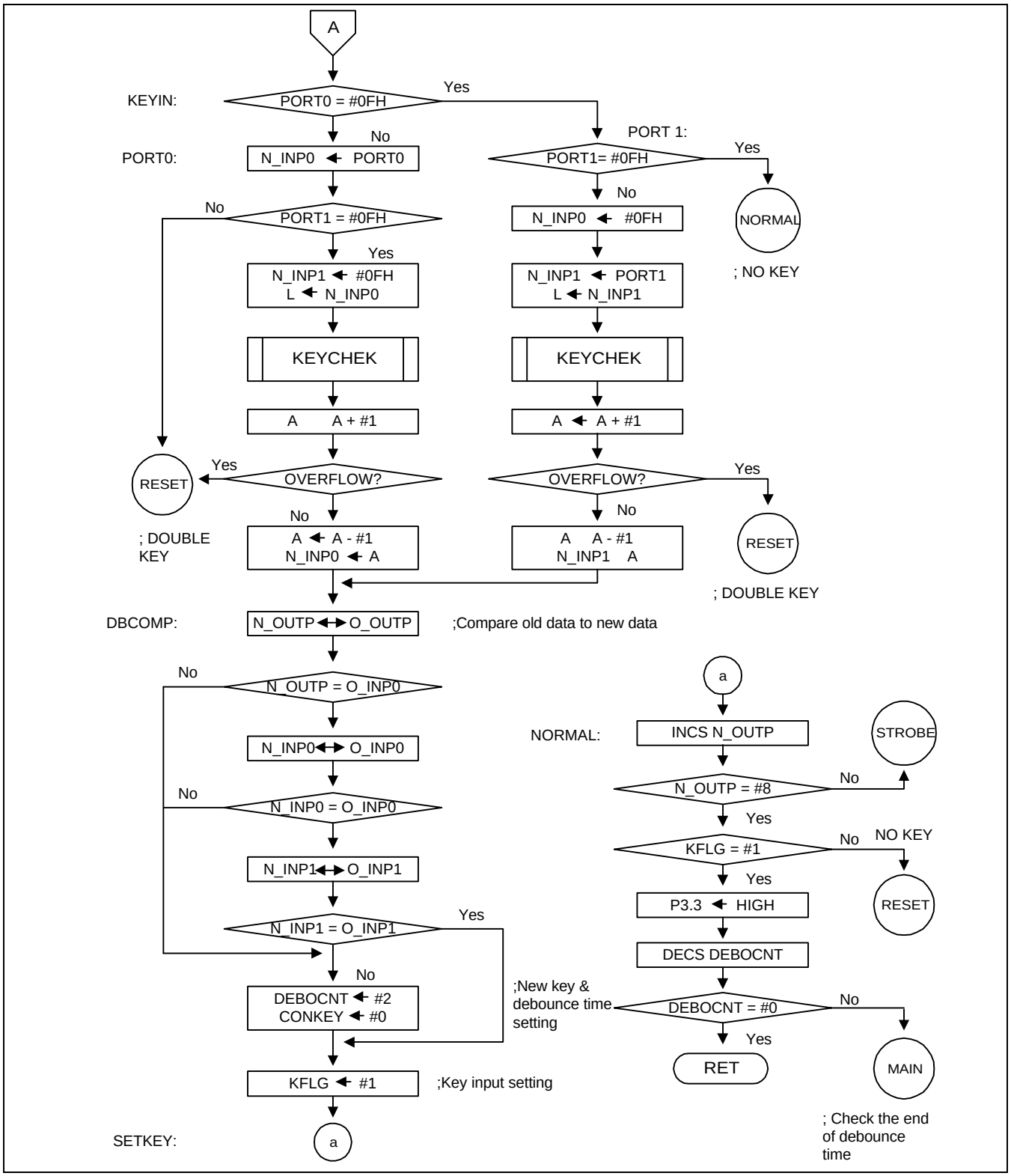


Figure 5-11. Program Flowchart 3

KS51840/51850 Keycheck Subroutine

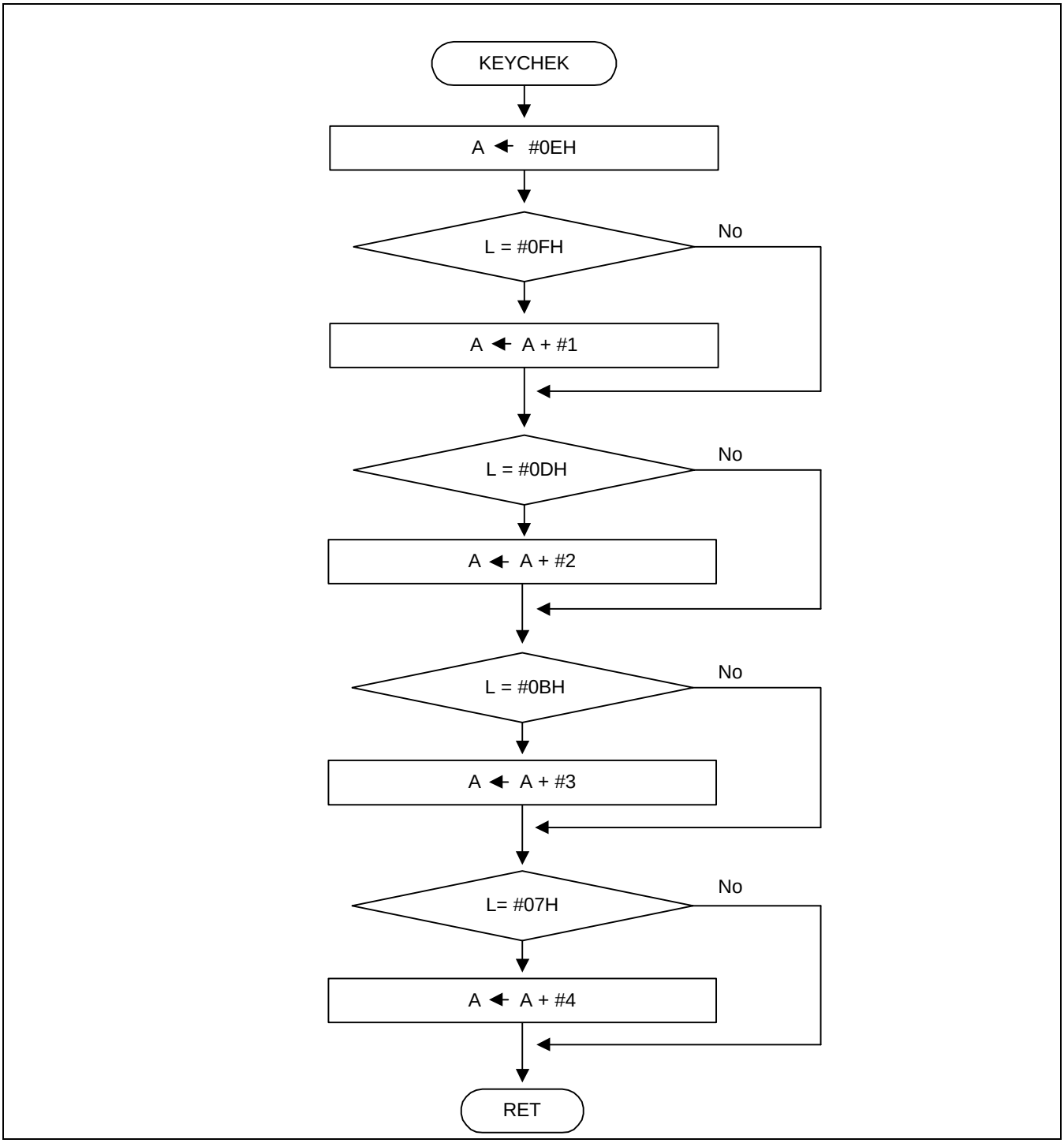


Figure 5-12. Ks51840/51850 Keycheck Subroutine

```
;*********************************************

                ORG         0F00H
;*********************************************

; If reset occurs, PA register is immediately initialized to #0FH
RESET       MOV         L,#1                ;   close indicator LED
            SETB        P2.(L)              ;
            MOV         L,#6                ;   non-select P2.6
            SETB        P2.(L)              ;

PRTCLR      CLR         A                   :
            OUT         P3,@SL + A          ;   low all the output ports
            MOV         L,#5                ;   (except P2.0, P2.1, P2.6)
            CLRB        P2.(L)              ;
            DECS        L                   ;
            CPNE        L,#1                ;
            JP          .-3                 ;

;;; initial all of the variables ----------------------------------------------------
;;; → input ports are connected with pull-up resistor
;;; → Therefore, normal state → high
            MOV         H,#0                ;   H register selects file #0
            MOV         L,#O_INP0           ;   port0 is #0fh
            MOV         @HL+,#0FH
            MOV         L, #O_INP1          ;   port1 is #0fh
            MOV         @HL +,#0FH
            MOV         L, #O_OUTP          ;   the strobe out is #0fh
            MOV         @HL+,#0FH
            MOV         L,#DEBOCNT          ;   debounce count is #0
            MOV         @HL+,#0
            MOV         L,#CONKEY           ;   continuous key is #0
            MOV         @HL+,#0

;;; check each input port (=key input) ----------------------------------

            MOV         L,#0DH              ;   check port0
            CLRB        P2.(L)              ;
            IN          A,P0                ;
            MOV         L,A                 ;
            CPNE        L,#0FH              ;
            JP          DELAYP0             ;

            MOV         L,#0DH              ;   check port1
            SETB        P2.(L)              ;
            IN          A,P0                ;
            MOV         L,A                 ;
            CPNE        L,#0FH              ;
            JP          J_MAIN              ;
```

```
;;; halt mode, after halt mode release, go to reset ------------------
                MOV     L,#0CH
                SETB    P2.(L)          ;   halt mode
                CLRB    P2.(L)          ;   halt mode release
                JP      RESET

PRTSET          CLR     A               ;
                ADDS    A,#0FH          ;
                OUT     P3,@SL+A        ;
                MOV     L,#5            ;
                SETB    P2.(L)          ;
                DECS    L               ;
                CPNE    L,#1            ;   If P2.0 is high, data Tx.
                JP      .- 3            ;   and auto reset counter clear
                RET

DELAYP0         MOV     H #0            ;   for the match of delay time
                MOV     H #0            ;
                MOV     H #0            ;
                MOV     H #0            ;
                MOV     H #0            ;
                MOV     H #0            ;
J_MAIN          JPL     MAIN
;
;
;
;
;
;
;*************************************************

                ORG     0000H
                JPL     RESET
;*************************************************

MAIN            MOV     H,#0            ;   H regisster selects file #0
                                        ;   useful when continous pulse Tx.
                                        ;
                CALLL   PRTSET          ;   high all the output ports

;;; initial useful variable in main routine ----------------------------------

                MOV     L,#N_OUTP       ; N_OUTP ← #0
                MOV     @HL+,#0         ;
                MOV     L,#1_TWICE      ;   I_TWICE (double increment) ← #1
                MOV     @HL+,#1         ;
                MOV     L,#KFLG         ;   KFLG ← #0 (input key flag)
                MOV     @HL+,#0         ;
```

```
;; select output pin by one and one ---------------------------------
STROBE    MOV       L,#N_OUTP
          CPBT      @HL.2           ;  If N_OUTP.2 is set, go to parall (parallel port)
          JP        PARALL          ;  otherwise, go to serial (serial port)
SERIAL    MOV       L,@HL           ;
          INCS      L               ;
          SETB      P2.(L)          ;
          INCS      L               ;
          CLRB      P2.(L)          ;
          JPL       KEYIN

PARALL    MOV       L,#5            ;  high P2.5
          SETB      P2.(L)          ;

;;;**********************************************
;;; A ← #0h
;;; A ← A-1_TWICE
;;; output P3
;;; I_TWICE ← I_TWICE + I_TWICE
;;; ;**********************************************

          CLR       A               ;  A ← #0FH
          ADDS      A, #0FH         ;
          MOV       L, #1_TWICE     ;
          XCH       @HL,A           ;  A ← A-I_TWICE
          SUBS      A,@HL           ;
          OUT       P3,@SL+A        ;  low port3
          SUBS      A,@HL           ;
          MOV       @HL,A           ;  recover I_TWICE
          ADDS      A,@HL           ;
          MOVZ      @HL,A           ;  I_TWICE ← I_TWICE + I_TWICE
          JPL       KEYIN

;;;**********************************************
;; check double key at each port
;; if a key pressed, do adds instruction
;; otherwise, induce overflow occurrence
;;;**********************************************

KEYCHEK   CLR       A
          ADDS      A,#0FH          ;  A ← #0fh
          CPNE      L,#0EH
          JP        .+2
          ADDS      A,#1            ;  A ← #0
          CPNE      L,#0DH
          JP        .+2
          ADDS      A,#2            ;  A ← #1
```

```
                CPNE    L,#0BH
                JP      .+2
                ADDS    A,#3                    ;   A ← #2
                CPNE    L,#7
                JP      .+2
                ADDS    A,#4                    ;   A ← #3
                RET

;
;
;
;
; ********************************************

                ORG     0100H
                JPL     RESET
; ********************************************

KEYIN           MOV     L,#0DH
                CLRB    P2.(L)

;;; select port0 ---------------------------------
                IN      A, P0
                MOV     L,A
                CPNE    L,#0FH                  ;   is key pressed in port0 ?
                JP      PORT0
                JP      PORT1

PORT0           MOV     L,#N_INP0               ;   setting at N_INP0
                MOV     @HL,A                   ;
                MOV     L,#ODH                  ;
                SETB    P2.(L)                  ;
                IN      A,P0                    ;
                MOV     L,A                     ;
                CPNE    L,#0FH                  ;
                JP      DBKEY                   ;   If also port1 input a key,
                                                ;   it is double key
                MOV      L,#N_INP1              ;   Only N_INP0 input
                MOV     @HL+,#0FH               ;
                MOV     L,#N_INP0               ;
                MOV     L,@HL                   ;
                CALLL   KEYCHEK                 ;
                ADDS    A,#1                    ;
                JP      DBKEY                   ;   if overflow occurs, it is double key
                DECS    A                       ;   because input value ranges
                MOV     L,#N_INP0               ;   from #0  to  #3
                MOVZ    @HL,A                   ;   N_INP0 ← A
                JPL     DBCOMP
```

```
;;; select port1 --------------------------------

PORT1       MOV         L,#0DH
            SETB        P2.(L)
            IN          A,P0
            MOV         L,A
            CPNE        L,#0FH              ;   is key pressed in port 1?
            JP          .+3
            JPL         NORMAL              ;   no key, go  to NORMAL
            MOV         L,#N_INP0           ;   setting N_INP0  to  #0fh
            MOV         @HL+,#0FH           ;   Only N_INP1 input
            MOV         L,#N_INP1           ;
            MOV         @HL,A               ;
            MOV         L,A                 ;   L ← N_INP1
            CALLL       KEYCHEK
            ADDS        A,#1                ;   if overflow occurs, go to double key
            JP          DBKEY               ;
            DECS        A                   ;   because input value ranges from
                                                #0  to  #3
            MOV         L,#N_INP1           ;   N_INP1 ← A
            MOVZ        @HL,A
            JPL         DBCOMP

;;; if double key occurs, go to reset ---------------------------------

DBKEY       JPL         RESET
;
;
;
;
; ***********************************************

            ORG         0200H
            JPL         RESET
; ***********************************************

;;; compare for the recognition of a new key----------------------
DBCOMP      MOV         H,#N_OUTP           ;   compare N_OUTP  to    O_OUTP
            MOV         A,@HL
            MOV         L,#O_OUTP
            XCH         @HL,A
            JP          FSTKEY
            MOV         L,#N_INP0           ;   compare N_INP0  to  O_INP0
            MOV         A,@HL
            MOV         L,#O_INP0
            XCH         @HL,A
            CPNE        @HL,A
            JP          FSTDLY
            MOV         L,#N_INP1           ;   compare N_INP1  to  O_INP1
            MOV         A,@HL
            MOV         L,#O_INP1
```

```
                XCH         @HL,A
                CPNE        @HL,A
                JP          FSTKEY
                JP          SETKEY

FSTDLY          MOV         H,#0                ;   for match of delay time
                MOV         H,#0
                MOV         H,#0
                MOV         H,#0
                MOV         H,#0

;;; when new key input ----------------------------
FSTKEY          MOV         L,#DEBOCNT          ;   DEBOCNT ← #2
                MOV         @HL+,#2
                MOV         L,#CONKEY           ;   CONKEY ← #0
                MOV         @HL+,#0

SETKEY          MOV         L,#KFLG             ;   KFLG ← #1
                MOV         @HL+,#1

; ***********************************************

;;; increase N_OUTP
;;; check N_OUTP is equal to #8
;;; check no key (= DEBOCNT)
; ***********************************************

NORMAL          MOV         L,#N_OUTP           ;   increase N_OUTP
                INCS        A,@HL
                MOVZ        @HL,A
                ADDS        A,#8                ;   A ← #8
                CPNE        @HL,A               ;   compare N_OUTP to A
                JP          J_STRO              ;   go to stroble label
                MOV         L,#KFLG             ;
                CPBT        @HL.0               ;   check key flag
                JP          ONKEY

                JPL         RESET               ;   no key

ONKEY           CLR         A
                ADDS        A,#0FH
                OUT         P3,@SL + A          ;   set port3 to '1'
                MOV         L,#DEBOCNT          ;   decrease DEBOCNT
                DECS        A,@HL
                XCH         @HL,A
                CPNZ        @HL                 ;   compare DEBOCNT TO #0
                JP          J1_MAIN
                JPL         KEYSCAN
J1_MAIN         JPL         MAIN
J_STRO          JPL         STROBE
```

PROBEKS51840/51850 CODE GENERATION

Description

This program generates data code and custom code. The custom code is determined according to diodes between input ports and output pin (P2.6). The data code is as follows:

RAM ← DAT0 (D0-D3), DAT1 (D4-D7)

	D0	D1	D2	D3	D4	D5	D6	D7
KEY0	0	0	0	0	0	0	0	0
KEY1	1	0	0	0	0	0	0	0
. .				. .				
KEY31	1	1	1	1	1	0	0	0
KEY32	0	0	0	0	0	0	1	0
KEY33	1	0	0	0	0	0	1	0
. .				. .				
KEY63	1	1	1	1	1	0	1	0

RAM ← DAT0 (D0-D3), DAT1_0 (D4-D7)

	D0	D1	D2	D3	D4	D5	D6	D7
KEY0	0	0	0	0	0	0	0	0
KEY1	1	0	0	0	0	0	0	0
. .				. .				
KEY31	1	1	1	1	1	0	0	0
KEY32	0	0	0	0	0	1	0	0
KEY33	1	0	0	0	0	1	0	0
. .				. .				
KEY63	1	1	1	1	1	1	0	0

RAM Assingment

H register selects #4

HL ▶ 40H 41H 45H 49H

CUS0	CUS1	CUS2	CUS3	DAT0	DAT1	DAT2	DAT3	DAT1_0	DAT3_0

CUS0; Custom code (c0-c3)
CUS1: Custom code (c4-c7)
CUS2: The complement of CUS0
CUS3: The complement of CUS1
DAT0: Data code (d0-d3)]
DAT1: Data code (d4-d7)
: → 32 key: 00000010, 63 key: 1111010
DAT2: The complement of DAT0
DAT3: The complement of DAT1
DAT1_0 Data code (d4-d7)
: → 32 key: 00000100, 63 key: 1111100
DAT3_0 The complement of DAT1_0

Program Flowchrt

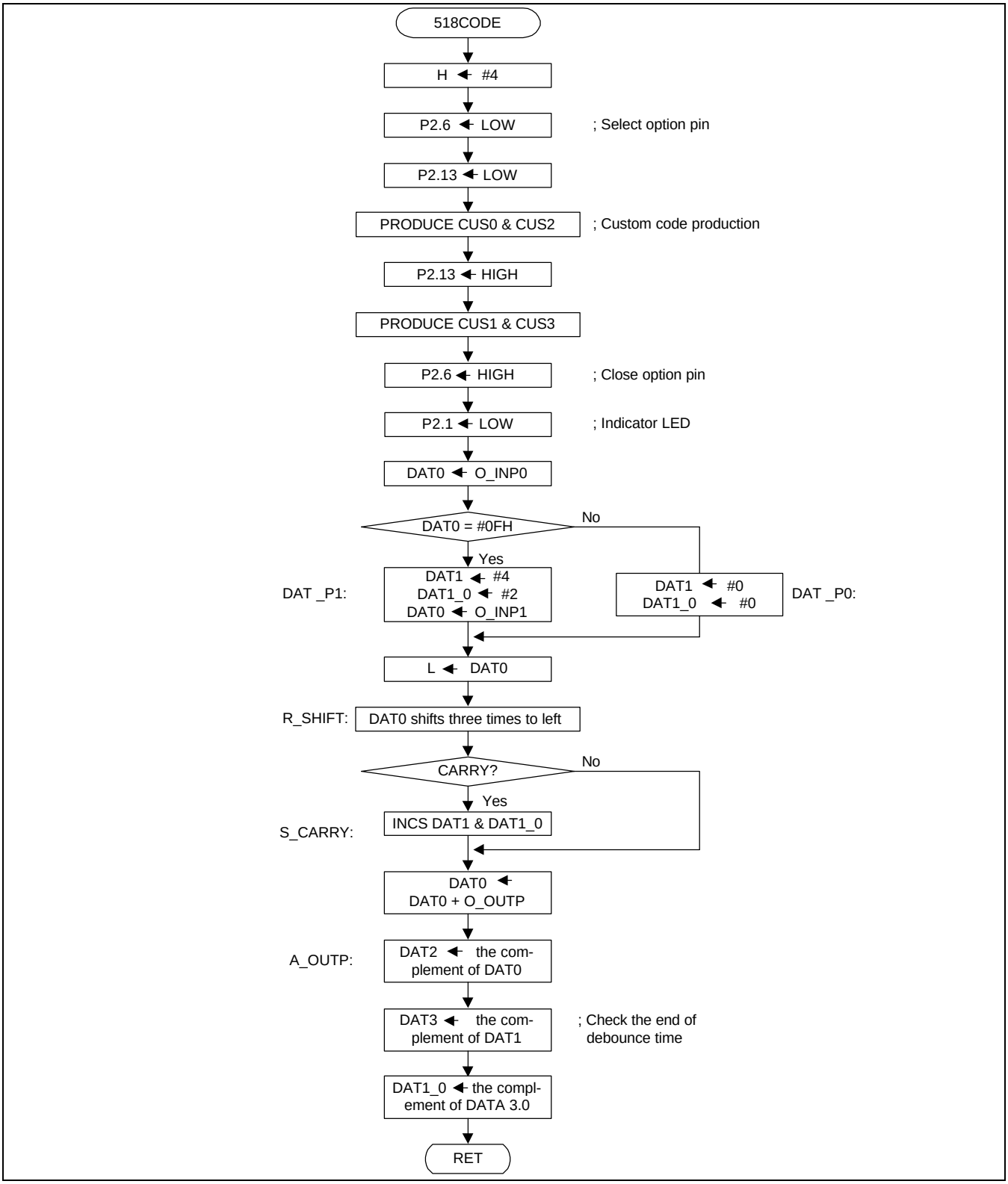


Figure 5-13. Program Flowchart 4

```
;*********************************************

                ORG         0300H
                JPL         RESET
;*********************************************

;;; select only a key ------------------------------------------------
KEYSCAN     MOV         H,#4

;;; product custom code ----------------------------------------------

                MOV         L,#6                ;   P2.6 ← low
                CLRB        P2.(L)              ;   check custom code

                MOV         L,#0DH              ;
                CLRB        P2.(L)              ;
                IN          A,P0                ;
                MOV         L,#CUS2             ;   CUS2 is the complement of CUS0
                MOV         @HL,A               ;
                NOTI        A                   ;
                DECS        A                   ;
                MOV         L,#CUS0             ;
                MOV         @HL,A               ;

                MOV         L,#0DH
                SETB        P2.(L)              ;   CUS3 is the complement of CUS1
                IN          A,P0                ;
                MOV         L,#CUS3             ;
                MOV         @HL,A               ;
                NOTI        A                   ;
                DECS        A                   ;
                MOV         L,#CUS1             ;
                MOVZ        @HL,A               ;

                MOV         L,#6                ;   high P2.6
                SETB        P2.(L)

                MOV         L,#1                ;   the indicator LED of a key input
                CLRB        P2.(L)
```

```
;;; product data code ------------------------------------------------
                MOV     H,#0
                MOV     L,#O_INP0               ;   DAT0 ← O_INP0
                MOV     A,@HL
                MOV     H,#4
                MOV     L,#DAT0
                MOVZ    @HL,A

                ADDS    A,#0FH                  ;   A ← #0fh
                CPNE    @HL,A                   ;   does input key exist in port0?
                JP      DAT_P0

DAT_P1          MOV     L,#DAT1                 ;   input key exists in port1
                MOV     @HL+,#04H               ;   DAT1 ← DAT1 + #4
                MOV     L,#DAT1_0
                MOV     @HL+,#02H               ;   DAT1_0 ← DAT1_0 + #2
                MOV     H,#0                    ;   DAT0 ← O_INP1
                MOV     L,#O_INP1               ;
                MOV     A,@HL                   ;
                MOV     H,#4                    ;
                MOV     L,#DAT0                 ;
                MOVZ    @HL,A                   ;
                JPL     R_SHIFT                 ;
DAT_P0          MOV     L,#DAT1                 ;   clear DAT1 & DAT1_0
                MOV     @HL+,#0
                MOV     L,#DAT1_0
                MOV     @HL+,#0
                MOV     L,#DAT0
                MOV     H,#4                    ;   delay time
                MOV     H,#4                    ;
                MOV     H,#4                    ;
                MOV     H,#4                    ;
                MOV     H,#4                    ;
                JPL     R_SHIFT

;
;
;
```

```
;*********************************************
            ORG         0400H
            JPL         RESET
;*********************************************

R_SHIFT     MOV         A,@HL               ;   DAT0 shifts three times to the left
            ADDS        A,@HL               ;
            MOV         @HL,A               ;
            ADDS        A,@HL               ;
            MOV         @HL,A               ;
            JP          S_CARRY
            JP          N_CARRY

S_CARRY     MOV         L,#DAT1             ;if carry occurs, increase DAT1 & DAT1_0
            XCH         @HL,A               ;
            INCS        A                   ;
            XCH         @HL,A               ;

            MOV         L,#DAT1_0           ;
            XCH         @HL,A               ;
            INCS        A                   ;
            XCH         @HL,A               ;

            JP          A_OUTP
N_CARRY     MOV         H,#0                ; if carry doesn't occur, delay time
            MOV         H,#0                ;
            MOV         H,#0                ;
            MOV         H,#0                ;
            MOV         H,#0                ;
            MOV         H,#0                ;
            MOV         H,#0                ;
            MOV         H,#0                ;
```

```
A_OUTP    MOV     H,#0
          MOV     L,#O_OUTP          ;  DAT0 ← DAT0 + O_OUTP
          ADDS    A,@HL              ;
          MOV     H,#4               ;
          MOV     L,#DAT0
          MOV     @HL,A              ;

          NOTI    A
          DECS    A                  ;  DAT2 ← complement of DAT0
          MOV     L,#DAT2
          MOVz    @HL,A

          MOV     L,#DAT1
          MOV     A,@HL
          NOTI    A                  ;  DAT3 ← complement of DAT1
          DECS    A
          MOV     L,#DAT3
          MOVZ    @HL,A

          MOV     L,#DAT1_0
          MOV     A,@HL
          NOTI    A
          DECS    A
          MOV     L,#DAT3_0          ;  DAT3_0 ← complement of DAT 1_0
          MOV     @HL,A
          JPL     TX

;************************ The END of KS51840/51850 DATA & CUSTOM CODE GEN. **********************
```

KS51840/51850 SIGNAL TRANSMISSION

Description

This program is for signal transmissions in SAMSUNG standard format. If one key is pressed, two frames are transmitted consecutively. The repeat pulse is transmitted until key-off. The frame interval is 60 ms.
Each frame consists of leader code, custom code, and data code:

— Leader code (high level for 4.5 ms and low level for 4.5 ms)

— 12-bit custom code

— 8-bit data code

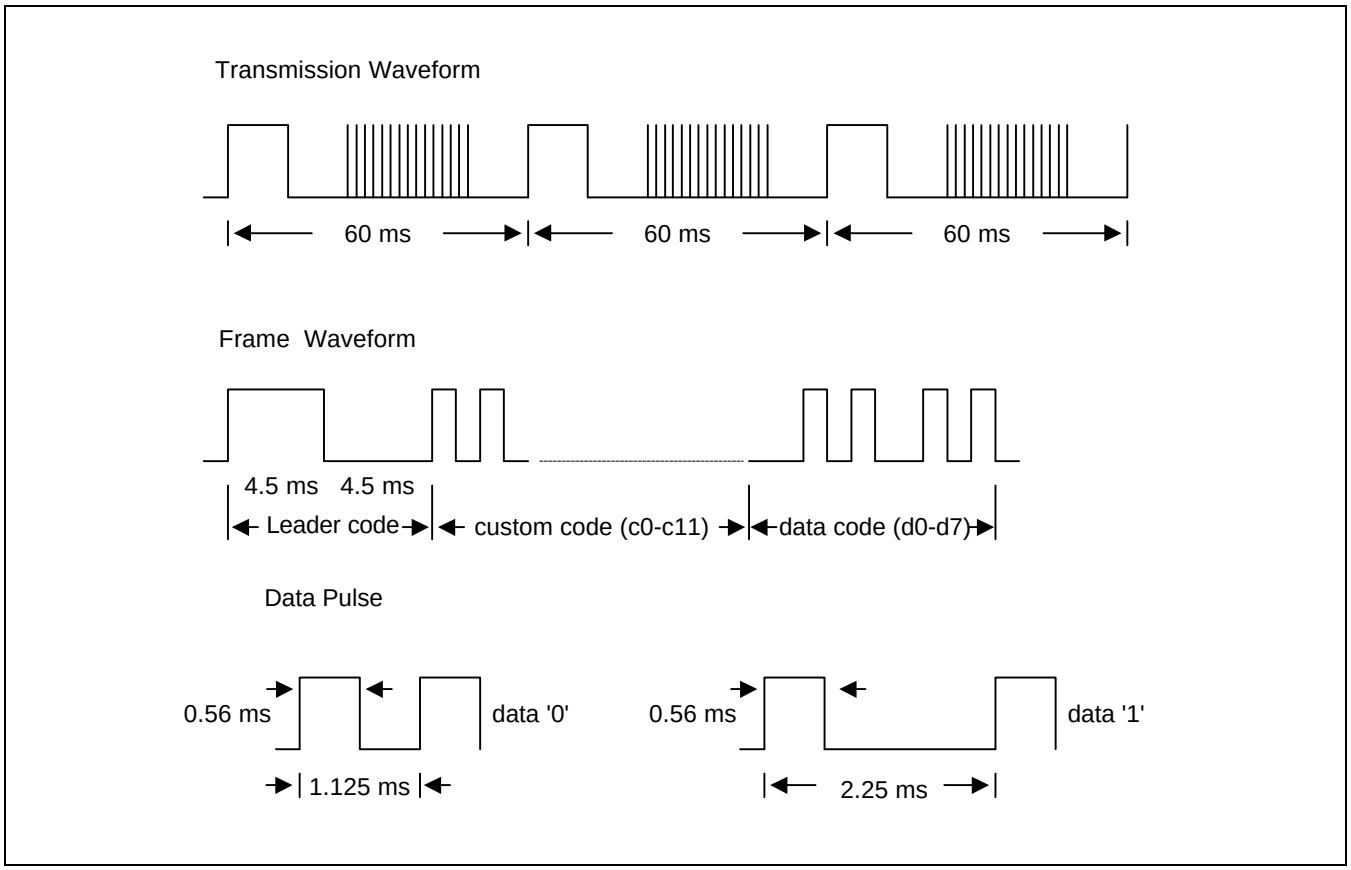


Figure 5-14. Transmission Waveforms

RAM Assignment

This part is the same as for keyscan and code generation.

KS51840/51850 Program Flowchart

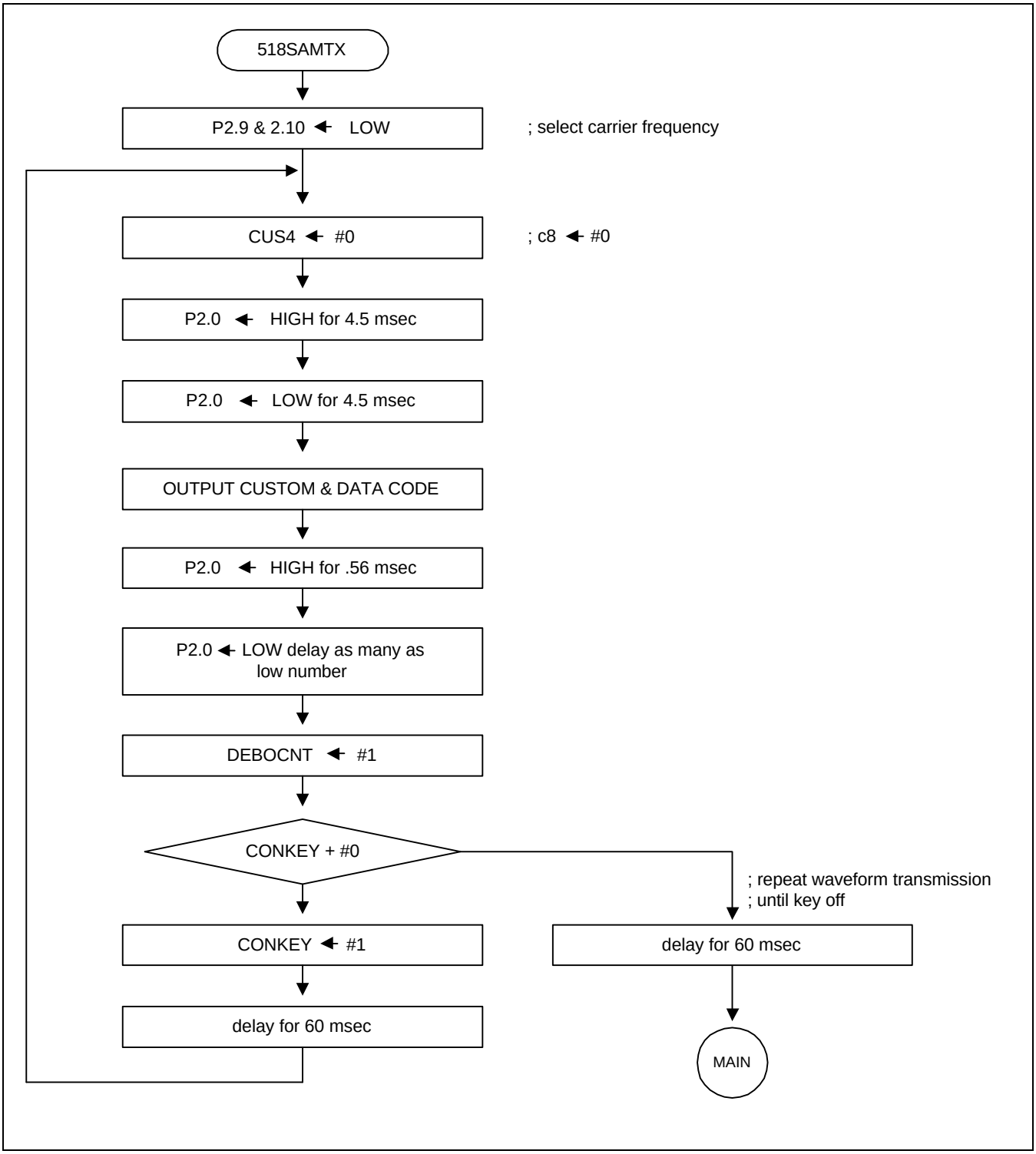


Figure 5-14. Ks51840/51850 Program Flowchart 5

```
;*********************************************
                ORG     0500H
                JPL     RESET
;*********************************************

TX              MOV     L,#9            ;   select farrier frequency
                CLRB    P2.(L)          ;   37.9 kHz, 1/3 duty
                MOV     L,#0AH          ;   clear P2.9 & 2.10
                CLRB    P2.(L)

SIGOUT          MOV     L,#CUS4         ;   custom code (c8-c11) ← #0
                MOB     @HL+,#0         ;   if device is KS51910, c8 ← #1

;;; output head pulse ----------------------------------------------

                MOV     L,#0            ;   high for delay time 4.5 msec
                SETB    P2.(L)
                CALLL   D4_5

                MOV     L,#0
                CLRB    P2.(L)          ;   low for delay time 4.5 msec
                CALLL   D4_5D

;;; output custom code (c0-c11) & data code (d0-d7)
                MOV     L,#CUS0         ;   custom code (c0-c3)
                CALLL   DATGEN
                MOV     L,#CUS1         ;   custom code (c4-c7)
                CALLL   DATGEN
                MOV     L,#CUS4         ;   custom code (c8-c11)
                CALLL   DATGEN
                MOV     L,#DAT0         ;   data code (d0-d3)
                CALLL   DATGEN
                MOV     L,#DAT1_0       ;   data code (d4-d7)
                CALLL   DATGEN
                MOV     L,#1
                DECS    L
                JP      .-1

                MOV     L,#0            ;   EOB (end of bit)
                SETB    P2.(L)          ;   high for .56msec
                CALLL   D_560F
                MOV     L,#0
                CLRB    P2.(L)
                JPL     LOWCHEK
```

```
;*********************************************

            ORG         0600H
            JPL         RESET
;*********************************************

;;; check low code of custom code & data code ------------------------------
LOWCHEK     MOV         L,#CUS0             ;   custom code (c0-c3)
            CALL        LCHEK
            MOV         L,#CUS1             ;   custom code (c4-c7)
            CALL        LCHEK
            MOV         L,#CUS4             ;   custom code (c8-c11)
            CALL        LCHEK
            MOV         L,#DAT0             ;   data code (d0-d3)
            CALL        LCHEK
            MOV         L,#DAT1_01          ;   the maximum value of upper bit is #3
            MOV         A,L                 ;   data code (d4-d7)
            CALL        LCHEK_1             ;   check from the second bit

;== notice ===============================================================

; If value of DAT1_0 is greater than #3, programmer must change instruction
; CALLLCHEK_1 to other instruction such as CALLCHECK_2 or CALL
; LCHEK. And you must check the fram interval (= 60 msec)
;=========================================================================

;;; re-setting debounce count to #1-------------------------------------
SETDBT      MOV         H,#0
            MOV         L,#DEBOCNT          ;   DEBOCNT ← #1
            MOV         @HL+,#1

;-------------------------------------------------------------------------------

;;; If conkey flag isn't '0', transmit repeat pulse
;;; otherwise, after setting, transmit again (two frames)
;-------------------------------------------------------------------------------

CONCHEK     MOV         H,#0
            MOV         L,#CONKEY           ;   CONKEY == #0?
            CPNZ        @HL                 ;   If CONKEY is #0, CONKEY ← #1
            JP          LJ_MAIN             ;   transmit frame again

            MOV         @HL+,#1
            CALLL       D4_5D               ;   time is 60 msec per frame
            CALLL       D2_25
            CALLL       D1_125
            MOV         L,#0CH
            MOV         H,#4
            MOV         H,#4
            DECS        L
            JP          .-3
            JPL         SIGOUT
```

```
;;; output repeat pulse ------------------------------------------------------
LJ_MAIN     CALL        D1_125
            JPL         MAIN

;;; output delay time as many as low numbers ---------------------------
LCHEK       MOV         A,L
            CPBT        @HL,3               ;   if @hl.3 is low, call d2_25d.
            JP          LCHEK_2
            CALLL       D2_25D

LCHEK_2     MOV         L,A
            CPBT        @HL.2               ;   if @hl.2 is low, call d2_25d.
            JP          LCHEK_1
            CALLL       D2_25D

LCHEK_1     MOV         L,A
            CPBT        @HL.1               ;   if @hl.1 is low, call d2_25d.
            JP          LCHEK_0
            CALLL       D2_25D

LCHEK_0     MOV         L,A
            CPBT        @HL.0               ;   if @hl. 0 is low, call d2_25d.
            JP          LCHEK_R
            CALLL       D2_25D

LCHEK_R RET
;
;
;
;
;
;
;*********************************************

            ORG         0700H
            JPL         RESET
;*********************************************

;; custom code & data code generation ----------------------------------
DATGEN      MOV         A,L
            CALL        D_560               ;   high for .56 msec
            CPBT        @HL.0               ;   if @hl.0 is high, low for 2.25 msec.
            CALL        D2_25               ;   otherwise, low for 1.125 msec.
            CALL        D1_125
```

```
                CALL    D_560           ;   high for .56 msec
                CPBT    @HL.1           ;   if @hl.1 is high, low for 2.25 msec.
                CALL    D2_25           ;   otherwise, low for 1.125 msec.
                CALL    D1_125

                CALL    D_560           ;   high for .56 msec
                CPBT    @HL.2           ;   if @hl.2 is high, low for 2.25 msec.
                CALL    D2_25           ;   otherwise, low for 1.125 msec.
                CALL    D1_125

                CALL    D_560           ;   high for .56 msec
                CPBT    @HL.3           ;   if @hl.3 is high, low for 2.25 msec.
                CALL    D2_25           ;   otherwise, low for 1.125 msec.
                CALL    D1_125D
                RET

;;; delay time subroutine by programming ------------------------------------
D_560           MOV     L,#0
                SETB    P2,(L)
;
                MOV     L,#0CH
                MOV     H,#4
                DECS    L
                JP      .-2
                MOV     H,#4
;
                MOV     L,#0
                CLRB    P2.(L)
                MOV     L,A
                RET

D4_5D           MOV     L,#02H          ;   delay time 4.5 msec
                JP      .+2
D4_5            MOV     L,#06H
                DECS    L
                JP      -1

                MOV     L,#05H
                CLR     A
                ADDS    A,#0BH
                MOV     H,#4
D_A             DECS    A
                JP      .-2
                DECS    L
                JP      .-6
```

```
D2_25D      MOV         L,#0EH
            JP          .+2
D2_25       MOV         L,#0FH
            MOV         H,#4
            DECS        L
            JP          .-2

D1_125      MOV         L,#0AH
            JP          .+6
D1_125D     MOV         L,#08H
            JP          .+8
D_560F      MOV         L,#0BH                  ;   delay time .56 msec (for EOB)
            MOV         H,#4
            MOV         H,#4
            DECS        L
            JP          .-2
            RET
;
;
            org         0800h
            jpl         reset
            org         0900h
            jpl         reset
            org         oaooh
            jpl         reset
            org         obooh
            jpl         reset
            org         ocooh
            jpl         reset
            org         odooh
            jpl         reset
            org         oeooh
            jpl         reset
;
;****************** The END of KS51840/51850 SAMSUNG FORMAT TX. **************************
```

```
;;; select output pin by one and one ------------------------------------
STROBE    MOV     L, #N_OUTP
          CPBT    @HL.2             ;  If N_OUTP.2 is set, go to parall (parallel port)
          JP      PARALL            ;  otherwise, go to serial (serial port)
SERIAL    MOV     L, @HL            ;
          INCS    L                 ;
          SETB    P2.(L)
```

```
;
;
;////////////////////////////////////////////////////////////////////
;////////////////////////////////////////////////////////////////////
;/// If you want to use the following files, run only after    ////
;/// removing the comments(;) in the first column.   You can   ////
;/// modify these files, but be sure to verify the ROM address ////
;/// of transmit files.   Also you must  reset debounce time   ////
;/// (=DEBOCNT) by a frame size. DEBOCNT is located in the     ////
;/// transmit file.                                            ////
;////////////////////////////////////////////////////////////////////
;////////////////////////////////////////////////////////////////////
;
;*****************************************************************
;1. SELECT .DEF FILE
;*****************************************************************
;(1)
        CHIP C:\SMDSII\DATA\51840.DEF
;*****************************************************************
;2. SELECT EQU FILE
;   ===> RAM assignment
;*****************************************************************
;(1)
        INCLUDE C:\SMDSII\HEAD\518EQU.H
;*****************************************************************
;3. SELECT KEYSCAN FILE
;   ===> double keys check in key matrix(64-key)
;*****************************************************************
;(1)
        INCLUDE C:\SMDSII\SRC\518KSCAN.SRC
;*****************************************************************
;4. SELECT CUSTOM CODE & DATA CODE PRODUCTION FILE
;   ===> when a key press, code prodouction
;*****************************************************************
;(1)
        INCLUDE C:\SMDSII\SRC\518CODE.SRC
;*****************************************************************
;5. SELECT TRANSMISSION FILE
;   ===> remocon tx. waveform.
;*****************************************************************
;;  ==> samsung tx. format
        INCLUDE C:\SMDSII\SRC\518SAMTX.SRC
;;  ==> sony tx. format
;        INCLUDE C:\SMDSII\SRC\518SNYTX.SRC
;;  ==> rc-5 tx. format
;        INCLUDE C:\SMDSII\SRC\518RC5TX.SRC
;;  ==> toshiba tx. format
;        INCLUDE C:\SMDSII\SRC\518TOSTX.SRC
;;  ==> rca tx. format
;        INCLUDE C:\SMDSII\SRC\518RCATX.SRC
;;  ==> panasonic tx. format
;        INCLUDE C:\SMDSII\SRC\518PANTX.SRC
;;  ==> mitsubish tx. format
;        INCLUDE C:\SMDSII\SRC\518MITTX.SRC
;;  ==> nec(custom code 8-bit) tx. format
;        INCLUDE C:\SMDSII\SRC\518NECTX.SRC
;;  ==> nec(custom code 16-bit)tx. format
;        INCLUDE C:\SMDSII\SRC\518_16TX.SRC

        END
```

```
;;for SONY FORMAT
;;
;;If a key is pressed, after a frame is transmitted,repeat pulse is
;;transmitted until key off.
;;the frame interval is 45 msec.
;;
;;*** a frame waveform ****
;;  ______    ____    __    ____    __       __    __        __    __
;;|      |  |    |  |  |  |    |  |  |     |  |  |  |      |  |  |  |
;;|      |  |    |  |  |  |    |  |  |     |  |  |  |      |  |  |  |
;;|      |__|    |__|  |__|    |__|  |_..._|  |__|  |_.....|  |__|  |_
;; 2.4ms  <----- data code (d0 - d6) ------><-- data code(c0 - c4)--->EOB
;;
;;*** data pulse ***
;;              _____              _____
;;   data '1'   |     |          data '0'  |     |
;;              |     |                    |     |
;;         _____|     |               _____|     |
;;         0.6ms  1.2ms               0.6ms  0.6ms
;;
;*******************************
        ORG     0500H
        JPL     RESET
;*******************************
TX      MOV     L,#9            ;select carrier frequency
        CLRB    P2.(L)          ;40kHz -> 37.9 KHz
        MOV     L,#0AH          ;clear p2.9 & p2.10
        CLRB    P2.(L)

;;;output head pulse ------------------------------------------------
        MOV     L,#0            ;high for delay time 2.4msec
        SETB    P2.(L)
        CALLL   D2_4

;;;output data code (= 7bits) ---------------------------------------
        MOV     L,#DAT0         ;data code (d0-d3)
        CALLL   DATGEN
        MOV     L,#DAT1_0       ;data code (d4-d6)
        CALLL   DATGEN2

;;output custom code (= 5bits) --------------------------------------
        MOV     L,#CUS0         ;custom code (c0 ~c3)
        CALLL   DATGEN
        MOV     L,#CUS1         ;custom code (c4)
        CALLL   DATGEN0
        MOV     L,#1            ;finish code delay time
        DECS    L
        JP      .-1

        MOV     L,#0
        CLRB    P2.(L)

;;;check low bit of data code & custom code -------------------------
        MOV     L,#DAT0         ;data code (d0 ~d3)
        CALLL   LCHEK
        MOV     L,#DAT1_0       ;data code (d4 ~d6)
        MOV     A,L
        CALLL   LCHEK_2
        MOV     L,#CUS0         ;custom code(c0-c3)
        CALLL   LCHEK
        MOV     L,#CUS1         ;custom code(c4)
        MOV     A,L
        CALLL   LCHEK_0

        JPL     S_DBCNT
;
;
;
;
;
;*******************************
        ORG     0600H
        JPL     RESET
;*******************************
;;custom code & data code generation ------------------------------
DATGEN  MOV     A,L
        CALL    D0_6                    ;high for 0.6 msec
        CPBT    @HL.0                    ;if @hl.0 is high, low for 1.2msec
        CALL    D1_2                    ;otherwise, low for 0.6msec
        CALL    D0_6H

        CALL    D0_6                    ;high for 0.6msec
        CPBT    @HL.1                    ;if @hl.1 is high, low for 1.2msec
        CALL    D1_2                     ;otherwise, low for 0.6msec
        CALL    D0_6H

        CALL    D0_6                    ;high for 0.6 msec
        CPBT    @HL.2                    ;if @hl.2 is high, low for 1.2msec
        CALL    D1_2                    ;otherwise, low for 0.6msec
        CALL    D0_6H

        CALL    D0_6                    ;high for 0.6msec
        CPBT    @HL.3                    ;if @hl.3 is high, low for 1.2msec
        CALL    D1_2                    ;otherwise, low for 0.6msec
        CALL    D0_6HD
        RET


DATGEN2 MOV     A,L
        CALL    D0_6                    ;check @hl.0 - @hl.2
        CPBT    @HL.0
        CALL    D1_2
        CALL    D0_6H

        CALL    D0_6
        CPBT    @HL.1
        CALL    D1_2
        CALL    D0_6H

        CALL    D0_6
        CPBT    @HL.2
        CALL    D1_2
        CALL    D0_6HD
        RET


DATGEN2 MOV     A,L
        CALL    D0_6                    ;check only @hl.0
        CPBT    @HL.0
        CALL    D1_2
        CALL    D0_6HD
        RET

;;;delay 0.6msec
D0_6    MOV     L,#0
        CLRB    P2.(L)
;
        MOV     L,#0DH
        MOV     H,#4
        DECS    L
```

```
131           JP        .-2
132  ;
133           MOV       L,#0
134           SETB      P2.(L)
135           MOV       L,A
136           RET
137
138  ;;delay 1.2msec
139  D1_2L    MOV       L,#0
140           JP        .+2
141
142  D1_2     MOV       L,#01H
143           MOV       H,#4
144           DECS      L
145           JP        .-2
146
147  ;;;delay 0.6msec for EOB
148  D0_6H    MOV       L,#0BH
149           JP        .+3
150  D0_6HD   MOV       L,#09H
151           MOV       H,#4
152           MOV       H,#4
153           DECS      L
154           JP        .-2
155           RET
156  ;
157  ;
158  ;
159  ;
160  ;
161  ;*****************************
162           ORG       0700H
163           JPL       RESET
164  ;*****************************
165  ;;;a frame for 45msec  ------------------------------------------
166  S_DBCNT  CALL      D45
167
168
169  ;;;re_setting debounce count to #3 ---------------------------------
170           MOV       H,#0
171           MOV       L,#DEBOCNT          ;re-setting debounce time
172           MOV       @HL+,#3
173           JPL       MAIN
174
175
176
177  LCHEK    MOV       A,L
178           CPBT      @HL.3               ;if @hl.3 is low, call d1_2l
179           JP        LCHEK_2
180           CALLL     D1_2L
181
182  LCHEK_2  MOV       L,A
183           CPBT      @HL.2               ;if @hl.2 is low, call d1_2l
184           JP        LCHEK_1
185           CALLL     D1_2L
186
187  LCHEK_1  MOV       L,A
188           CPBT      @HL.1               ;if @hl.1 is low, call d1_2l
189           JP        LCHEK_0
190           CALLL     D1_2L
191
192  LCHEK_0  MOV       L,A
193           CPBT      @HL.0               ;if @hl.0 is low, call d1_2l
194           JP        LCHEK_R
195           CALLL     D1_2L
196  LCHEK_R RET
197
198
199  ;;;for a frame delay time ------------------------------------------
200  D45      MOV       L,#0EH
201           MOV       H,#4
202           MOV       H,#4
203           MOV       H,#4
204           MOV       H,#4
205           DECS      L
206           JP        .-5
207
208
209           MOV       L,#0FH
210           JP        .+2
211
212  ;;leader pulse for delay 2.4msec ----------------------------------
213  D2_4     MOV       L,#0DH
214           CLR       A
215           ADDS      A,#3
216           DECS      A
217           JP        .-1
218           DECS      L
219           JP        .-5
220           MOV       H,#4
221           MOV       H,#4
222           MOV       H,#4
223           MOV       H,#4
224           RET
225
226  ;-----------------------------------------------------------------
227           org       0800h
228           jpl       reset
229           org       0900h
230           jpl       reset
231           org       0a00h
232           jpl       reset
233           org       0b00h
234           jpl       reset
235           org       0c00h
236           jpl       reset
237           org       0d00h
238           jpl       reset
239           org       0e00h
240           jpl       reset
241
242
243
```

```
;;for RC-5 FORMAT
;
;If a key is pressed, after a frame is transmitted, repeat pulse is
;transmitted until key off, and every time a key is released, control
;bit is converted.   The frame interval is 64 bits (= 113.8 msec).
;
;*** a frame waveform ****
;   __   ____   __   __    ____    _____   __   __   __    __
;  |  |  |    | |  | |  |  |    |  |    | |  | |  | |  |  |  |
;  ^  |  ^    v |  v |  v  ^    v  ^    v |  v |  v |  v  ^  v
;__|  |__|    |__|  |__|  |____|  |____|  |__|  |__|  |__|  |____|  |
; start bit |    |         custom code      |           data code       |
;<--2bits-->|1bit|<-------   5bits   ------>|<-----------  6bits ------->|
;         control bit
;<----------------------------14 bits ---------------------------------->|
;
;
;*** data pulse ***
;          _____                          _____
;         |                                    |
;         ^                                    v
;     ____|       data '1'                    |_____ data '0'
;    |            |                         |          |
;    |<--1.778ms-->|                         |<--1.778ms-->|
;     = 1bit time
;    < rising edge >                         < falling edge >
;*********************************
;
        ORG         0500H
        JPL         RESET
;*********************************
TX      MOV     L,#9            ;select carrier frequency
        CLRB    P2.(L)          ;(1/3 duty, fosc/12)
        MOV     L,#0AH          ;37.9 kHz
        CLRB    P2.(L)

;;;re-setting debounce count to #0fh -------------------------------
DEBOSET MOV     H,#0
        MOV     L,#DEBOCNT
        MOV     @HL+,#0FH

;-----------------------------------------------------------------
;;;if continuous flag is reset(= '0'),
;;;continuous flag(=conkey) is set to '1' and
;;;control flag is converted.(namely, every time key release)
;-----------------------------------------------------------------
        MOV     H,#0
        MOV     L,#CONKEY
        CPBT    @HL.0
        JP      STRDLY

        MOV     @HL+,#1         ;setting conkey flag to '1'
        MOV     L,#CTLFLG       ;convert control flag (= CTLFLG)
        CPBT    @HL.0
        JP      .+3
C_LOW   MOV     @HL+,#1         ;if CTLFLG is low, CTLFLG <- #1
        JP      .+3             ;otherwise, CTLFLG <- #0
C_HIGH  MOV     @HL+,#0
        MOV     H,#4            ;delay time
        JP      START

;;;start delay time (generate at continuous key) --------------------
STRDLY  MOV     L,#2
        DECS    L
        JP      .-1
        MOV     H,#4

;;;output start bit (All 2bits are always high) & control bit -------
START   CALLL   DAT_1           ;the first start bit
        MOV     L,#0
        DECS    L
        JP      .-1
        MOV     H,#4
        JPL     CTLOUT

;; delay for a frame ----------------------------------------------
D113_8  MOV     L,#05H
        MOV     H,#4
        MOV     H,#4
        MOV     H,#4
        MOV     H,#4
        DECS    L
        JP      .-5
        RET
;
;
;
;
;
;*********************************
        ORG     0600H
        JPL     RESET
;*********************************
CTLOUT  CALLL   DAT_1           ;the second start bit
        MOV     H,#0
        MOV     L,#CTLFLG       ;control bit
        CALLL   DATGEN0

;;;;output custom code & data code --------------------------------
        MOV     L,#CUS0         ;custom code (c0)
        CALLL   DATGEN0

        MOV     L,#CUS0         ;custom code (c1)
        CALLL   DATGEN1

        MOV     L,#CUS0         ;custom code (c2)
        CALLL   DATGEN2

        MOV     L,#CUS0         ;custom code (c3)
        CALLL   DATGEN3

        MOV     L,#CUS1         ;custom code (c4)
        CALLL   DATGEN0

        MOV     L,#DAT0         ;data code (d0)
        CALLL   DATGEN0

        MOV     L,#DAT0         ;data code (d1)
        CALLL   DATGEN1

        MOV     L,#DAT0         ;data code (d2)
        CALLL   DATGEN2

        MOV     L,#DAT0         ;data code (d3)
        CALLL   DATGEN3

        MOV     L,#DAT1_0       ;data code (d4)
        CALLL   DATGEN0

        MOV     L,#DAT1_0       ;data code (d5)
        CALLL   DATGEN1
```

```
          MOV       L,#2
          DECS      L
          JP        .-1

          MOV       L,#0
          CLRB      P2.(L)



;;;per one frame delay time (= 64bits) -----------------------------
          MOV       L,#07H
A_F       CLR       A
          ADDS      A,#0FH
S_F       MOV       H,#4
          MOV       H,#4
          MOV       H,#4
          MOV       H,#4
          MOV       H,#4
          DECS      A
          JP        S_F
          DECS      L
          JP        A_F

          CALLL     D113_8

          JPL       MAIN




;**********************************
        ORG     0700H
        JPL     RESET
;**********************************
;;; each bit data (= 4bits) generation -----------------------------
DATGEN0   CPBT      @HL.0           ;if @hl.0 is high, rising edge.
          JP        .+3             ;otherwise, falling edge.
          CALL      DAT_0           ;falling edge
          JP        .+2
          CALL      DAT_1           ;rising edge
          RET

DATGEN1   CPBT      @HL.1           ;if @hl.1 is high, rising edge.
          JP        .+3             ;otherwise, falling edge.
          CALL      DAT_0
          JP        .+2
          CALL      DAT_1
          RET

DATGEN2   CPBT      @HL.2           ;if @hl.2 is high, rising edge.
          JP        .+3             ;otherwise, falling edge.
          CALL      DAT_0

          JP        .+2
          CALL      DAT_1
          RET

DATGEN3 CPBT    @HL.3           ;if @hl.3 is high, rising edge.
        JP      .+3             ;otherwise, falling edge.
        CALL    DAT_0
        JP      .+2
        CALL    DAT_1
        RET



;;;the falling edge of low data subroutine ---------------------------
DAT_0   MOV     L,#0
        SETB    P2.(L)

        MOV     L,#0FH          ;high for 0.889 msec
        MOV     H,#4
        MOV     H,#4
        DECS    L
        JP      .-3

        MOV     L,#0
        CLRB    P2.(L)

        MOV     L,#0CH          ;low for 0.889 msec
        MOV     H,#4
        MOV     H,#4
        DECS    L
        JP      .-3
        MOV     H,#4
        MOV     H,#4
        MOV     H,#4
        RET

;;;the rising edge of high data subroutine --------------------------
DAT_1   MOV     L,#0
        CLRB    P2.(L)

        MOV     L,#0FH          ;low for 0.889 msec
        MOV     H,#4
        MOV     H,#4
        DECS    L
        JP      .-3

        MOV     L,#0
        SETB    P2.(L)

        MOV     L,#0DH          ;high for 0.889 msec
        MOV     H,#4
        MOV     H,#4
        DECS    L
        JP      .-3
        RET

;-----------------------------------------------------------------
        org     0800h
        jpl     reset
        org     0900h
        jpl     reset
        org     0a00h
        jpl     reset
        org     0b00h
        jpl     reset
        org     0c00h
        jpl     reset
        org     0d00h
        jpl     reset
        org     0e00h
        jpl     reset
```

```
;;for TOSHIBA FORMAT
;;
;;If a key is pressed, after a frame is transmitted, continuous pulse
;;is transmitted until key off
;;The frame interval is 108 msec or 126 msec according as custom code.
;;Namely, if the high count of custom code (c0-c7) is greater then #4,
;;delay time is 126msec, otherwise, delay time is 108 msec. But,
;;continuous pulse is only 108msec. A frame consists of leader code,
;;custom code and data code.
;;-- leader code consists of high for 4.5 msec and low for 4.5 msec
;;-- custom code consists of 16-bit (= custom code (c0-c7) & custom
;;   code (c0-c7)).
;;-- data code consists of 16-bit (= data code (d0-d7) & complement
;;   of data code)
;;
;;*** transmission waveform ***
;; ____                                ____                ____
;;|    |    ||||||||||                |    |    ||       |    |    ||
;;|    |    ||||||||||                |    |    ||       |    |    ||
;;|    |____||||||||||________________|    |____||_....._|    |____||__
;; 4.5m 4.5m                          4.5m 4.5m
;;|<------- 108msec or 124msec ------>|<---- 108msec --->|
;;
;;*** a frame waveform ***
;; _____       __    __
;;|     |     |  |  |  |
;;|     |     |  |  |  |
;;|     |_____|  |__|  |__
;; 4.5m 4.5m  |     |EOB(end of bit)
;;            |<--->|
;;                  '1' or '0' according as complement of custom code 'c0'
;;                  namely, if 'c0' of custom code is '1', the value is '0'
;;                  otherwise, the value is '1'.
;;
;; *** data pulse ***
;;                    ___                          ___
;;                   |   |                        |   |
;;                -> |   | <- data '0'         -> |   | <- data '1'
;;                   |   |____                    |   |_________
;;                    0.56ms                       0.56ms
;;                   |<------>|                  |<---------->|
;;                    1.125msec                       2.25msec
;;
;*******************************
        ORG     0500H
        JPL     RESET
;*******************************
TX      MOV     L,#09H                  ;select carrier frequency
        CLRB    P2.(L)                  ;37.9 kHz, 1/3 duty
        MOV     L,#0AH
        CLRB    P2.(L)

;;If CONKEY is '1', transmit continuous pulse ------------------------
        MOV     H,#0
        MOV     L,#CONKEY               ;check continuous key
        CPNZ    @HL
        JP      SIGCON
        JP      SIGOUT

;;;re-setting debounce count to #0H ----------------------------------
;;;output continuous pulse per 108 msec ------------------------------
SIGCON  MOV     L,#DEBOCNT              ;setting debounce time again
        MOV     @HL+,#0H

;;output leader pulse -----------------------------------------------
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D4_5                    ;high for delay 4.5 msec
        MOV     L,#0                    ;low for delay 4.5 msec
        CLRB    P2.(L)
        CALLL   D4_5
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D_560F

;;;The value is determined by the complement of custom code (=c0) -----
        MOV     L,#0
        CLRB    P2.(L)
        MOV     L,#CUS0

        CPBT    @HL.0
        JP      .+3
        CALLL   D2_25
        CALLL   D1_125

        MOV     L,#0
        SETB    P2.(L)
        CALLL   D_560F                  ;EOB
        MOV     L,#0
        CLRB    P2.(L)

;;;a frame is 108 msec in continuous pulse ---------------------------
J_SIGC  MOV     L,#CUS0
        CPBT    @HL.0
        JP      .+2
        JP      .+3
        CALLL   D2_25D

        MOV     L,#0FH
        CALLL   D_A
        MOV     L,#04H
        CALLL   D_A
        MOV     L,#03H
        DECS    L
        JP      .-1
        JPL     MAIN}

;
;
;
;
;
;*******************************
        ORG     0600H
        JPL     RESET
;*******************************
;;; -------------------------------------------------------------------
SIGOUT  MOV     @HL+,#1                 ;CONKEY <- #1

;;;output leader pulse -----------------------------------------------
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D4_5                    ;high for delay time 4.5 msec
        MOV     L,#0
        CLRB    P2.(L)
        CALLL   D4_5L                   ;low for delay time 4.5 msec


;;;output custom code ------------------------------------------------
        MOV     L,#CUS0                 ;output custom code (c0-c3)
```

```
131           CALLL     CDGEN
132           MOV       L,#CUS1             ;output custom code (c4-c7)
133           CALLL     CDGEN
134           MOV       L,#CUS0              ;repeat above custom code
135           CALLL     CDGEN
136           MOV       L,#CUS1
137           CALLL     CDGEN
138
139  ;;;output data code ------------------------------------------------
140           MOV       L,#DAT0              ;output data code (d0-d3)
141           CALLL     CDGEN
142           MOV       L,#DAT1             ;output data code (d4-d7)
143           CALLL     CDGEN
144           MOV       L,#DAT2              ;complement of data code (d0-d3)
145           CALLL     CDGEN
146           MOV       L,#DAT3             ;complement of data code (d4-d7)
147           CALLL     CDGEN
148           MOV       L,#1
149           DECS      L
150           JP        .-1
151           MOV       H,#4
152
153           MOV       L,#0
154           SETB      P2.(L)               ;high for delay 560 usec
155           CALLL     D_560F
156           MOV       L,#0
157           CLRB      P2.(L)
158
159  ;;;check the high value of custom code (c0-c7)-----------------------
160  ;;;
161  H_CHEK    CLR       A
162           MOV       L,#CUS0
163           CALLL     H_0
164           MOV       L,#CUS1
165           CALLL     H_0
166
167
168           JPL       FDLY
169  ;
170  ;
171  ;
172  ;
173  ;
174  ;*******************************
175           ORG     0700H
176           JPL     RESET
177  ;*******************************
178  ;-----------------------------------------------------------------
179  ;;If A is greater than #4, call delay 126 msec
180  ;;;otherwise, call delay 108msec
181  ;-----------------------------------------------------------------
182  FDLY      MOV       H,#0
183           MOV       L,#DEBOCNT
184           ADDS      A,#0BH
185           JP        J_126
186
187           MOV       @HL+,#6             ;setting debounce time
188           MOV       L,#3                ;for 108 msec
189           JP        J_E
190
191  J_126     MOV       @HL+,#0AH           ;setting debounce time
192           MOV       L,#3
193           DECS      L
194           JP        .-1
195           MOV       L,#0
196  J_E       CALL      D_A
197
198
199  ;output delay as many as the low of custom code --------------------
200  L_DELAY   MOV       L,#CUS0             ;custom code (c0-c3)
201           CALLL     LOWDLY
202           MOV       L,#CUS1
203           CALLL     LOWDLY              ;custom code (c4-c7)
204           MOV       L,#CUS0             ;custom code (c0-c3)
205           CALLL     LOWDLY
206           MOV       L,#CUS1
207           CALLL     LOWDLY              ;custom code (c4-c7)
208
209           JPL       MAIN
210
211  H_0       CPBT      @HL.0               ;count up high data
212           JP        .+2
213           JP        H_1
214           INCS      A                   ;if data is high, increase value of A
215
216  H_1       CPBT      @HL.1               ;count up high data
217           JP        .+2
218           JP        H_2
219           INCS      A
220
221  H_2       CPBT      @HL.2               ;count up high data
222           JP        .+2
223           JP        H_3
224           INCS      A
225
226  H_3       CPBT      @HL.3               ;count up high data
227           JP        .+2
228           JP        H_R
229           INCS      A
230  H_R       RET
231
232
233
234  ;;;delay time subroutine by programming ----------------------
235  D4_5      MOV       L,#1                ;delay 4.5msec
236           DECS      L
237           JP        .-1
238           MOV       H,#4
239
240  D4_5L     MOV       L,#5
241  D_A       CLR       A
242           ADDS      A,#0FH
243           MOV       H,#4
244           DECS      A
245           JP        .-2
246           DECS      L
247           JP        .-6
248
249           MOV       L,#07H
250           DECS      L
251           JP        .-1
252
253           RET
254
255
256  ;
257  ;
258  ;
259  ;
260  ;
```

```
;*****************************
        ORG     0800H
        JPL     RESET
;*****************************
;;;generate custom & data code ----------------------------
CDGEN     MOV     A,L
          CALL    D_560               ;high for 0.56msec
          CPBT    @HL.0               ;if @HL.0 is high, low for 2.25msec
          CALL    D2_25               ;otherwise, low for 1.125msec
          CALL    D1_125

          CALL    D_560               ;high for 0.56msec
          CPBT    @HL.1               ;if @HL.1 is high, low for 2.25msec
          CALL    D2_25               ;otherwise, low for 1.125msec
          CALL    D1_125

          CALL    D_560               ;high for 0.56msec
          CPBT    @HL.2               ;if @HL.2 is high, low for 2.25msec
          CALL    D2_25               ;otherwise, low for 1.125msec
          CALL    D1_125

          CALL    D_560               ;high for 0.56msec
          CPBT    @HL.3               ;if @HL.3 is high, low for 2.25msec
          CALL    D2_25               ;otherwise, low for 1.125msec
          CALL    D1_125F
          RET


D2_25D    MOV     L,#0EH
          JP      .+2

D2_25     MOV     L,#0FH
          DECS    L
          JP      .-1
          JP      .+6
D1_125    MOV     L,#0AH
          JP      .+6
D1_125F   MOV     L,#08H
          JP      .+4
D_560F    MOV     L,#0BH
          MOV     H,#4
          MOV     H,#4
          DECS    L
          JP      .-2
          RET


D_560     MOV       L,#0
          SETB      P2.(L)
;
          MOV       L,#0CH
          MOV       H,#4
          DECS      L
          JP        .-2
          MOV       H,#4
;
          MOV       L,#0
          CLRB      P2.(L)
          MOV       L,A
          RET

;;;for low delay subroutine ---------------------------------------
LOWDLY    MOV       A,L
          CPBT      @HL.3               ;if @HL.3 is low, call D2_25D
          JP        LOW_2
          CALL      D2_25D

LOW_2     MOV       L,A
          CPBT      @HL.2               ;if @HL.2 is low, call D2_25D
          JP        LOW_1
          CALL      D2_25D

LOW_1     MOV       L,A
          CPBT      @HL.1               ;if @HL.1 is low, call D2_25D
          JP        LOW_0
          CALL      D2_25D

LOW_0     MOV       L,A
          CPBT      @HL.0               ;if @HL.0 is low, call D2_25D
          JP        LOW_R
          CALL      D2_25D
LOW_R     RET


;-------------------------------------------------------------------
          org       0900h
          jpl       reset
          org       0a00h
          jpl       reset
          org       0b00h
          jpl       reset
          org       0c00h
          jpl       reset
          org       0d00h
          jpl       reset
          org       0e00h
          jpl       reset
```

```
;;for RCA FORMAT
;;
;;If a key is pressed, a frame is transmitted.   And repeat pulse is
;;transmitted until key off. The frame gap is 8 msec.  The head pulse
;;of the first frame is high for 20 msec and low for 4 msec.
;;But, the head pulse of continuous frame is high for 4 msec and low
;;for 4 msec.
;;*** transmission waveform ***
;; ______                ___                 ___
;;|      |   |||||||||   |   |   |||||||||   |   |   |||||||||
;;|      |   |||||||||   |   |   |||||||||   |   |   |||||||||
;;|      |___|||||||||___|   |___|||||||||___|   |___|||||||||
;;head pulse          |   |   |       |   |
;;<-20ms->4msec       8ms 4ms 4ms         8ms
;;
;;*** a frame waveform ***
;; ____
;;|    |    ||  ||  ||  || ||  ||  ||     ||  || || || ||  || || ||    || | |
;;|    |    ||  ||  ||  || ||  ||  ||     ||  || || || ||  || || ||    || | |
;;|    |____||__||__||__||_||__||__||.....||__||_||_||_||__||_||_||...|| | |
;;|         |                   |  _________  |  _________    | |
;;|    |    | custom code  | data code(8-bit) | custom code|  data code   | |
;; 4ms   4ms        (4-bit)                                              1ms
;;
;; *** data pulse ***
;;       data '1'  ___                  data '0' ___
;;                |   |                         |   |
;;                |   |                         |   |
;;                |   |_____                    |   |___________
;;               0.5ms  1ms                    0.5ms    2ms
;*******************************
        ORG     0500H
        JPL     RESET
;*******************************
TX      MOV     L,#9                    ;select carrier frequency
        SETB    P2.(L)                  ;56.9 kHz, 1/2 duty
        MOV     L,#0AH
        CLRB    P2.(L)

        MOV     H,#0
        MOV     L,#CONKEY               ;check continuous key
        CPNZ    @HL
        JP      SIGCON

;;;the first frame ------------------------------------------------
;;;output head pulse (= 20msec) -----------------------------------
SIGOUT  MOV     L,#CONKEY               ;setting conkey flag to '1'
        MOV     @HL+,#1
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D20                     ;high for delay 20 msec
        JP      .+5

;;;the repeat pulse -----------------------------------------------
;;;output head pulse (= 4msec) ------------------------------------
SIGCON  MOV     L,#0
        SETB    P2.(L)
        CALLL   D4                      ;high for delay 4 msec

        MOV     L,#0                    ;low for delay 4 msec
        CLRB    P2.(L)
        CALLL   D4_L

;;;output custom code ---------------------------------------------
        MOV     H,#4
        MOV     L,#CUS0                 ;custom code 4bits
        CALLL   DATGEN

;;;output data code -----------------------------------------------
        MOV     L,#DAT0                 ;data code 8bits
        CALLL   DATGEN
        MOV     L,#DAT1_0
        CALLL   DATGEN
;;;      __________
;;;output custom code ---------------------------------------------
        MOV     L,#CUS2                 ;complement of custom code
        CALLL   DATGEN
;;;      _________
;;;output data code -----------------------------------------------
        MOV     L,#DAT2                 ;complement of data code
        CALLL   DATGEN
        MOV     L,#DAT3_0
        CALLL   DATGEN
        MOV     L,#1
        DECS    L
        JP      .-1

;;;output EOB -----------------------------------------------------
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D1                      ;high for delay 1 msec
        MOV     L,#1
        DECS    L
        JP      .-1

        MOV     L,#0
        CLRB    P2.(L)

;;;re-setting debounce count to #1 --------------------------------
        MOV     H,#0
        MOV     L,#DEBOCNT
        MOV     @HL+,#1

;;;the frame gap is 8msec  ----------------------------------------
        CALLL   D8
        JPL     MAIN
;
;
;
;
;
;*******************************
        ORG     0600H
        JPL     RESET
;*******************************
;;;-----------------------------------------------------------------
;;;data & custom code generation
;;;high for .5 msec through p2.0
;;;If the value is '1', low for 1 msec
;;;otherwise, low for 2 msec.
;;;-----------------------------------------------------------------
DATGEN  MOV     A,L
        CALL    D0_5
        CPBT    @HL.0
        JP      .+2
        CALL    D2
        CALL    D1

        CALL    D0_5
        CPBT    @HL.1
```

```
131         JP      .+2
132         CALL    D2
133         CALL    D1
134
135         CALL    D0_5
136         CPBT    @HL.2
137         JP      .+2
138         CALL    D2
139         CALL    D1
140
141         CALL    D0_5
142         CPBT    @HL.3
143         JP      .+2
144         CALL    D2
145         CALL    D1_F
146         RET
147
148
149 D0_5    MOV     L,#0
150         SETB    P2.(L)
151 ;
152         MOV     L,#0BH
153         MOV     H,#4
154         DECS    L
155         JP      .-2
156 ;
157         MOV     L,#0
158         CLRB    P2.(L)
159         MOV     L,A
160         RET
161
162 D2      MOV     L,#01H          ;delay for 2 msec
163         MOV     H,#4
164         MOV     H,#4
165         MOV     H,#4
166         DECS    L
167         JP      .-4
168         MOV     H,#4
169         MOV     H,#4
170 D1_F    MOV     L,#0DH          ;be used in the fourth bit
171         MOV     H,#4
172         MOV     H,#4
173         JP      .+2
174 D1      MOV     L,#0FH          ;delay for 1 msec
175         MOV     H,#4
176         MOV     H,#4
177         MOV     H,#4
178         DECS    L
179         JP      .-3
180         RET
181
182 ;
183 ;
184 ;
185 ;
186 ;*******************************
187         ORG     0700H
188         JPL     RESET
189 ;*******************************
190 ;;;dalay time subroutine by programming ----------------------------
191 D20     CALL    D4
192         CALL    D4
193         CALL    D4
194         MOV     L,#0EH
195         MOV     H,#4
196         DECS    L
197         JP      .-2
198         MOV     H,#4
199         MOV     L,#0FH
200         JP      D_A
201
202 D8      MOV     L,#08H          ;delay for 8 msec
203         DECS    L
204         JP      .-1
205         MOV     L,#02H
206         JP      D_A
207
208 D4      MOV     L,#06H          ;delay for 4 msec
209         MOV     H,#4
210         JP      .+2
211
212 D4_L    MOV     L,#04H
213         DECS    L
214         JP      .-1
215
216         MOV     L,#07H
217 D_A     CLR     A
218         ADDS    A,#09H
219         MOV     H,#4
220         DECS    A
221         JP      .-2
222         MOV     H,#4
223         DECS    L
224         JP      .-7
225         RET
226
227 ;---------------------------------------------------------------
228         org     0800h
229         jpl     reset
230         org     0900h
231         jpl     reset
232         org     0a00h
233         jpl     reset
234         org     0b00h
235         jpl     reset
236         org     0c00h
237         jpl     reset
238         org     0d00h
239         jpl     reset
240         org     0e00h
241         jpl     reset
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
```

```
;;for PANASONIC  FORMAT
;;
;;If a key is pressed, a frame is transmitted, and repeat pulse is
;;transmitted until key off.   The frame interval is 100 msec.

;;*** transmission waveform ***
;;  ___                ___                ___
;;|   |   ||||||||||  |   |   ||||||||||  |   |   ||||||||||
;;|   |   ||||||||||  |   |   ||||||||||  |   |   ||||||||||
;;|   |___||||||||||__|   |___||||||||||__|   |___||||||||||
;;<-------- 100 ms ---------->
;;|               |
;;|               |_____________________________________________________
;;|                                                                     |
;;v                                                                     v
;;*** a frame waveform ***
;;  ____
;;|    |    ||  ||  || ||   || || || ||   || || ||  ||   ||  ||  ||  ||   ||
;;|    |    ||  ||  || ||   || || || ||   || || ||  ||   ||  ||  ||  ||   ||
;;|    |____||__||__||_||...||_||_||_||...||_||_||__||...||__||__||__||...||
;;|         |                |                | ___________ |  ________    |
;;head pulse<- custom code -><-data code -><-custom code-><-- data code -->
;;                5bits           6bits          5bits            6bits
;;
;;
;;*** head pulse & data pulse ***
;;        _______             ___                ___
;; head  |       |           data'0' |   |      data'1' |   |
;; pulse |       |                    |   |              |   |
;;       |       |________            |   |___           |   |________
;;         3.4ms   3.4ms                0.8ms              2.4ms
;;                                    0.9ms              0.9ms
;*********************************
        ORG     0500H
        JPL     RESET
;*********************************
TX      MOV     L,#09H          ;carrier frequency(= 56.9 kHz,1/2 duty)
        SETB    P2.(L)          ;2.9 <- high,  2.10 <- low
        MOV     L,#0AH
        CLRB    P2.(L)

;;;output head pulse ------------------------------------------------
        MOV     H,#4
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D3_4            ;high for delay 3.4 msec
        MOV     L,#0
        CLRB    P2.(L)          ;low for delay 3.4 msec
        CALLL   D3_4L

;;;output custom code (= 5bits) -------------------------------------
        MOV     L,#CUS0         ;output custom code(c0 - c3)
        CALLL   DATGEN
        MOV     L,#CUS1         ;output custom code(c4)
        CALLL   DATGEN0

;;;output data code (= 6bits) ----------------------------------------
        MOV     L,#DAT0         ;output data code(d0 - d3)
        CALLL   DATGEN
        MOV     L,#DAT1_0       ;output data code(d4,d5)
        CALLL   DATGEN1         ;

;;;      __________
;;;output custom code (= 5bits) -------------------------------------
        MOV     L,#CUS2         ;complement of custom code(c0-c3)
        CALLL   DATGEN
        MOV     L,#CUS3         ;complement of custom code(c4)
        CALLL   DATGEN0

;;;       _________
;;;output data code (= 6bits) ----------------------------------------
        MOV     L,#DAT2
        CALLL   DATGEN          ;complement of data code(d0-d3)
        MOV     L,#DAT3_0
        CALLL   DATGEN1         ;complement of data code(d4,d5)
        MOV     H,#4            ;delay time
        JPL     E_OF
;
;
;
;
;
;
;*********************************
        ORG     0600H
        JPL     RESET
;*********************************
;;;EOF bit -----------------------------------------------------------
E_OF    MOV     H,#4
        CALLL   D0_9            ;finished bit


;;;a frame delay time (= 100 msec) ----------------------------------
        CALLL   D100

;;;re-setting debounce count to #6 ----------------------------------
        MOV     H,#0
        MOV     L,#DEBOCNT
        MOV     @HL+,#06H
        JPL     MAIN
;
;*********************************
        ORG     0700H
        JPL     RESET
;*********************************
;;;generate custom code & data code ----------------------------------
DATGEN  MOV     A,L
        CALLL   D0_9            ;high for 0.9 msec
        CPBT    @HL.0           ;if @hl.0 is high, low for 2.4 msec.
        CALL    D2_4            ;otherwise, low for 0.8 msec.
        CALL    D0_8

        CALLL   D0_9            ;high for 0.9 msec
        CPBT    @HL.1           ;if @hl.1 is high, low for 2.4 msec.
        CALL    D2_4            ;otherwise, low for 0.8 msec.
        CALL    D0_8

        CALLL   D0_9            ;high for 0.9 msec
        CPBT    @HL.2           ;if @hl.2 is high, low for 2.4 msec.
        CALL    D2_4            ;otherwise, low for 0.8 msec.
        CALL    D0_8

        CALLL   D0_9            ;high for 0.9 msec
        CPBT    @HL.3           ;if @hl.3 is high, low for 2.4 msec.
        CALL    D2_4            ;otherwise, low for 0.8 msec.
        CALL    D0_8F
        RET

DATGEN1 MOV     A,L             ;check @hl.0 - hl.1
        CALLL   D0_9
        CPBT    @HL.0
```

```
131          CALL    D2_4
132          CALL    D0_8
133
134          CALLL   D0_9
135          CPBT    @HL.1
136          CALL    D2_4
137          CALL    D0_8F
138          RET
139
140  DATGEN0 MOV     A,L             ;check only @hl.0
141          CALLL   D0_9
142          CPBT    @HL.0
143          CALL    D2_4
144          CALL    D0_8F
145          RET
146
147
148  D2_4    MOV     L,#0DH          ;delay for 2.4 msec
149          MOV     H,#4
150          MOV     H,#4
151          MOV     H,#4
152          DECS    L
153          JP      .-4
154          MOV     H,#4
155          MOV     H,#4
156          MOV     H,#4
157
158  D0_8F   MOV     L,#0AH          ;delay for .8 msec
159          JP      .+4
160  D0_8    MOV     L,#0BH
161          MOV     H,#4
162          MOV     H,#4
163          DECS    L
164          JP      .-3
165          MOV     H,#4
166          RET
167  ;
168  ;
169  ;
170  ;
171  ;
172  ;
173  ;*******************************
174          ORG     0800H
175          JPL     RESET
176  ;*******************************
177  ;;;delay time subroutine by programming ----------------------------
178  D100    MOV     L,#4
179          CLR     A
180          ADDS    A,#0AH
181          MOV     H,#4
182          DECS    A
183          JP      .-2
184          DECS    L
185          JP      .-6
186
187          MOV     L,#0CH
188          DECS    L
189          JP      .-1
190
191
192
193  D3_4    MOV     L,#2            ;delay for 3.4 msec
194          DECS    L
195          JP      .-1
196  D3_4L   MOV     L,#04H
197          CLR     A
198          ADDS    A,#0EH
199          MOV     H,#4
200          DECS    A
201          JP      .-2
202          DECS    L
203          JP      .-6
204          RET
205
206  D0_9    MOV     L,#0
207           SETB    P2.(L)
208  ;
209          MOV     L,#0CH
210          MOV     H,#4
211          MOV     H,#4
212          MOV     H,#4
213          DECS    L
214          JP      .-4
215          MOV     H,#4
216  ;
217           MOV     L,#0
218           CLRB    P2.(L)
219           MOV     L,A
220           RET
221
222  ;---------------------------------------------------------------------
223           org     0900h
224           jpl     reset
225           org     0a00h
226           jpl     reset
227           org     0b00h
228           jpl     reset
229           org     0c00h
230           jpl     reset
231           org     0d00h
232           jpl     reset
233           org     0e00h
234           jpl      reset
235
236
237
```

```
;for MITSUBISHI FORMAT
;
;If a key is pressed, a frame waveform is transmitted.  And repeat
;pulse is transmitted until key off.   The frame interval is 60 msec.
;Separator is between custom code and data code.
;
;*** a frame waveform ***
;                   c0    c1 ....  c7             d0  d1 ... d7
; _________         _     _        _  _           _  _       _  _
;|         |       | |   | |      | | | |         | | | |     | | | |
;|         |       | |   | | .... | | | |         | | | | ... | | | |
;|         |_______| |___| |_    _| |_| |_________| |_| |___ _| |_| |
;
;|<- leader code ->|<--custom code -->| |seperator|<--data code ->|
;|<- 8ms ->|<-4ms->|    8-bit         | |<- 4ms ->|    8-bit      |
;
;*** data pulse ***
;     ___                      ___
;    |   |                    |   |
;    |   |  data '0'          |   |  data '1'
;    |   |___                 |   |___________
;  ->|.5ms|<-               ->|.5ms|<-
;    |<------->|              |<----- 2msec ---->|
;       1msec
;
;** SEPARATOR **
;The separator delimits custom code and data code and consists of low
;for 4 msec.  The separator can be determined at the receiver to
;avoid interference with other remote control system and to establish
;a system with very few operation errors.
;
;*********************************
        ORG     0500H
        JPL     RESET
;*********************************
;;; carrier frequency  = 37.9 kHz, 1/3 duty -------------------------
TX      MOV     L,#9
        CLRB    P2.(L)          ;clear p2.9 & p2.10
        MOV     L,#0AH
        CLRB    P2.(L)

        MOV     H,#4            ;H register selects file #4
;;;output head pulse ----------------------------------------------
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D8              ;high for delay 8 msec
        MOV     L,#0
        CLRB    P2.(L)
        MOV     H,#4
        MOV     H,#4
        CALLL   D4              ;low for delay 4 msec

;;;output custom code ----------------------------------------------
        MOV     L,#CUS0
        CALLL   CDGEN           ;output custom code (c0 - c3)
        MOV     L,#CUS1
        CALLL   CDGEN           ;output custom code (c4 - c7)
        MOV     H,#4
        MOV     H,#4
        MOV     H,#4
        MOV     H,#4
        CALLL   D0_5            ;high for delay 0.5 msec

;;;separator delay time --------------------------------------------
        CALLL   D4              ;separator = low for delay 4 msec

;;;output data code ------------------------------------------------
        MOV     L,#DAT0
        CALLL   CDGEN           ;output data code (d0 - d3)
        MOV     L,#DAT1_0
        CALLL   CDGEN           ;output data code (d4 - d7)
        MOV     H,#4
        MOV     H,#4

        JPL     S_DBCNT
;
;
;
;
;
;*********************************
        ORG     0600H
        JPL     RESET
;*********************************
;-----------------------------------------------------------------
;;;low value checksum
;;;all values are supposed as high
;;;If the value is low, delay time for 1 msec.
;-----------------------------------------------------------------
S_DBCNT CALLL   D0_5            ;high for delay 0.5 msec
;;;re-setting debounce count to #1 ---------------------------------
        MOV     H,#0
        MOV     L,#DEBOCNT
        MOV     @HL+,#1

;;;a frame delay time ----------------------------------------------
F_DELAY CALL    D60


;;;check low number of custom code & data code
LOWSUM  MOV     H,#4
        MOV     L,#CUS0         ;custom code (c0 - c3)
        CALL    LOWCHEK
        MOV     L,#CUS1         ;custom code (c4 - c7)
        CALL    LOWCHEK
        MOV     L,#DAT0         ;data code (d0 - d3)
        CALL    LOWCHEK
        MOV     L,#DAT1_0       ;data code (d4 - d7)
        CALL    LOWCHEK
        JPL     MAIN



LOWCHEK MOV     A,L
        CPBT    @HL.0
        JP      LOW_1
        CALLL   D2_F            ;add 1 msec

LOW_1   MOV     L,A
        CPBT    @HL.1
        JP      LOW_2
        CALLL   D2_F            ;add 1 msec

LOW_2   MOV     L,A
        CPBT    @HL.2
        JP      LOW_3
        CALLL   D2_F            ;add 1 msec

LOW_3   MOV     L,A
        CPBT    @HL.3
```

```
JP      LOW_R
        CALLL   D2_F            ;add 1 msec
LOW_R   RET



;;delay time subroutine ------------------------------------------
D60       MOV       L,#0FH
          MOV       H,#4
          DECS      L
          JP        .-2

          MOV     L,#03H
          JP        D_A

D8        MOV       L,#0EH
          DECS      L
          JP        .-1

          MOV       L,#9
          JP        D_A
D4        MOV       L,#4
D_A       CLR       A
          ADDS      A,#0CH
D_A1      MOV       H,#4
          MOV       H,#4
          DECS      A
          JP        .-3
          DECS      L
          JP        .-7

          MOV       L,#3
          DECS      L
          JP        .-1
          RET

;
;
;
;
;
;*********************************
          ORG     0700H
          JPL     RESET
;*********************************
;;;generate custom code & data code --------------------------------
CDGEN   MOV     A,L
        CALL    D0_5            ;high for 0.5 msec
        CPBT    @HL.0
        CALL    D2              ;If HL.0 is high, low for 2 msec
        CALL    D1              ;     "     low,  low for 1 msec

        CALL    D0_5            ;high for 0.5 msec
        CPBT    @HL.1           ;if HL.1 is high, low for 2 msec
        CALL    D2              ;otherwise, low for 1 msec
        CALL    D1

        CALL    D0_5            ;high for 0.5 msec
        CPBT    @HL.2           ;if HL.2 is high, low for 2 msec
        CALL    D2              ;otherwise, low for 1 msec
        CALL    D1

        CALL    D0_5            ;high for 0.5 msec
        CPBT    @HL.3           ;if HL.3 is high, low for 2 msec
        CALL      D2            ;otherwise, low for 1 msec
        CALL      D1_F
        RET

D2_F      MOV       L,#09H
          JP        .+3
D2        MOV       L,#0AH
          JP        .+4
          MOV       H,#4
          MOV       H,#4
          MOV       H,#4
          DECS      L
          JP        .-4

D1_F      MOV       L,#5
          JP        .+2
D1        MOV       L,#7
          MOV       H,#4
          DECS      L
          JP        .-2
          MOV       H,#4
          MOV       H,#4
          RET

D0_5      MOV       L,#0
          SETB      P2.(L)
;
          MOV       L,#0BH
          MOV       H,#4
          DECS      L
          JP        .-2
;
          MOV       L,#0
          CLRB      P2.(L)
          MOV       L,A
          RET
;
;
;
;
;
;----------------------------------------------------------------
          org       0800h
          jpl       reset
          org       0900h
          jpl       reset
          org       0a00h
          jpl       reset
          org       0b00h
          jpl       reset
          org       0c00h
          jpl       reset
          org       0d00h
          jpl       reset
          org       0e00h
          jpl       reset
```

```
;; for NEC format
;;
;;If a key pressed, a frame waveform is transmitted.  And, continuous
;;pulse is transmitted until key off.  The frame interval is 108 msec.
;;A frame  consists of leader code, custom code, data code.
;;-- leader code consists of high for 9 msec and low for 4.5 msec.
;;-- custom code consists of 16-bit (custom code 8-bit,complement of
;;   custom code 8-bit)
;;-- data code consists of 16-bit (data code 8-bit,complement of data
;;   code 8-bit)
;;*** transmission waveform ***
;;   _____                 _____
;;  |     |  ||||||||      |     |  |
;;  |     |  ||||||||      |     |  |
;;__|     |__||||||||______|     |__|_____________
;;  |<-- 67.5ms --->|
;;  |<------- 108 ms ------>|<------ 108 ms ------->|
;;
;;*** a frame waveform ***
;; _______     _  _                                  _     _  _     _
;;|       |    | || |                                | |    | || ||    | |
;;|       |    | || |                                | |    | || ||    | |
;;|       |____| |_| |___   ........................___| |____| |_| |____| |
;;    9ms   4.5ms                _________                _________   |
;;<- leader  ->|<-custom code->|<-custom code->|<-data code->|<-data code->|
;;                  8bits           8bits          8bits         8bits
;;*** continuous signal ***
;;       _______     _
;;      |       |  ->| |<- 0.56msec
;;      |       |    | |
;;      |       |____| |________
;;    ->| 9msec |<-  |
;;      |       |<-->|
;;       2.25msec
;;
;;*** data pulse ****
;;   __                          __
;;  | |                         |  |
;;  | |    data '0'             |  |          data '1'
;;  | |___                      |  |_________
;;->|  |<-                    ->|  |<-
;; 0.56ms                       0.56msec
;;->|     |<- 1.125ms           |<-2.25msec->|
;;
;*******************************
'          ORG      0500H
           JPL      RESET
;*******************************
TX      MOV     L,#9            ;select carrier frequency
        CLRB    P2.(L)          ;(1/3 duty, 37.9 kHz)
        MOV     L,#0AH          ;clear p2.9 & p2.10
        CLRB    P2.(L)

        MOV     H,#0
        MOV     L,#CONKEY       ;if CONKEY is not zero,
        CPNZ    @HL             ;tx. continuous signal
        JP      J_SIGCON

SIGOUT    MOV       @HL+,#1
;;;output leader pulse -------------------------------------------
        MOV     L,#0
        SETB    P2.(L)
        CALLL   D9              ;high for delay 9 msec
        MOV     L,#0
        CLRB    P2.(L)
        CALLL   D4_5            ;low for delay 4.5 msec

;;;output custom code ---------------------------------------------
        MOV     L,#CUS0
        CALLL   DATGEN          ;custom code (c0 - c3)
        MOV     L,#CUS1
        CALLL   DATGEN          ;custom code (c4 - c7)
        MOV     L,#CUS2
        CALLL   DATGEN          ;complement of custom code (c0 - c3)
        MOV     L,#CUS3
        CALLL   DATGEN          ;complement of custom code (c4 - c7)

;;;output data code -----------------------------------------------
        MOV     L,#DAT0
        CALLL   DATGEN          ;data code (d0 - d3)
        MOV     L,#DAT1_0
        CALLL   DATGEN          ;data code (d4 - d7)
        MOV     L,#DAT2
        CALLL   DATGEN          ;complement of data code (d0 - d3)
        MOV     L,#DAT3_0
        CALLL   DATGEN          ;complement of data code (d4 - d7)
        MOV     H,#4            ;delay time
        MOV     H,#4
        MOV     H,#4

;;;EOB (end of bit) -----------------------------------------------

        CALLL   D_56

        CALLL   D108

;;;re-setting debounce count to #6 --------------------------------
        MOV     H,#0
        MOV     L,#DEBOCNT
        MOV     @HL+,#6

        JPL     MAIN

J_SIGCON
        JPL     SIGCON
;
;
;
;
;
;
;*********************************
        ORG     0600H
        JPL     RESET
;*********************************
;;; output continue signal -----------------------------------------
;;
;;; output leader pulse --------------------------------------------
SIGCON  MOV     L,#0
        SETB    P2.(L)
        CALLL   D9              ;high for delay 9 msec
        MOV     L,#0
        CLRB    P2.(L)

        CALLL   D2_25           ;low for delay 2.25 msec
        MOV     L,#0CH
        CALLL   D2_25F

;;;EOB (end of bit) -----------------------------------------------
        CALLL   D_56            ;high for delay 0.56 msec

```

```
131
132  ;;;re-setting debounce count to #0fh --------------------------------
133          MOV     H,#0
134          MOV     L,#DEBOCNT
135          MOV     @HL+,#0FH
136
137  ;;;
138          CALL    D96_19      ;delay time 96.19 msec
139
140          JPL     MAIN
141
142  ;;;; delay time of signal output(p2.0)
143  D108    MOV     L,#0AH
144          MOV     H,#4
145          DECS    L
146          JP      .-2
147          MOV     L,#0BH
148          JP      D_A
149
150  D96_19  MOV     L,#0FH
151          CALL    D_A
152  D9      MOV     L,#0CH
153          JP      D_A
154
155  D4_5    MOV     L,#7
156          DECS    L
157          JP      .-1
158          MOV     L,#05H
159  D_A     CLR     A
160          ADDS    A,#0FH
161          MOV     H,#4
162          DECS    A
163          JP      .-2
164          DECS    L
165          JP      .-6
166
167          RET
168  ;
169  ;
170  ;
171  ;
172  ;
173  ;*******************************
174          ORG     0700H
175          JPL     RESET
176  ;*******************************
177  ;; subroutine for signal out of custom code & data code
178  DATGEN  MOV     A,L
179          CALL    D_56            ;high for delay .56 msec
180          CPBT    @HL.0           ;if @hl.0 is high, low for 2.25 msec
181          CALL    D2_25           ;otherwise, low for 1.125 msec
182          CALL    D1_125
183
184          CALL    D_56            ;high for delay .56 msec
185          CPBT    @HL.1           ;if @hl.1 is high, low for 2.25 msec
186          CALL    D2_25           ;otherwise, low for 1.125 msec
187          CALL    D1_125
188
189          CALL    D_56            ;high for delay .56 msec
190          CPBT    @HL.2           ;if @hl.2 is high, low for 2.25 msec
191          CALL    D2_25           ;otherwise, low for 1.125 msec
192          CALL    D1_125
193
194          CALL    D_56            ;high for delay .56 msec
195          CPBT    @HL.3           ;if @hl.3 is high, low for 2.25 msec
196          CALL    D2_25           ;otherwise, low for 1.125 msec
197          CALL    D1_125F
198          RET
199
200  D_56    MOV     L,#0
201          SETB    P2.(L)
202  ;
203          MOV      L,#0CH
204          MOV      H,#4
205          DECS     L
206          JP       .-2
207          MOV      H,#4
208  ;
209          MOV      L,#0
210          CLRB     P2.(L)
211          MOV      L,A
212          RET
213
214  D2_25   MOV      L,#0FH
215  D2_25F  MOV      H,#4
216          DECS     L
217          JP       .-2
218          MOV      H,#4
219          MOV      H,#4
220  D1_125  MOV      L,#09H
221          JP       .+2
222  D1_125F MOV      L,#08H
223          MOV      H,#4
224          DECS     L
225          JP       .-2
226          MOV      H,#4
227          RET
228  ;---------------------------------------------------------------------
229          org      0800h
230          jpl      reset
231          org      0900h
232          jpl      reset
233          org      0a00h
234          jpl      reset
235          org      0b00h
236          jpl      reset
237          org      0c00h
238          jpl      reset
239          org      0d00h
240          jpl      reset
241          org      0e00h
242          jpl      reset
243
244
245
```

```
;; for NEC FORMAT
;;
;;If a key is pressed, a frame is transmitted.  Continuous pulse is
;;transmitted until key off.   The frame interval is 108 msec.
;;A frame consists of leader code, custom code (16-bit), data code
;;(16-bit).
;;--- reader code consists of high for 9msec and low for 4.5msec.
;;--- data code consists of data code (8-bit), the complement of
;;    data code (8-bit).
;;--- custom code consists of according to diodes(= 16-bit) between
;;    p2.6 and input port and between p2.1 and input port.
;;
;;*** transmission waveform ***
;;   _____                   _____
;;  |     |  ||||||||       |     |  |
;;  |     |  ||||||||       |     |  |
;;__|     |__||||||||_______|     |__|_____________
;;  |<-- 67.5ms --->|
;;  |<------- 108 ms ------>|<------ 108 ms ------->|
;;
;;*** a frame waveform ***
;; _______     _   _     _                             _     _
;;|       |    | | |    | |                           | |   | |
;;|       |    | | |    | |                           | |   | |
;;|       |____| |_| |____| |____  ......................_| |___| |_
;;    9ms   4.5ms                                     _______   |
;;<- leader  ->|<- custom code ->|<- data code ->|<- data code ->|
;;                 16-bit           8-bit           8-bit
;;
;;*** continuous signal ***
;; __________      _
;;|          |    | |
;;|          |   ->| |<- 0.56msec
;;|          |_____| |______
;;|<--9msec->|<--->|
;;            2.25msec
;;
;;*** data pulse ***
;;          __                        __
;;         |  |                      |  |
;; data'0' |  |           data'1'    |  |
;;         |  |___                   |  |________
;;       ->|  |<-                  ->|  |<- 0.56msec
;;        0.56ms                     |<--------->|
;;       ->|      |<- 1.125ms           2.25msec
;;
;; Carrier frequency: fosc/12 = 38 kHz, 1/3 duty
;*******************************
          ORG       0500H
          JPL       RESET
;*******************************
;;; product custom code (c8 - c15) ---------------------------------
TX        MOV       L,#0DH
          CLRB      P2.(L)
          MOV       H,#4
          MOV       L,#CUS2
          IN        A,P0
          NOTI      A
          DECS      A
          MOVZ      @HL,A           ;product custom code (c8 - c11)

          MOV       L,#0DH
          SETB      P2.(L)
          MOV       L,#CUS3
          IN        A,P0
          NOTI      A
          DECS      A
          MOVZ      @HL,A           ;product custom code (c12 - c15)

;;;before code tx., programmer must select carrier frequency.
;;;select carrier frequency (= 1/3 duty, 37.9 kHz) -----------------
;         MOV       L,#9
;         CLRB      P2.(L)
;         MOV       L,#0AH
;         CLRB      P2.(L)

;;;continuous signal generate if CONKEY is not zero ----------------
          MOV       H,#0
          MOV       L,#CONKEY
          CPNZ      @HL
          JP        J_SIGCON

SIGOUT    MOV       @HL+,#1
;;;output head pulse ------------------------------------------------
          MOV       L,#0
          SETB      P2.(L)
          CALLL     D9              ;high for delay 9 msec
          MOV       L,#0
          CLRB      P2.(L)
          CALLL     D4_5            ;low for delay 4.5 msec

;;;output custom code -----------------------------------------------
          MOV       L,#CUS0
          CALLL     DATGEN          ;custom code (c0 - c3)
          MOV       L,#CUS1
          CALLL     DATGEN          ;custom code (c4 - c7)
          MOV       L,#CUS2
          CALLL     DATGEN          ;custom code (c8 -c11)
          MOV       L,#CUS3
          CALLL     DATGEN          ;custom code (c12-c15)

;;;output data code -------------------------------------------------
          MOV       L,#DAT0
          CALLL     DATGEN          ;data code (d0 - d3)
          MOV       L,#DAT1_0
          CALLL     DATGEN          ;data code (d4 - d7)
          MOV       L,#DAT2
          CALLL     DATGEN          ;the complement of DAT0
          MOV       L,#DAT3_0
          CALLL     DATGEN          ;the complement of DAT1_0
          MOV       H,#4            ;delay time
          MOV       H,#4
          MOV       H,#4

;;;EOB (end of bit) -------------------------------------------------

          CALLL     D_56            ;high for .56 msec

          JPL       D108

J_SIGCON
          JPL       SIGCON
;
;
;
;
;
;*********************************
          ORG       0600H
          JPL       RESET
```

```
;*******************************
;;; until 108 msec (= a frame) ------------------------------------
D108      MOV       L,#7
          DECS      L
          JP        .-1

          MOV       L,#4
          CALL      D_A


;;;re-setting debounce count to #5 ---------------------------------
          MOV       H,#0
          MOV       L,#DEBOCNT
          MOV       @HL+,#5


;;; delay time as many as low numbers of custom code ----------------
LOWCHEK   MOV       H,#4
          MOV       L,#CUS0         ;custom code (c0 - c3)
          CALLL     LOWDLY
          MOV       L,#CUS1         ;custom code (c4 - c7)
          CALLL     LOWDLY
          MOV       L,#CUS2         ;custom code (c8 -c11)
          CALLL     LOWDLY
          MOV       L,#CUS3         ;custom code (c12-c15)
          CALLL     LOWDLY
          JPL       MAIN
;;; output continuous signal  ---------------------------------------
;;
;;; output head pulse -----------------------------------------------
SIGCON    MOV       L,#0
          SETB      P2.(L)
          CALLL     D9              ;high for delay 9 msec
          MOV       L,#0
          CLRB      P2.(L)

          CALLL     D2_25           ;low for delay 2.25 msec
          MOV       L,#0CH
          CALLL     D2_25F

;;;EOB (end of bit) ------------------------------------------------

          CALLL     D_56            ;high for delay 0.56 msec

;;;re-setting debounce count to #0fh -------------------------------
          MOV       H,#0
          MOV       L,#DEBOCNT
          MOV       @HL+,#0FH

;;;
          CALL      D96_19      ;delay time 96.19 msec

          JPL       MAIN

;;;; delay time of signal output(p2.0)

D96_19    MOV       L,#0CH
          MOV       H,#4
          DECS      L
          JP        .-2
          MOV       L,#0EH
          CALL      D_A

D9        MOV       L,#0CH
          JP        D_A


D4_5      MOV       L,#7
          DECS      L
          JP        .-1
          MOV       L,#05H
D_A       CLR       A
          ADDS      A,#0FH
          MOV       H,#4
          DECS      A
          JP        .-2
          DECS      L
          JP        .-6

          RET
;
;
;
;
;
;*******************************
          ORG       0700H
          JPL       RESET
;*******************************
;; subroutine for signal out of custom code & data code
DATGEN    MOV       A,L
          CALL      D_56            ;high for .56 msec
          CPBT      @HL.0           ;if @hl.0 is high, low for 2.25 msec.
          CALL      D2_25           ;otherwise low for 1.125 msec.
          CALL      D1_125

          CALL      D_56            ;high for .56 msec
          CPBT      @HL.1           ;if @hl.1 is high, low for 2.25 msec.
          CALL      D2_25           ;otherwise low for 1.125 msec.
          CALL      D1_125

          CALL      D_56            ;high for .56 msec
          CPBT      @HL.2           ;if @hl.2 is high, low for 2.25 msec.
          CALL      D2_25           ;otherwise low for 1.125 msec.
          CALL      D1_125

          CALL      D_56            ;high for .56 msec
          CPBT      @HL.3           ;if @hl.3 is high, low for 2.25 msec.
          CALL      D2_25           ;otherwise low for 1.125 msec.
          CALL      D1_125F
          RET

D_56      MOV       L,#0
          SETB      P2.(L)
;
          MOV       L,#0CH
          MOV       H,#4
          DECS      L
          JP        .-2
          MOV       H,#4
;
          MOV       L,#0
          CLRB      P2.(L)
          MOV       L,A
          RET


D2_25D    MOV       L,#0FH
          JP        .+4
D2_25     MOV       L,#0FH
D2_25F    MOV       H,#4
          MOV       H,#4
```

```
261             MOV       H,#4
262             DECS      L
263             JP        .-2
264   D1_125    MOV       L,#09H
265             JP        .+2
266   D1_125F   MOV       L,#08H
267             MOV       H,#4
268             DECS      L
269             JP        .-2
270             MOV       H,#4
271             RET
272
273   LOWDLY    MOV       A,L
274             CPBT      @HL.3           ;if @hl.3 is low, add delay 1.125 msec
275             JP        LOW_2
276             CALL      D2_25D
277
278   LOW_2     MOV       L,A
279             CPBT      @HL.2           ;if @hl.2 is low, add delay 1.125 msec
280             JP        LOW_1
281             CALL      D2_25D
282
283   LOW_1     MOV       L,A
284             CPBT      @HL.1           ;if @hl.1 is low, add delay 1.125 msec
285             JP        LOW_0
286             CALL      D2_25D
287
288   LOW_0     MOV       L,A
289             CPBT      @HL.0           ;if @hl.0 is low, add delay 1.125 msec
290             JP        LOW_R
291             CALL      D2_25D
292
293   LOW_R     RET
294
295   ;-----------------------------------------------------------------------
296             org       0800h
297             jpl       reset
298             org       0900h
299             jpl       reset
300             org       0a00h
301             jpl       reset
302             org       0b00h
303             jpl       reset
304             org       0c00h
305             jpl       reset
306             org       0d00h
307             jpl       reset
308             org       0e00h
309             jpl       reset
310
311
312
313
```