HT9580
Character Pager Controller

Features

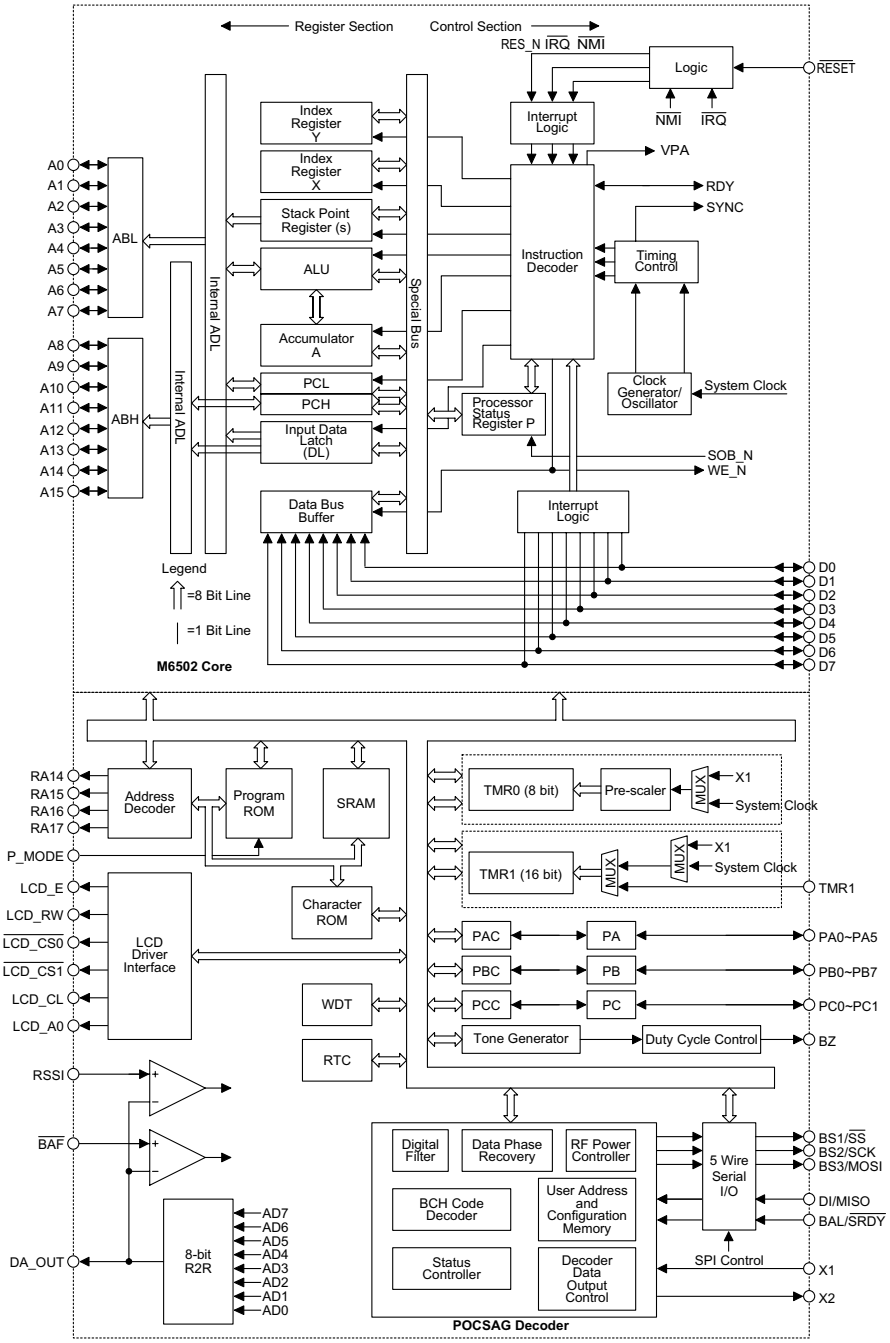
- Operating voltage: 2.4V~3.3V
- Temperature range: –30°C to +85°C
- Low power, high performance M6502 core
- Low power crystal oscillator control
  – 512/1200/2400 bps data rate operation
- "CCIR Radio Paging Code No.1" (POCSAG) compatible
- 76.8kHz crystal for all available data rates
- High/low system clock switching capability
- 44 Kbytes program ROM
- 848 bytes global data RAM
- Internal 2 Mbits Character ROM
- 32 Kbytes internal SRAM
- External option up to 2 Mbits Character ROM or 2 Mbits SRAM
- SED15X(KSX), MC141X and HD66410 series LCD driver compatible interface option
- 46 bytes message buffer
- Built-in 5 wire serial I/O
- One 16-bit timer and one 8-bit timer
- One internal 8-bit A/D converter
- Internal 2Hz or 1Hz RTC option
- Single buzzer generator output (BZ) with duty cycle control
- Low current HALT mode operation
- 16-bit Watchdog Timer
- Built-in data filter (16-times over-sampling) and bit clock recovery
- Advanced synchronization algorithm
- 2-bit random and optional 4-bit burst error correction for address and message
- Up to 6 user addresses and 6 user frames, independently programmable
- 3 RF power-on timing control pins and Received data inversion (optional)
- Out-of-range condition indicator
- Battery fail and battery low detection
- 80-pin LQFP package

General Description

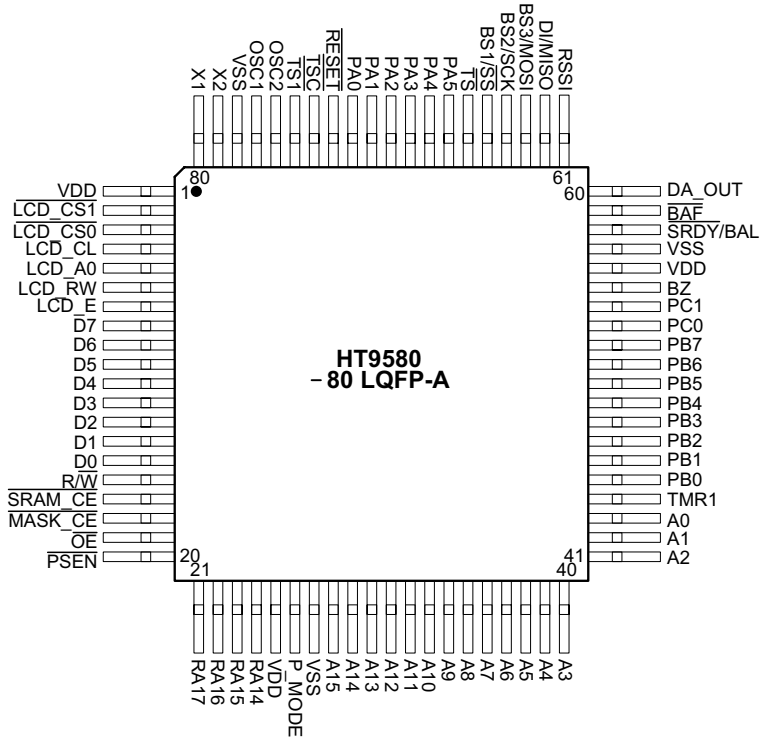
The HT9580 is a high performance pager controller which can be used for Character Pager system applications. The HT9580 4-in-1 Character Pager Controller combines a POCSAG decoder with a M6502 microprocessor core, 2 Mbits Character ROM and 32 Kbytes SRAM to provide both high decoder performance and excellent system flexibility. The decoder utilizes a 2-bit random error correction algorithm and therefore provides excellent decoder sensitivity. The controller contains a full function pager decoder at a 512, 1200, 2400 bps data rates. Using an M6502 core takes advantage of a flexible external control interface, LCD driver chips and abundant programming resources from worldwide providers. The internal 5 wire serial I/O could communicate with FLEX^{TM} high speed pager decoder.

FLEX^{TM} is a trademark of Motorola Inc.

Block Diagram



Pin Assignment



Pin Description

Pin No.	Pin Name	I/O	Description
1, 25, 56	VDD	—	Positive power supply
2	$\overline{\text{LCD\_CS1}}$	O	LCD driver chip select control (for slave LCD driver)
3	$\overline{\text{LCD\_CS0}}$	O	LCD driver chip select control (for master LCD driver)
4	LCD_CL	O	LCD driver clock output
5	LCD_A0	O	LCD driver data/command select control
6	LCD_RW	O	LCD Driver Read/Write signal output
7	LCD_E	O	LCD driver enable clock control
15~8	D0~D7	I/O	8-bit, tristate, bidirectional I/O data bus
16	R/$\overline{\text{W}}$	O	Read/Write signal output
17	$\overline{\text{SRAM\_CE}}$	O	SRAM Chip Enable. This signal is generated to provide read or write timing for external SRAM devices. (See Application Circuit)
18	$\overline{\text{MASK\_CE}}$	O	Mask ROM Chip Enable. This signal is generated to provide read timing for external Character ROM devices. (See Application Circuit)
19	$\overline{\text{OE}}$	O	Mask ROM or SRAM Output Enable. This signal is generated to provide read timing for external Character ROM and SRAM devices. (See Application Circuit)
20	$\overline{\text{PSEN}}$	O	Program Strobe Enable. This pin is used to connect the $\overline{\text{OE}}$ and $\overline{\text{CE}}$ pins of the external 44 Kbytes program ROM when the "P_MODE" pin is connected to VSS.
21~24	RA17~RA14	O	Extended address bus pins
26	P_MODE	I	Internal or external program ROM selection without pull-high resistor. If the pin connects to VDD, the internal program ROM will be fetched (normal type), otherwise the external program ROM will be fetched when the pin connects to VSS (ROMless).
27, 57, 78	VSS	—	Negative power supply, ground
43~28	A0~A15	O	Address bus pins. This is used for memory and I/O exchanges on the data bus.
44	TMR1	I	Schmitt trigger input for timer1 counter with pull-high resistor.
45~52	PB0~PB7	I/O	General Input/Output Port B. The input cell structures can be selected as CMOS or CMOS with pull-high resistors.
53~54	PC0~PC1	I/O	General Input/Output Port C. The input cell structures can be selected as CMOS or CMOS with pull-high resistors.
55	BZ	O	Buzzer non-inverting BZ output

Pin No.	Pin Name	I/O	Description
58	BAL	I	Battery voltage detector input with pull-high resistor.
	$\overline{\text{SRDY}}$	I	SPI slave ready — This slave ready pin is a Schmitt trigger input with pull-high resistor. When the slave initiates the SPI transfer, a high to low transition activates an interrupt. When the master initiates the SPI transfer, a high to low transition triggers the master to start the transfer.
59	$\overline{\text{BAF}}$	I	Battery fail indication input.
60	DA_OUT	O	A/D converter output. This pin is an 8-bit A/D analog output.
61	RSSI	I	RSSI output from IF circuit. This pin should be externally pulled high or low when this pin is not used.
62	DI	I	POCSAG code input serial data. CMOS input with pull-high resistor.
	MISO	I	SPI master-in-slave-out — this is the data input with pull-high resistor for 5 wire serial I/O communications.
63	BS3	O	PLL power control enable, CMOS output
	MOSI	O	SPI master-out-slave-in — this is the data output for 5 wire serial I/O communications.
64	BS2	O	RF quick charge control enable, CMOS output
	SCK	I/O	SPI serial clock — the SCK signal is used to synchronize the data transfer. If HT9580 is in the master mode, the SCK is an output clock. Otherwise, SCK is an input clock if HT9580 is in the slave mode.
65	BS1	O	Pager receiver power control enable output, CMOS output
	$\overline{\text{SS}}$	O	SPI slave select — this signal is used to enable the 5 wire slave of the 5 wire serial I/O for transfer.
66	$\overline{\text{TS}}$	I	Decoder test mode input pin, active low with pull-high resistor.
72~67	PA0~PA5	I/O	General Input/Output Port A. These ports can be programmed to have a wake-up capability for applications in key operations or as normal I/O. Also the input cell structures are all Schmitt trigger types and can be selected between CMOS or CMOS with pull-high resistors.
73	$\overline{\text{RESET}}$	I	Schmitt trigger reset input, active low.
74	$\overline{\text{TSC}}$	I	μC test mode input pin, active low with internal pull-high resistor. The test circuit will be activated when this pin pulls low.
75	$\overline{\text{TS1}}$	I	Decoder test mode input pin, active low with pull-high resistor. The internal test mode will be activated when this pin pulls low.
77 76	OSC1 OSC2	I O	OSC1 and OSC2 are connected to an RC network to form a main clock oscillator
80 79	X1 X2	I O	X1 and X2 are connected to a crystal to form an internal low power clock oscillator (32.768kHz, 76.8kHz, or 153.6kHz)

Absolute Maximum Ratings

Supply Voltage.....–0.3V to 3.6V
Input Voltage.....V_{SS}–0.5V to V_{DD}+0.5V
Current Drain Per Pin Excluding V_{DD} and V_{SS}.....10mA
Storage Temperature .....–55°C to 150°C
Operating Temperature .....–30°C to 85°C

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to the device. Functional operation of this device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	3V application	2.4	3.0	3.3	V
I_{DD}	Operating Current	3V	No load, f_{X1}=76.8kHz	—	50	—	μA
I_{STP}	HALT Mode Current	3V	No load, μC clock stop, f_{X1}=76.8kHz	—	30	40	μA
V_{IL}	Input low Voltage for I/O Port	3V	—	0	—	0.3×V_{DD}	V
V_{IH}	Input High Voltage for I/O Port	3V	—	0.7×V_{DD}	—	3	V
V_{IL1}	Input low Voltage	3V	—	0	—	0.3×V_{DD}	V
V_{IH1}	Input High Voltage	3V	—	0.7×V_{DD}	—	3	V
V_{IL2}	Input low Voltage ($\overline{BAF}$)	3V	—	0	—	0.9	V
V_{IH2}	Input High Voltage ($\overline{BAF}$)	3V	—	1.0	—	3	V
V_{OL}	Output low Voltage	3V	—	—	—	0.4	V
V_{OH}	Output High Voltage	3V	—	2.3	—	—	V
I_{OL}	I/O Port Sink Current	3V	V_{OL}=0.3V	2.0	3.6	—	mA
I_{OH}	I/O Port Source Current	3V	V_{OH}=2.7V	–1.2	–2.2	—	mA
I_{OL1}	BZ, PC0~PC1 Sink Current	3V	V_{OL}=0.3V	2	4.5	—	mA
I_{OH1}	BZ, PC0~PC1 Source Current	3V	V_{OH}=2.7V	–1.5	–2.5	—	mA
I_{OL2}	BS1, BS2, BS3 Sink Current	3V	V_{OL}=0.3V	350	—	—	μA
I_{OH2}	BS1, BS2, BS3 Source Current	3V	V_{OH}=2.7V	–1.0	—	—	mA
R_{OSC}	RC Oscillator Resistor	3V	f_{OSC}=300kHz	—	210	—	kΩ
R_{PH1}	I/O Port Pull-high Resistance (Port A)	3V	—	—	150	—	kΩ
R_{PH2}	I/O Port Pull-high Resistance (Excluding Port A)	3V	—	—	250	—	kΩ

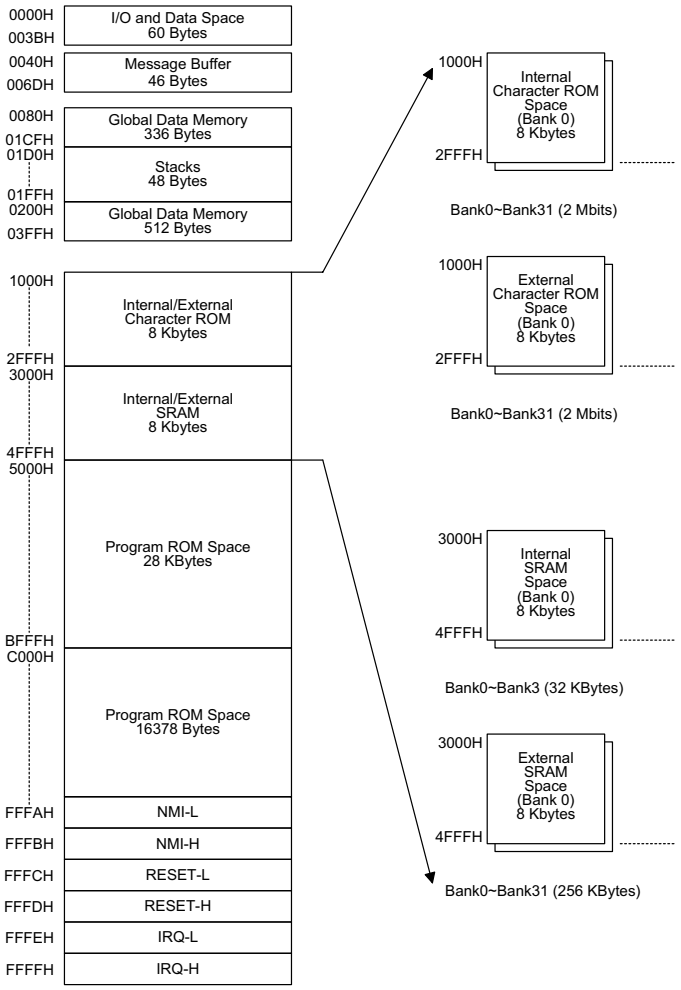
A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{OSC1}	Main Clock (RC OSC)	3V	—	300	1000	2000	kHz
D_{OSC1}	Main Clock Duty Cycle	3V	—	40	50	60	%
f_{X1}	Pager Clock Input (Crystal OSC)	3V	—	32.768	76.8	153.6	kHz

Functional Description

Memory map



HT9580 memory mapping table (I/O and data space)

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0000H	Config.	HALT	CLK_SEL	OSC_MOD	LPM	RTC	BZ_CLK	MDUT	MGEN	0001 0000
0001H	WDT-TMR	X	X	TMR0_PR1	TMR0_PR0	WDTEN	WS2	WS1	WS0	0000 0111
0002H	CLR WDT	X	X	X	X	X	X	X	X	uuuu uuuu
0003H	BZ-L	BZL7	BZL6	BZL5	BZL4	BZL3	BZL2	BZL1	BZL0	0000 0000
0004H	BZ-H	BZH7	BZH6	BZH5	BZH4	BZH3	BZH2	BZH1	BZH0	0000 0000
0005H	INT ctrl	0	F_MSK	R_MSK	RTCEN	ORMSK	RTCMSK	TM1IMSK	TM0IMSK	0110 1111
0006H	INT flag	0	RTC_FG	DR_FG	BF_FG	WDTOVFG	OR_FG	TM1OVFG	TM0OVFG	0000 0000
0007H	TMRC	TMR1MOD	X	TMR1CLK	TMR0CLK	TMR1EDG	TMR0EDG	TMR1EN	TMR0EN	0000 0000
0008H	TMR1L	TM1D7	TM1D6	TM1D5	TM1D4	TM1D3	TM1D2	TM1D1	TM1D0	uuuu uuuu
0009H	TMR1H	TM1D15	TM1D14	TM1D13	TM1D12	TM1D11	TM1D10	TM1D9	TM1D8	uuuu uuuu
000AH	TMR0	TM0D7	TM0D6	TM0D5	TM0D4	TM0D3	TM0D2	TM0D1	TM0D0	uuuu uuuu
000BH	PA data	X	X	PA5	PA4	PA3	PA2	PA1	PA0	uu11 1111
000CH	PB data	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	1111 1111
000DH	PC data	X	X	X	X	X	X	PC1	PC0	uuuu uu11
000EH	PAC	X	X	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0	uu11 1111
000FH	PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC3	PBC1	PBC0	1111 1111
0010H	PCC	X	X	X	X	X	X	PCC1	PCC0	uuuu uu11
0011H	PA WUE	X	X	PAWUE5	PAWUE4	PAWUE3	PAWUE2	PAWUE1	PAWUE0	uu00 0000
0012H	PA IM	X	X	PAIM5	PAIM4	PAIM3	PAIM2	PAIM1	PAIM0	uu11 1111
0013H	PB IM	PBIM7	PBIM6	PBIM5	PBIM4	PBIM3	PBIM2	PBIM1	PBIM0	1111 1111
0014H	PC IM	X	X	X	X	X	X	PCIM1	PCIM0	uuuu uu11
0015H	MROM-BP	BP_MODM1	BP_MODM0	M_BP5	M_BP4	M_BP3	M_BP2	M_BP1	M_BP0	0000 0000
0016H	SRAM-BP	BP_MODS1	BP_MODS0	S_BP5	S_BP4	S_BP3	S_BP2	S_BP1	S_BP0	0000 0000
0017H	LCD_CTRL	LCD-CHIP1	LCD-CHIP0	LCD-CLK	CLK-MOD	LCD-CS1	LCD-CS0	LCD-A0	LCD-WRB	0000 1101
0018H	LCD_CMD	LCD_D7	LCD_D6	LCD_D5	LCD_D4	LCD_D3	LCD_D2	LCD_D1	LCD_D0	uuuu uuuu
0019H	Decoder Control/ flag	X	$\overline{BL}$	$\overline{OR}$	X	STB	X	$\overline{RES}$	$\overline{ON}$	uu0u uu01
001AH~ 002EH	Decoder Configuration Memory									uuuu uuuu
002FH	A/D-L	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0000 0000
0030H	A/D-H	X	X	X	X	X	A/D_PD	RSSI	BAT	uuuu u1uu
0031H	Buffer Status	MSG_END	X	count_5	count_4	count_3	count_2	count_1	count_0	0uuu uuuu
0032H	SPI-CONFIG	S/M	LEN1	LEN0	REQST	SPIFG	CLK_EDG	SPI_EN	START	0111 1000
0033H	SPI-SPEED	X	X	X	X	X	X	X	SP0	0000 0000
0034H	SPI-OUT3	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0035H	SPI-OUT2	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0036H	SPI-OUT1	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0037H	SPI-OUT0	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0038H	SPI-IN3	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0039H	SPI-IN2	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
003AH	SPI-IN1	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
003BH	SPI-IN0	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000

HT9580 memory attribute table (I/O and data space)

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0000H	Config.	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0001 0000
0001H	WDT-TMR	X	X	R/W	R/W	R/W	R/W	R/W	R/W	0000 0111
0002H	CLR WDT	W	W	W	W	W	W	W	W	uuuu uuuu
0003H	BZ-L	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0004H	BZ-H	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0005H	INT ctrl	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0110 1111
0006H	INT flag	0	R/W	R/W	R	R/W	R/W	R/W	R/W	0000 0000
0007H	TMRC	R/W	X	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0008H	TMR1L	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
0009H	TMR1H	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
000AH	TMR0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
000BH	PA data	X	X	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
000CH	PB data	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
000DH	PC data	X	X	X	X	X	X	R/W	R/W	uuuu uuuu
000EH	PAC	X	X	R/W	R/W	R/W	R/W	R/W	R/W	uu11 1111
000FH	PBC	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	1111 1111
0010H	PCC	X	X	X	X	X	X	R/W	R/W	uuuu uu11
0011H	PA WUE	X	X	R/W	R/W	R/W	R/W	R/W	R/W	uu00 0000
0012H	PA IM	X	X	R/W	R/W	R/W	R/W	R/W	R/W	uu11 1111
0013H	PB IM	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	1111 1111
0014H	PC IM	X	X	X	X	X	X	R/W	R/W	uuuu uu11
0015H	MROM-BP	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0016H	SRAM-BP	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0017H	LCD_CTRL	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 1101
0018H	LCD_CMD	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
0019H	Decoder Control/ flag	X	R/W	R	X	R	X	R/W	R/W	uu0u uu01
001AH~ 002EH	Decoder Configuration Memory	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	uuuu uuuu
002FH	A/D-L	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0030H	A/D-H	X	X	X	X	X	R/W	R	R	uuuu u1uu
0031H	Buffer Status	R	X	R	R	R	R	R	R	0uuu uuuu
0032H	SPI-CONFIG	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	0111 1000
0033H	SPI-SPEED	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0034H	SPI-OUT3	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0035H	SPI-OUT2	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0036H	SPI-OUT1	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0037H	SPI-OUT0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	0000 0000
0038H	SPI-IN3	R	R	R	R	R	R	R	R	0000 0000
0039H	SPI-IN2	R	R	R	R	R	R	R	R	0000 0000
003AH	SPI-IN1	R	R	R	R	R	R	R	R	0000 0000
003BH	SPI-IN0	R	R	R	R	R	R	R	R	0000 0000

Note: "R" Read Only
"W" Write Only
"R/W" Read or Write
"X" N/A

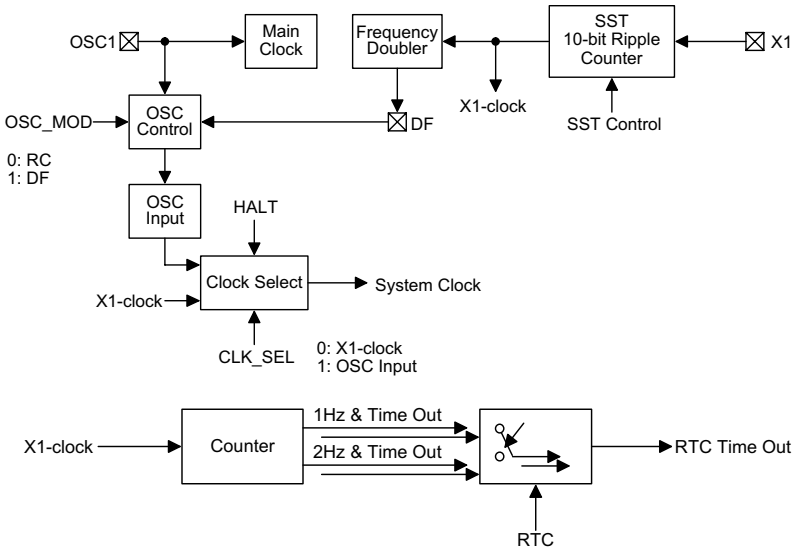
Configuration register

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0000H	Config.	HALT	CLK_SEL	OSC_MOD	LPM	RTC	BZ_CLK	MDUT	MGEN	0001 0000

Oscillator configuration

There are two clock source input pins on the chip, the main clock and the pager decoder input clock. The main clock is generated by an RC network. The system clock may be the OSC input or the X1-clock depending on bit "CLK_SEL". The pager decoder input clock comes from two external pins, X1 and X2. The frequency of the DF will be double that of the X1, X2 input clock. The OSC1 main clock will be generated from an RC network that needs an external resistor (see Application Circuit). The system clock may be X1-clock, DF or RC clock. If no higher frequency (RC) is needed, the external resistor between OSC1 and OSC2 can be removed. The system clock can be switched by bit "CLK_SEL". If "CLK_SEL"=0 (POR State), the system clock will be X1-clock, otherwise, with "CLK_SEL"=1, the OSC input clock will be the system clock. The clock switching function will assist software programmers to change the μC system clock within an adequate time if necessary. The "OSC_MOD" bit selects the OSC input clock to be either RC or DF. If "OSC_MOD" is set to "low" then the RC configuration is selected, otherwise the DF application is selected. The programmer should note that the condition of "CLK_SEL", "HALT" and "OSC_MOD" ensures that the system clock is working properly. It is recommended that the OSC clock source is either DF or RC. If DF and RC are necessary, it is required to switch the system clock to X1-clock before switching between DF and RC. Then switch the system clock back to the OSC input by using bit CLK_SEL if the switching action of DF and RC is complete. Before entering a HALT mode, the system clock must select X1-clock.

The HT9580 will generate two RTC frequencies, 1Hz and 2Hz respectively, determined by bit RTC. If the bit is logic low, the 1Hz RTC frequency will be selected, otherwise the 2Hz RTC frequency will be selected. The RTC counter is enabled/disabled by bit RTCEN and can be masked or not masked as determined by the bit RTCMSK of the interrupt control register



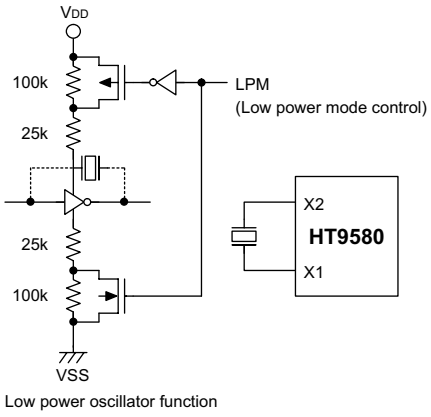
RTC block diagram

(0005H). If the RTC counter is enabled, the RTC counter will start to count. The RTC counter source clock is the X1-clock, so the X1 clock setting via by SPF12, SPF13 and SPF14 should be correct.

In order to guarantee that the system clock has started and stabilized, the SST (System Start-up Timer) provides an extra delay of system clock pulse when the system is powered up.

	1	0
RTC	Select 2Hz as the RTC	Select 1Hz as the RTC

The low power oscillator of the pager decoder input clock should be crystal type. The decoder subsystem low power oscillator, on the other hand, is of a crystal type which is designed with a power on start-up function to reduce the stabilization time of the oscillator. This start-up function is enabled by bit "LPM" which is initially set high at power on reset, and should be cleared to low so as to enable the low-power oscillator function. Then oscillator configuration is running in the low power mode.



low power oscillator

The system clock oscillator can be enabled/disabled by the register bit, "HALT". The system clock circuit is powered down, when the bit is set to high. When this bit is set high, the CPU is also stopped. When this bit is cleared low, the CPU core returns to its normal operation. After this is set HIGH by the software, it may also be cleared low when reset, interrupt ($\overline{IRQ}$ or $\overline{NMI}$), RTC timeout, and port wake-up conditions are met.

	0	1
HALT	System clock enable	System clock powered down

WDT-TMR (Watchdog Timer) register

The WDT is a 16-bit counter and its source from the X1-clock divided by 8. The counter is segmented as a 9-bit prescaler and a 7-bit user programmable counter. The input clock is first divided by 512 (9-stage) to get the nominal time-out period. The output of the 9-bit prescaler can then be divided by a 7-bit programmable counter to generate the longer watchdog time-out depending on the user's requirements. The 7-bit programmable counter is controlled by 3 register bits, WS0~2. The Watchdog Timer is enabled/disabled by a control bit WDTEN. To prevent the overflow of this Watchdog Timer, a clear-WDT operation should be executed before the timer overflows. The clear-WDT operation is to write any number to the register, CLRWDT (0002H). When the Watchdog Timer overflows (checked by bit 3 of 0006H "WDTOVFG"), the program counter is set to FFFC H and FFFD H to read the program start vector. The definitions of the control bits are listed below.

	1	0
WDTEN	Enable the Watchdog Timer	Disable the Watchdog Timer

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0001H	WDT-TMR	X	X	TMR0_PR1	TMR0_PR0	WDTEN	WS2	WS1	WS0	0000 0011
0002H	CLR WDT	X	X	X	X	X	X	X	X	uuuu uuuu

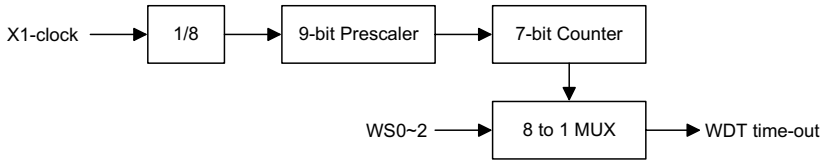
WDT-TMR (Watchdog Timer) register

The WDT 7-bit counter is programmed by bits WS0~WS2. The division ratio for the counter is listed in the table.

WS2	WS1	WS0	Division Ratio
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128

The other pair "TMR0_PR0" and "TMR0_PR1" are used to select the prescaler ratio for timer0. The definition is shown in the table.

TMR0_PR1	TMR0_PR0	TMR0 Prescaler Ratio
0	0	1/4
0	1	1/8
1	0	1/16
1	1	1/32



Buzzer generator registers

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0003H	BZ-L	BZL7	BZL6	BZL5	BZL4	BZL3	BZL2	BZL1	BZL0	0000 0000
0004H	BZ-H	BZH7	BZH6	BZH5	BZH4	BZH3	BZH2	BZH1	BZH0	0000 0000

The buzzer generator is composed of a 16-bit PFD counter and a duty cycle control. The counter value is set by two registers, namely BZ-H and BZ-L. The source for this generator may be the system clock or the X1-clock. The buzzer generator is enabled/disabled by the register bit "MGEN" in the configuration register(0000H). When this bit is set high, the buzzer generator is activated. There is another bit in the configuration register(0000H) which controls the buzzer output volume, bit "MDUT". If the bit is logic high, the output of the BZ will be modulated by the X1-clock. The clock source of the buzzer is selected by bit "BZ_CLK". When BZ_CLK=0, the clock source is the system clock. On the other hand, when BZ_CLK=1, the value of the selector will be the X1-clock.

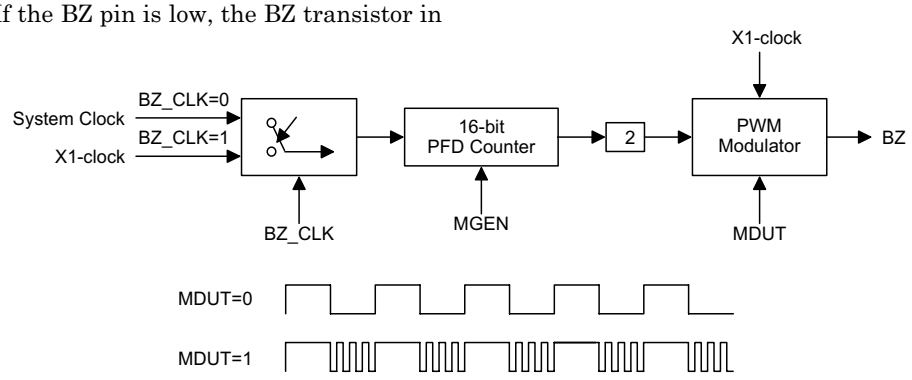
The truth table for enabling/disabling the buzzer generator is shown in the table.

	1	0
MGEN	Enable the buzzer generator	Disable the buzzer generator

When BZ-L and BZ-H are all 00H, the tone generator is disabled and BZ is high. The value of the frequency divider, ranges from 2 (BZ-L=01H, BZ-H=00H)~65536 (BZ-L=FFH, BZ-H=FFH). Writing to BZ-L only writes the data into a low byte buffer, while writing to BZ-H will write the high byte data and the contents of the low byte buffer into the PFD counter.

When the buzzer generator is disabled by clearing the "MGEN" bit in the configuration register (0000H), the BZ pin remains at its last state. If the BZ pin is low, the BZ transistor in the application circuits is always active. Therefore it is recommended that both BZ-L and BZ-H be cleared and that the "MGEN" bit in the configuration register (0000H) also be cleared, when it is desired to disable or stop the buzzer.

The output of the 16-bit PFD counter is divided by 2 to generate a BZ output with or without modulation. For example, if the desired output of BZ is 1.6kHz with modulation and the frequency source is X1-clock (76.8kHz), then the value of 16-bit PFD counter is set to BZ-L=17H, BZ-H=00H and "MDUT" is set high.



Interrupt registers

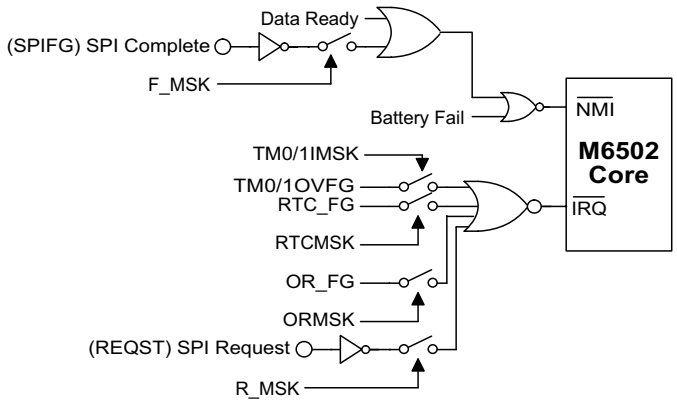
Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0005H	INT ctrl	0	F_MSK	R_MSK	RTCEN	ORMSK	RTCMSK	TM1IMSK	TM0IMSK	0110 1111
0006H	INT flag	0	RTC_FG	DR_FG	BF_FG	WDTOVFG	OR_FG	TM1OVFG	TM0OVFG	0000 0000

There are two interrupts for the HT9580: a Non-Mask Interrupt (NMI) and a generic interrupt request (IRQ). The data ready interrupt and battery fail interrupt share the NMI call location. Which interrupt occurred can be determined by checking bit BF_FG and the data ready interrupt bit DR_FG or SPI complete flag SPIFG (in SPI-CONFIG register). DR_FG is the data ready interrupt indication bit. When a valid call is detected, data begins to transfer. Either one call is terminated or a message buffer is full or one batch is over but the message is not terminated, the data ready interrupt will occur and DR_FG is set high. The DR_FG bit should be cleared low by the μC software after a data ready condition has occurred.

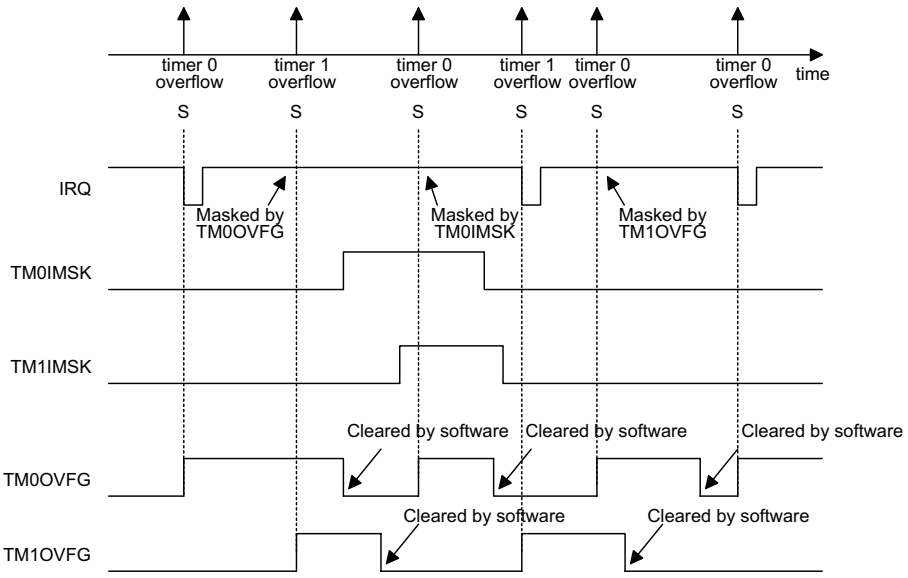
A battery fail condition is triggered by a high to low transition on pin $\overline{BAF}$ and will set the battery fail interrupt request flag BF_FG to logic high. For details, refer to the POCSAG Decoder section. The sources for the IRQ are timer 0 overflow, timer 1 overflow, out-of-range status changes, RTC time out and SPI request. The five interrupt sources all could be masked, but the five corresponding interrupt flags will still be set when the interrupt conditions are met. All the five flags are readable/writeable. When an interrupt condition is met, a flag will be set. If an interrupt routine is performed, the software should check which flag is set to high then determine what kind of interrupt source occurred. The WDTOVFG is the flag for the

Watchdog Timer overflow and RTC_FG is an indicator for the RTC time out interrupt request flag. The OR_FG will be set high when an out-of-range status from low to high or high to low transition occurs. Those flags such as TM0OVFG, TM1OVFG, BF_FG, DR_FG, OR_FG and RTC_FG should be cleared by the software after they are activated.

	1	0
RTCEN	RTC counter is enabled	RTC counter is disabled
RTCMSK	RTC interrupt is masked	RTC interrupt is not masked
TM0IMSK	Timer 0 overflow interrupt is masked	Timer 0 overflow interrupt is not masked
TM1IMSK	Timer 1 overflow interrupt is masked	Timer 1 overflow interrupt is not masked
ORMSK	Out-of-range interrupt is masked	Out-of-range interrupt is not masked
F_MSK	Flex data exchange complete interrupt is masked	Flex data exchange complete interrupt is not masked
R_MSK	Interrupt from Flex decoder request is masked	Interrupt from Flex decoder request is not masked
TM0OVFG	Timer 0 overflows	No timer 0 overflow
TM1OVFG	Timer 1 overflows	No timer 1 overflow
WDTOVFG	Watchdog Timer has overflown	No Watchdog Timer overflow
BF_FG	Battery fail request	No battery fail request
DR_FG	Data ready request	No data ready request
OR_FG	Out-of-range request	No out-of-range request
RTC_FG	RTC interrupt request	No RTC interrupt request



Block diagram of NMI and IRQ



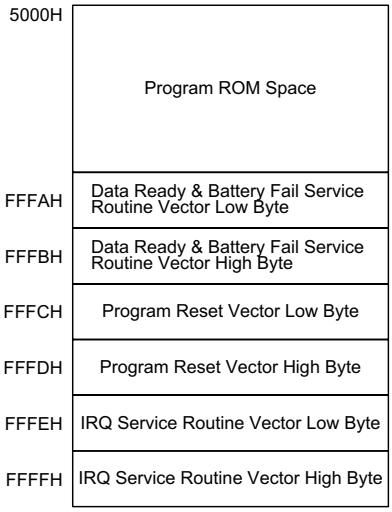
Timer0 and Timer1 timing diagram

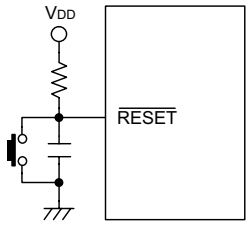
Reset conditions

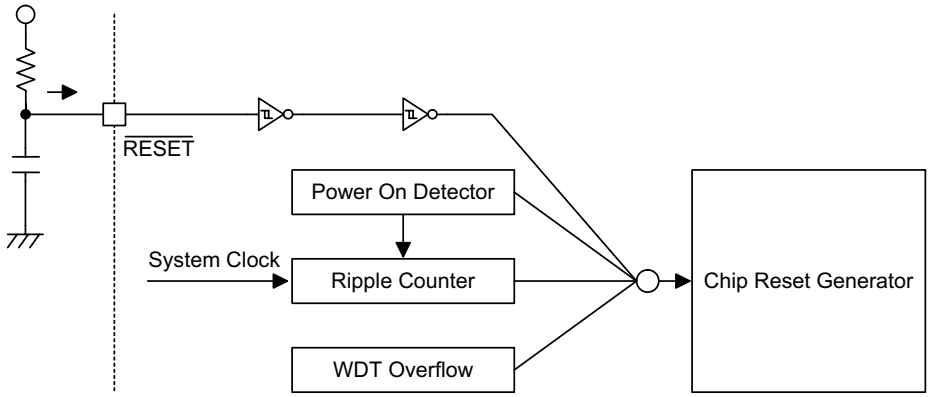
The HT9580 will reset the whole chip when the following conditions are met:

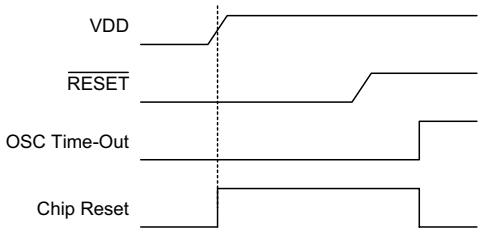
- Power On
- The external $\overline{\text{RESET}}$ pin is edge triggered
- The WDT overflows

The input is used to reset the μC, the program counter is loaded with the restart vector from locations FFFC (low byte) and FFFD (high byte). This is the start location for program control. The voltage detector (Ex: HT99S201) is needed for HT9580 application to ensure system reliability. (See application circuits)

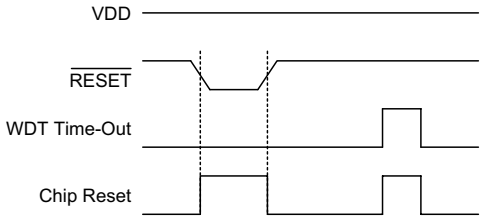








Power on Reset Timing



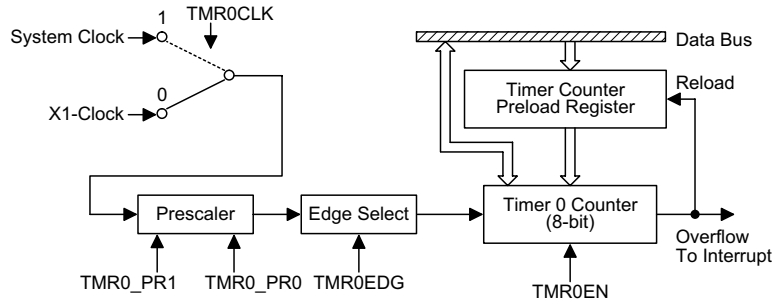
RESET active and WDT Time-out

Timer registers

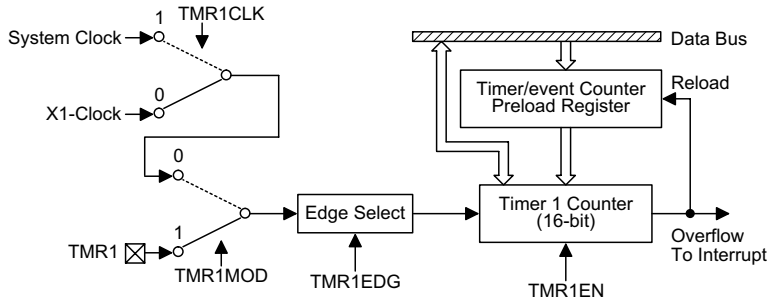
Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0007H	TMRC	TMR1MOD	X	TMR1CLK	TMR0CLK	TMR1EDG	TMR0EDG	TMR1EN	TMR0EN	0u00 0000
0008H	TMR1L	TM1D7	TM1D6	TM1D5	TM1D4	TM1D3	TM1D2	TM1D1	TM1D0	uuuu uuuu
0009H	TMR1H	TM1D15	TM1D14	TM1D13	TM1D12	TM1D11	TM1D10	TM1D9	TM1D8	uuuu uuuu
000AH	TMR0	TM0D7	TM0D6	TM0D5	TM0D4	TM0D3	TM0D2	TM0D1	TM0D0	uuuu uuuu

In addition to the Watchdog Timer, the HT9580 provides two timers: an 8-bit timer (timer 0) and one 16-bit timer (timer 1). Those two timers are controlled and configured by the register TMRC. Both timers are programmable up-count counters whose clocks may be derived from the X1-clock (32.768kHz, 76.8kHz or 153.6kHz). To program the timers, TMR0, TMR1L, and TMR1H should be written with a start value. When the timers are enabled, they will count-up from the start value. If the timers overflow, corresponding interrupts will be generated. When the timers are disabled, the counter contents will not be reset. To reset the counter contents, the software should write the start value again. Since timer1 is a 16-bit counter, it is important to note the method of writing data to both TMR1L and TMR1H. Writing to TMR1L only writes the data into a low byte buffer, while writing to TMR1H will simultaneously write the high byte data and the contents of the low byte buffer into the Timer Counter preload register (16-bit). Note that the Timer counter preload register contents are changed by a TMR1H write operation while writing to TMR1L does not change the contents of the preload register. Reading TMR1H will also latch the contents of TMR1L into the byte buffer to avoid false timing problem. Reading TMR1L return the contents of the low byte buffer. In other words, the low byte of the timer counter cannot be read directly. It must first read TMR1H to latch the low byte contents of the timer counter into the buffer. TMRC is the timer counter control register, which defines the timer counter options. The timer1 clock source can be selected from either an internal clock or an external input clock by bit TMR1MOD of the TMRC register. Timer0/timer1 can also select its clock source by bits TMR0CLK/TMR1CLK. TMRC as shown in the table.

Labels (TMRC0 and TMRC1)	Bits	Function
TMR0EN, TMR1EN	0 1	Enable/disable timer counting (0=disable; 1=enable)
TMR0EDG, TMR1EDG	2 3	Define the TMR0 and TMR1 active edge (0=active on low to high; 1=active on high to low)
TMR0CLK	4	Select TMR0 clock source (0=X1-clock; 1=system clock)
TMR1CLK	5	Select TMR1 clock source if internal clock input is selected (0=X1-clock; 1=system clock)
TMR1MOD	7	Define the TMR1 operation mode (0=internal clock input; 1=external clock input)



Timer 0 block diagram



Timer 1 block diagram

I/O port configuration registers

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
000BH	PA data	X	X	PA5	PA4	PA3	PA2	PA1	PA0	uu11 1111
000CH	PB data	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	1111 1111
000DH	PC data	X	X	X	X	X	X	PC1	PC0	uuuu uu11
000EH	PAC	X	X	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0	uu11 1111
000FH	PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0	1111 1111
0010H	PCC	X	X	X	X	X	X	PCC1	PCC0	uuuu uu11
0011H	PA WUE	X	X	PAWUE5	PAWUE4	PAWUE3	PAWUE2	PAWUE1	PAWUE0	uu00 0000
0012H	PA IM	X	X	PAIM5	PAIM4	PAIM3	PAIM2	PAIM1	PAIM0	uu11 1111
0013H	PB IM	PBIM7	PBIM6	PBIM5	PBIM4	PBIM3	PBIM2	PBIM1	PBIM0	1111 1111
0014H	PC IM	X	X	X	X	X	X	PCIM1	PCIM0	uuuu uu11

The HT9580 has three general purpose I/O ports. The I/O cell structures can be configured. Details are shown in the table.

Port A

Port A is a general-purpose I/O port. The PAC register controls the data directions for port A. When set as input data type, this port has a wake-up capability and the input cell structures are Schmitt trigger types. While in a "HALT" condition, a falling edge signal on Port A can wake-up the μC. In addition, the input cell structures can be configured as pull-high or non-pull-high. When set as an output data type, the output structures are CMOS outputs.

	1	0
PA	Pin output logic high	Pin output logic low
PAC	Input pin	Output pin
PAWUE	Pin has wake-up capability	Pin has no wake-up capability
PAIM	CMOS input structure with pull-high resistor	CMOS input structure without pull-high resistor

Port B

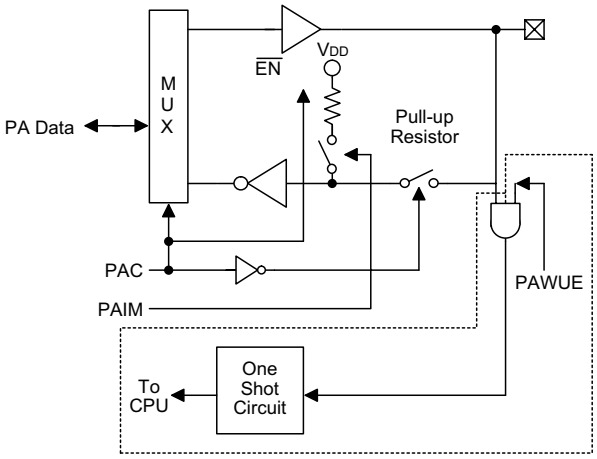
Port B is a general-purpose I/O port controlled by the PBC register. The PBIM register controls the input cell structures: normal CMOS inputs or CMOS inputs with pull-high resistors.

	1	0
PB	Pin output logic high	Pin output logic low
PBC	Input pin	Output pin
PBIM	CMOS input structure with pull-high resistor	CMOS input structure without pull-high resistor

Port C

This is a general-purpose I/O port controlled by the PCC register. The PCIM register controls the input cell structures: normal CMOS inputs or CMOS inputs with pull-high resistors.

	1	0
PC	Pin output logic high	Pin output logic low
PCC	Input pin	Output pin
PCIM	CMOS input structure with pull-high resistor	CMOS input structure without pull-high resistor



Port A I/O Structure

Mask ROM (Character ROM) bank point register

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0015H	MROM-BP	BP_MODM1	BP_MODM0	M_BP5	M_BP4	M_BP3	M_BP2	M_BP1	M_BP0	0000 0000

The Mask ROM bank point register can switch between the internal 2 Mbits Mask ROM or an external 2 Mbits Mask ROM space. The selection table is based on the table. The space size of each Mask ROM bank is 8 Kbytes. The bits BP_MODM1 and BP_MODM0 define whether internal or external Mask ROM devices are used. (BP_MODM1, BP_MODM0)=(0, 1), selects the internal Mask ROM device. (BP_MODM1, BP_MODM0)=(1, 0), selects the external Mask ROM device. The internal/external Mask ROM can switch from bank 0 to bank 31 by software programming. In addition, the address range of the internal/external Mask ROM will all range from 1000H to 2FFFH.

The Mask ROM bank point register selection table is shown in the table.

BP_MODM1	BP_MODM0	M_BP5	M_BP4	M_BP3	M_BP2	M_BP1	M_BP0	BP Value	Memory Area
0	0	X	X	X	X	X	X	X	Reserved
0	1	0	0	0	0	0	0	0	Internal 2 Mbits Mask ROM (low 8 Kbytes)
0	1	↓						↓	↓
0	1	0	1	1	1	1	1	31	Internal 2 Mbits Mask ROM (High 8 Kbytes)
0	1	1	0	0	0	0	0	32	Reserved
0	1	↓						↓	Reserved
0	1	1	1	1	1	1	1	63	Reserved
1	0	0	0	0	0	0	0	0	External 2 Mbits Mask ROM (low 8 Kbytes)
1	0	↓						↓	↓
1	0	0	1	1	1	1	1	31	External 2 Mbits Mask ROM (High 8 Kbytes)

If the internal 2 Mbits Character ROM is placed as shown in the figure and the software programmer obtains a start address from CNS (Taiwan) code or a GB (China) code, A0~A17. The following steps will map from the start address to the bank point register, then the hardware address decode circuit will point to the real 2 Mbits space.

- Step 1

  The formula obtains A0~A17 from the received GB or CNS code.

- Step 2

  Set (BP_MODM1, BP_MODM0)=(0, 1)

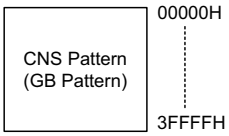
- Step 3

  Assign correct "M_BP0"~"M_BP5" as shown:

  - A13→M_BP0
  - A14→M_BP1
  - A15→M_BP2
  - A16→M_BP3
  - A17→M_BP4

- Step 4

  Adding $1000 H to A12~A0 to get new HEX value $\$B_3B_2B_1B_0$ H.



- Step 5

  The following example will load 32 bytes continuous (one Chinese word) pattern from the internal Character ROM and store them to the start address $\$C_3C_2C_1C_0$ H (if absolute index addressing mode is used).

```
LDX   #00H
LDY   #00H
READ:
LDA   $B3B2B1B0, X
STA   $C3C2C1C0, Y
INX
INY
CPX   #20H
BNZ   READ
```

	0	0	0	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
+	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	RA13	RA12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

B_3(0,0,RA13,RA12) B_2(A11,A10,A9,A8) B_1(A7,A6,A5,A4) B_0(A3,A2,A1,A0)

SRAM bank point register

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0016H	SRAM-BP	BP_MODS1	BP_MODS0	S_BP5	S_BP4	S_BP3	S_BP2	S_BP1	S_BP0	0000 0000

The SRAM bank point register can switch to either external 256 Kbytes or internal 32 Kbytes SRAM space. The selection table is based on the following table. The space size of each SRAM bank is 8 Kbytes. Bits BP_MODS1 and BP_MODS0 define whether internal or external SRAM devices are used. (BP_MODS1, BP_MODS0)=(0, 1), is for internal SRAM devices. (BP_MODS1, BP_MODS0)=(1, 0), is for external SRAM devices. The internal SRAM would switch from bank 0 to bank 3 and the external SRAM would switch from bank 0 to bank 31 by software programming. In addition, the address range of the internal/external SRAM will all range from 3000H to 4FFFH.

BP_MODS1	BP_MODS0	S_BP5	S_BP4	S_BP3	S_BP2	S_BP1	S_BP0	BP Value	Memory Area
0	0	X	X	X	X	X	X	X	Reserved
0	1	0	0	0	0	0	0	0	Internal 32 Kbytes SRAM (Low 8 Kbytes)
0	1	↓						↓	↓
0	1	0	0	0	0	1	1	3	Internal 32 Kbytes SRAM (High 8 Kbytes)
0	1	0	0	0	1	0	0	4	Reserved
0	1	↓						↓	Reserved
0	1	1	1	1	1	1	1	63	Reserved
1	0	0	0	0	0	0	0	0	External 256 Kbytes SRAM (Low 8 Kbytes)
1	0	↓						↓	↓
1	0	0	1	1	1	1	1	31	External 256 Kbytes SRAM (High 8 Kbytes)

LCD control and data register

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0017H	LCD_CTRL	LCD-CHIP1	LCD-CHIP0	LCD-CLK	CLK-MOD	LCD-CS1	LCD-CS0	LCD-A0	LCD-WRB	0000 1101
0018H	LCD_CMD	LCD_D7	LCD_D6	LCD_D5	LCD_D4	LCD_D3	LCD_D2	LCD_D1	LCD_D0	uuuu uuuu

The LCD control and command registers are used for LCD driver interface. There are three kinds of LCD driver chips available for the HT9580. These LCD drivers are SED15X(KSX) series, Motorola LCD driver chip MC141X series and HD66410 respectively according to the following "LCD-CHIP0" and "LCD-CHIP1" bit table settings. The combination of the LCD_CMD and LCD-CTRL registers can control the SED15X(KSX), MC141X series or HD66410 LCD drivers. Bits LCD-CS0/1 of the LCD-CTRL register corresponds to the chip select pin of the LCD driver. The bit "LCD-CS0" is used to control the master LCD driver chip while "LCD-CS1" is for the slave LCD driver chip. Both bits are active low. The bit "CLK_MOD" is used to enable or disable the pinout of LCD_CL. If the bit is set low, the clock output of pin LCD_CL will be disabled and LCD_CL output is low, otherwise the LCD_CL clock will be set according to the following Truth Table.

"LCD-CHIP0" and "LCD-CHIP1" Truth Table

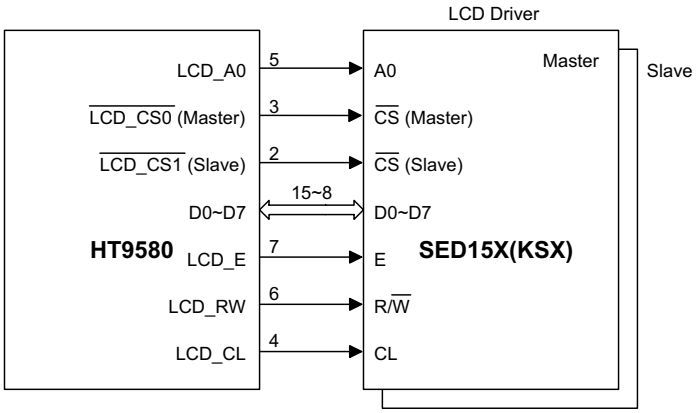
	LCD-CHIP0="0"	LCD-CHIP0="1"
LCD-CHIP1="0"	SED15X(KSX) series LCD driver is selected	MC141X series LCD driver is selected
LCD-CHIP1="1"	HD66410 LCD driver is selected	N/A

"LCD_CL" Truth Table

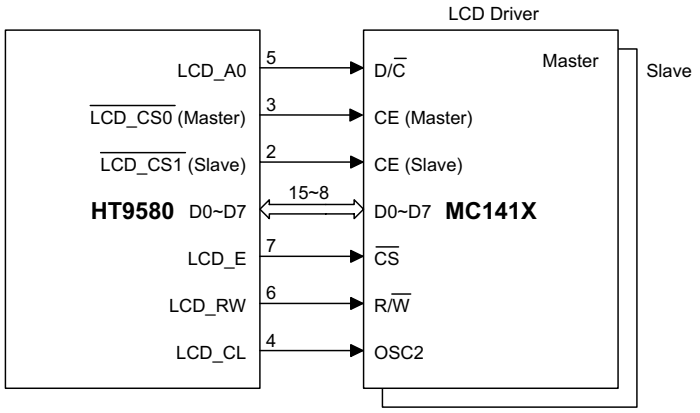
	LCD-CHIP0="0"	LCD-CHIP0="1"
LCD-CHIP1="0"	LCD_CL: 2 kHz output	LCD_CL: If "LCD-CLK"=0, 32 kHz output If "LCD-CLK"=1, X1-clock output
LCD-CHIP1="1"	LCD_CL: 10.9kHz output	N/A

The following is a comparison table of the HT9580 pin description between the SED15X (KSX) series and the MC141X series LCD driver.

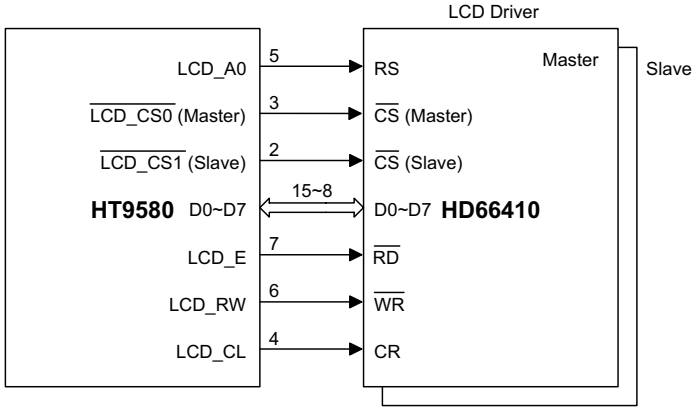
HT9580 (Pin)	**SED15X(KSX) Series**		**MC141X Series**	
LCD_A0	A0	Data/command select input. A0=0: Display control data on D0~D7 A0=1: Display data on D0~D7	D/C	This input pin acknowledges valid data on D0~D7. If high then D0~D7 contains display data, if low D0~D7 contains command data.
LCD_CS0	$\overline{CS}$ (Master)	Active low chip select input. (Master)	CE (Master)	When high, enables the control pins on the driver. (Master)
LCD_CS1	$\overline{CS}$ (Slave)	Active low chip select input. (Slave)	CE (Slave)	When high, enables the control pins on the driver. (Slave)
D0~D7	D0~D7	8-bit, tristate, bidirectional I/O bus.	D0~D7	Bidirectional bus for data/command transfer.
LCD_E	E	Enable clock input	$\overline{CS}$	This pin is normal low clock input. Data on D0~D7 is latched at the falling edge of CS.
LCD_RW	R/$\overline{W}$	Read/write input	R/$\overline{W}$	To read the display data RAM or the internal status, pull this pin high. The pin low indicates a write operation.
LCD_CL	CL	External clock input. (2kHz output from HT9580)	OSC2	Oscillator input for external clock is used. (32kHz or X1-clock output from HT9580 as determined by the "LCD-CLK").



The application circuit when bit "LCD-CHIP1" = 0 and "LCD-CHIP0" = 0



The application circuit when bit "LCD-CHIP1" = 0 and "LCD-CHIP0" = 1



The application circuit when bit "LCD-CHIP1" = 1 and "LCD-CHIP0" = 0

LCD Driver Chip Selection	Application	Note
LCD-CHIP0="0" LCD-CHIP1="0"	SEDX(EPSON) series LCD driver at 68 family MPU application mode.	RESET is low active
	KSX(SAMSUNG) series LCD driver at 68 family MPU application mode.	Pin options set as 68 family MPU application mode.
LCD-CHIP0="1" LCD-CHIP1="0"	MC14X(MOTOROLA) series LCD driver.	
LCD-CHIP0="0" LCD-CHIP1="1"	HD66410(HITACHI) series LCD driver.	
	SEDX(EPSON) series LCD driver at 80 family MPU application mode.	RESET is high active
	KSX(SAMSUNG) series LCD driver at 80 family MPU application.	Pin options set as 80 family MPU application mode.
LCD-CHIP0="1" LCD-CHIP1="1"	N/A	

Power down operation – HALT

The HALT mode is initiated by setting the configuration register bit HALT high and results in the following ...

The system clock turns off, the low power pager sub-clock, LCD driver, pager decoder and RTC keep on running.

The contents of the on-chip RAM and of the register remain unchanged.

As the WDT and the WDT prescaler depend on software control, the WDT will continue to count when the ″HALT″ bit is set high.

All the I/O ports remain in their original status.

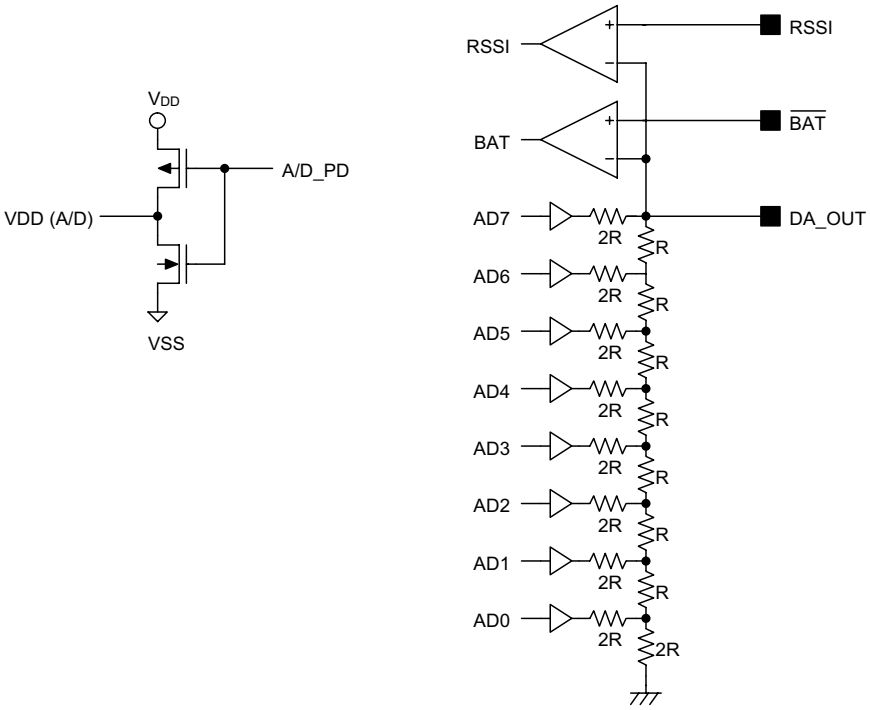
The system can quit from the HALT mode by an external reset, an interrupt, an external falling edge signal on port A or an RTC time out.

A/D registers

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
002FH	A/D-L	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0000 0000
0030H	A/D-H	X	X	X	X	X	A/D_PD	RSSI	BAT	uuuu u1uu

The HT9580 has one internal 8-bit A/D converter which can measure the battery voltage and the RSSI input signal from the IF of the RF circuit. The AD0~AD7 is the digital input of the A/D converter and the analog outputs to the pin named DA_OUT. Bit BAT of the AD-H register (0030H) is the output of the comparator. Its input at the "-" terminal is from the A/D output and the "+" terminal comes from the input pin $\overline{\text{BAF}}$. The bit RSSI of AD-H register (0030H) is the output of another comparator. Its input at "-" terminal is from the A/D output and "+" terminal comes from the input pin RSSI. The software can detect the battery voltage and the RSSI signal by writing to the bits AD0 ~AD7 (002FH) and reading the bits BAT, RSSI (0030H).

Bit "A/D_PD" is used for the A/D power down control. If this bit is logic high, the A/D will be in the power down mode. Otherwise, the A/D is in the normal condition. For details see the following figure.

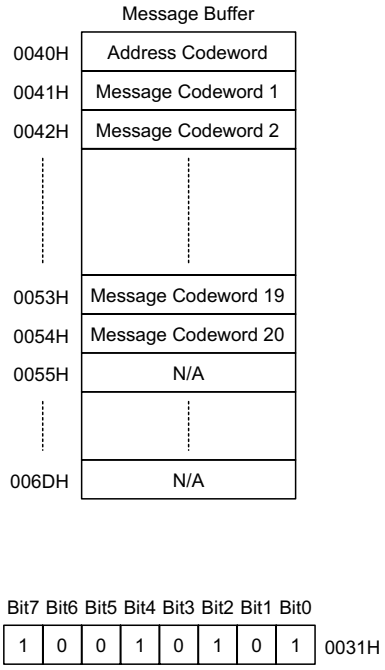


The configuration of the 8-bit A/D converter and power down control

Buffer status register

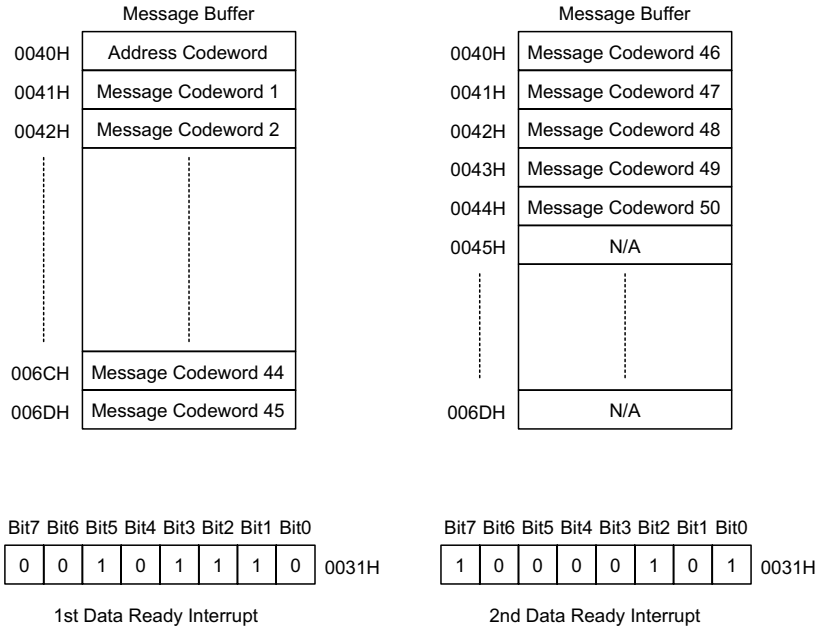
Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	State on POR
0031H	Buffer Status	MSG_END	X	count_5	count_4	count_3	count_2	count_1	count_0	0uuu uuuu

The buffer status register will relate to the μC the status of the message buffer when the data ready request interrupt occurred. The "MSG_END" bit will be set high when the data (including address codeword and message codeword) is at the end of this data ready interrupt call. The valid data length of the message buffer is determined by bit count_0 to count_5. If "MSG_END" is low, the data length is more than 46 or data is not at the end, the μC should wait for the next data ready interrupt until the bit "MSG_END" is set high. Example 1: if the data read from 0031H is "95H" when a new data ready interrupt occurred, it means the total data length is 21 including the address codeword in this call and the message is terminated (bit "MSG_END" =1). The figure below illustrates example 1.



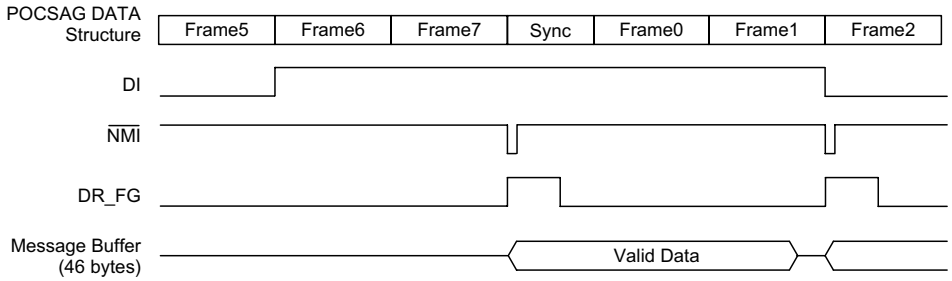
Example 1

Example 2: if the data read from 0031H is "2EH" when a new data ready interrupt occurred, that means the data length of this call is more than 46 and the next data ready interrupt will occur. If the next interrupt occurs and the contents of 0031H is "85H", the result are shown in the following figure. The programmer should note that the information on the message buffer must be read out before the next continuous codeword arrives. Otherwise the data on the message will be overwritten.



Example 2

The data ready interrupt will generate when message is terminated, synchronization code word is received or buffer is full. The following figure shows the typical operation.



The Message Buffer Timing Chart

SPI configure register

Address	Register Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	State on POR
0032H	SPI-CONFIG	S/M	LEN1	LEN0	REQST	SPIFG	CLK_EDG	SPI_EN	START	0111 1000

- S/M: Slave/master mode selection
  When S/M is 0", HT9580 is in the master mode. Otherwise, HT9580 is in the slave mode.

	0	1
S/M	Master mode (SCK is output)	Slave mode (SCK is input)

- LEN0, LEN1: Data length
  The LEN0 and LEN1 will determine the data length between exchange.

LEN1	LEN0	Data Length (Bit)
0	0	4
0	1	8
1	0	16
1	1	32

- REQST: SPI request (read only)
  When FLEX™ decoder wants to exchange data with HT9580, the REQST will be cleared to low then set to high by hardware after SPI command "START" was issued.
- SPIFG: SPI complete flag
  0 (clear): Data transfer to external device has been completed.
  1 (set): No valid completion of data transfer. The bit is cleared by hardware and set by software.
- CLK_EDG: Data sampling edge
  The CLK_EDG will determine the valid MISO and MOSI sampling edge of SCK clock.

	0	1
CLK_EDG	Rising edge	Falling edge

- SPI_EN: The SPI enable

	0	1
SPI_EN	When the SPI circuit is disabled, the POCSAG decoder I/O pins will be enabled	The SPI circuit and SPI I/O pins will be enabled

- START: The data exchange start or not

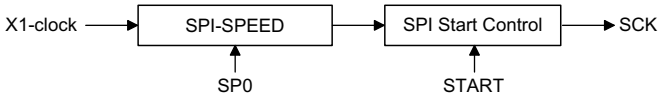
	0	1
START	No data exchange	Data exchange start

When the bit is set by software, the SPI data exchange will start. After the first bit data exchange is completed, the START bit will clear to low again by hardware.

SPI SPEED register (write only)

Address	Register Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	State on POR
0033H	SPI-SPEED	X	X	X	X	X	X	X	SP0	0000 0000

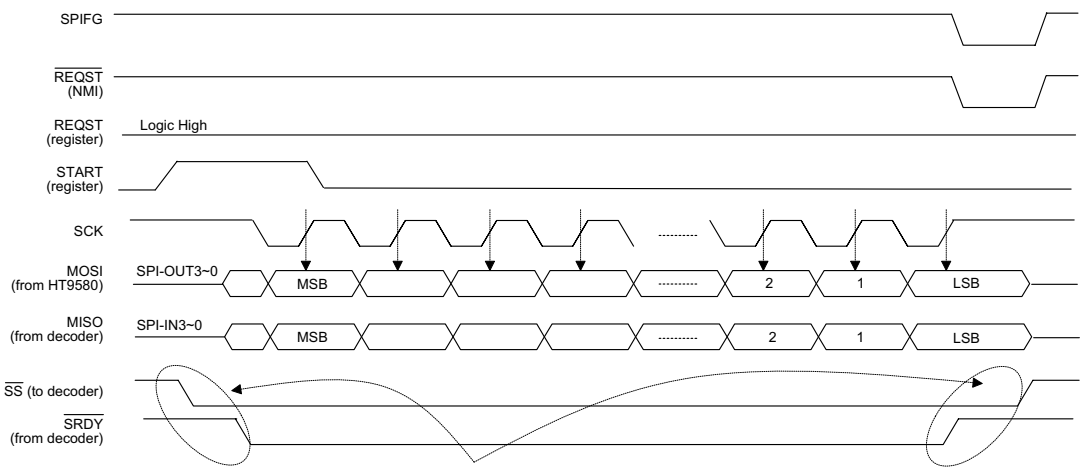
The register will determine the SCK clock frequency of SPI. When SPEED register are 00H, the SCK clock output is high. The frequency of the SCK is X1-clock when SP0=1.



SPI output buffer register (write only)

Address	Register Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	State on POR
0034H	SPI-OUT3	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0035H	SPI-OUT2	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0036H	SPI-OUT1	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0037H	SPI-OUT0	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000

The SPI-OUT3~0 are used when transmitting data on the serial bus. Only valid data is written to the register SPI-OUT3~0 and START initiating will begin the SPI data transmission from HT9580 to FLEX^{TM} decoder. After completion of the 4-byte data transfer, the SPIFG status bit will be cleared to low and NMI will occur. The bit7 of SPI-OUT3 is MSB and bit0 of SPI-OUT0 is LSB.

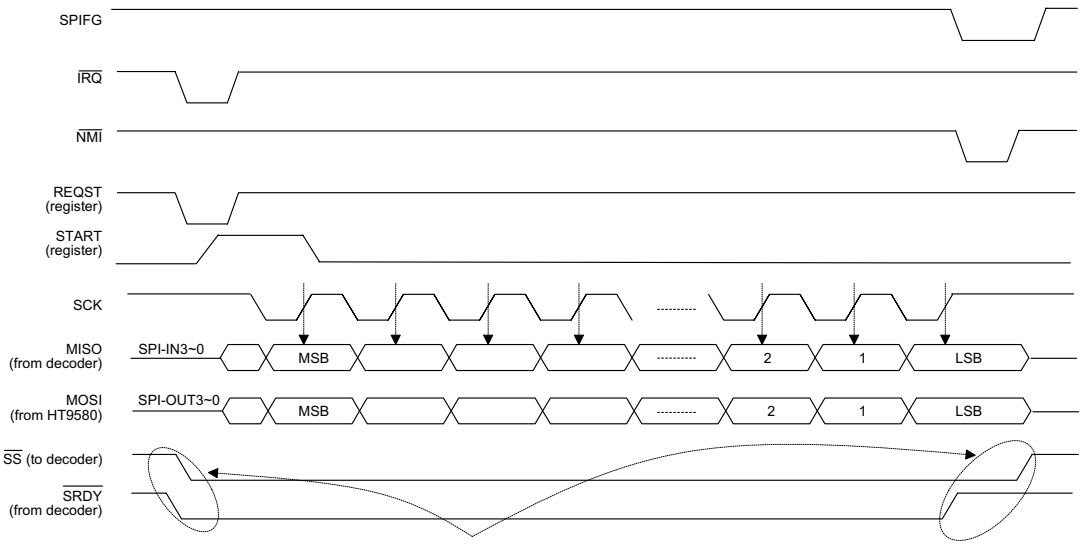


SPI packet exchange initiated by the HT9580

SPI input buffer register (read only)

Address	Register Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	State on POR
0038H	SPI-IN3	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
0039H	SPI-IN2	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
003AH	SPI-IN1	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000
003BH	SPI-IN0	D7	D6	D5	D4	D3	D2	D1	D0	0000 0000

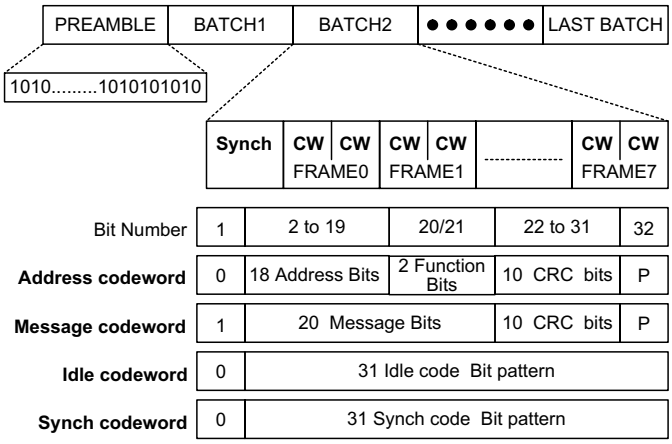
The SPI-IN3~0 are used when receiving data on the serial bus. When SPI transmits only valid data is written to the register SPI-OUT3~0, START will initiate the SPI data transmission between HT9580 and FLEX™ decoder. If FLEX™ decoder wants to exchange data with HT9580, then the request flag "REQST" will be cleared to low and IRQ will be generated. After completion of the 4-byte data transfer, the SPIFG status bit will be cleared to low and NMI will occur. The bit7 of SPI-IN3 is MSB and bit0 of SPI-IN0 is LSB.



SPI packet exchange initiated by decoder

The POCSAG paging code

A transmission using the "CCIR Radio paging Code No.1" (POCSAG code) is generated in accordance with the following rules (see the following Figure).



POCSAG code structure

The transmission is started by sending a preamble, consisting of at least 576 continuously alternating bits (10101010...). The preamble is followed by an arbitrary number of batch blocks. Only complete batches are transmitted.

Each batch comprises 17 code-words of 32 bits each. The first code-word is a synchronization code-word with fixed pattern. The sync word is followed by 8 frames (0~7) of 2 code-words each, containing message information. A code-word in a frame can either be an address, message or idle code-word.

Idle code-words also have fixed patterns and are used to fill empty frames or separate messages.

Address code-words are identified by an MSB of logic 0 and are coded as shown in the POCSAG code structure figure. A user address or RIC (Receiver Identity Code) consists of 21 bits. Only the upper 18 bits are encoded in the address code-word (bits 2 to 19). The lower 3 bits designate the frame number in which the address is transmitted.

Four different call types can be distinguished on each user address. The call type is determined by two functional bits in the address code-word (bits 20 and 21). The POCSAG standard recommends the use (in Taiwan) of combinations of data formats and function bits, as shown in the following table. Other combinations will be set by SPF16~SPF19.

Bit 20 (MSB)	Bit 21 (LSB)	Call Type	Data Format
0	0	Numeric	4-bit package
0	1	Alert only	—
1	0	Alert only	—
1	1	Alpha-numeric	7-bit package

Data formats and function bits

Alert-only calls consist of a single address code-word. Numeric and alphanumeric calls have message code-words following the address.

Message code-words are identified by an MSB of logic 1. The message information is stored in a 20-bit field (bits 2 to 21). The data format is determined by the call type: 4 bits per digit for numeric message and 7 bits per (ASCII) character for alphanumeric messages. Each code-word is protected against transmission errors by 10 CRC check bits (bit 22 to 31) and an even parity bit (bit 32).

This permits correction of a maximum of 2 random errors or up to 4 errors in a burst of 4 bits (a 4-bit burst error) per code-word.

- Error correction

Item	Description
Address code-word	two random bit errors, or 4-bit burst errors (optional)
Message code-word	two random bit errors, or 4-bit burst errors (optional)

Error correction

In the HT9580, error correction methods have been implemented as shown in the table above. Two random bit error corrections are the default for both address and message code-word. In another method, burst error correction can be switched by SPF programming. Up to 4 erroneous bits in a 4-bit burst can be corrected. The error type detected for each code-word is identified in the message data output to the microcontroller, allowing rejection of calls with too many errors.

- Operating states
  - ON status
  - STANDBY status

  The operating state is determined by control address (0019H) bit 0 and monitored by bit 3 of address (0019H).

  Truth table for decoder operating status

$\overline{ON}$	Operating Status
0	On state
1	STANDBY state

- On status

  In the ON status, the decoder pulses the receiver, quick charge and PLL enable outputs (respectively BS1, BS2 and BS3) according to the code structure and the synchronization algorithm. Data received serially at the data input (DI) is processed for call receipt.

- STB status

  In the STB status the decoder will neither activate the receiver, quick charge or PLL enable outputs, nor process any data at the data input. The crystal oscillator remains active to permit communication with the microcontroller.

- Battery saving

  To increase battery efficiency, reception and decoding of an address code-word is stopped as soon as the uncorrected address field differs by more than 3-bits from the enabled RICs. If the next code-word has to be received again, the receiver is re-enabled, thus observing the programmed establishment times t_{BS1} , t_{BS2} and t_{BS3}.

- Data reception and buffer

  Reception of a valid paging call is signaled to the microcontroller by means of an interrupt signal. The received address and message code-word can then be read via a 46 bytes message buffer (from 0040H to 006DH) for decoder data message. If the μC did not read the previous message within one code-word time from the message buffer, the message buffer data will be overwritten.

- Bit rates

  The HT9580 can be configured for data rates of 512, 1200 or 2400 bits/s by SPF programming. These data rates are derived from 32.768kHz, 76.8kHz or 153.6kHz oscillator frequencies.

- Input data processing

  The input data is noise filtered by means of a digital filter. Data is sampled at 16 times the data rate and averaged by majority decision.

The filtered data is used to synchronize an internal clock generator by monitoring transitions. The recovered clock phase can be adjusted in steps of 1/8, 3/32, 1/16, or 1/32 bit period per received bit.

All step size are used when bit synchronization has not been achieved, the smallest when a valid data sequence has been detected.

- Erroneous code-words

  Upon receipt of erroneous uncorrectable code-words, call termination occurs according to the conditions given below:

SPF08	SPF09	Description
0	X	Any two consecutive code-words or the code-word directly following the address code-word in error
1	0	Any single code-word in error
1	1	Any two consecutive code-words in error

- Message receiving mode

  The receiving message mode (numeric or alpha-numeric) depends on bits SPF16~SPF19. If one of these bits from SPF16~SPF19 is cleared to low, the decoder will be in numeric package receiving mode. Otherwise, the decoder is in the alphanumeric receiving mode. An example is shown below:

	Function Bits		Message Receiving Format
	Bit 20 (MSB)	Bit 21 (LSB)	
SPF16=0	0	0	Numeric (4-bit)
SPF17=0	0	1	Numeric (4-bit)
SPF18=1	1	0	Alphanumeric (7-bit)
SPF19=1	1	1	Alphanumeric (7-bit)

The decoder data output format is determined by the value SPF16~SPF19. When it is logic low, the 4-bit (numeric) package will be selected. Otherwise, the 7-bit (alphanumeric) package is selected. The following tables illustrate the above two different conditions.

Message code-word on the message buffer (numeric receiving mode)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Error Flag	0	0	0	D3	D2	D1	D0

Message code-word on the message buffer (alphanumeric receiving mode)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Error Flag	D6	D5	D4	D3	D2	D1	D0

- Synch word indication

  The synch word recognized by the HT9580 is the standard POCSAG synchronization code-word as shown in the following table.

Bit No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit	0	1	1	1	1	1	0	0	1	1	0	1	0	0	1	0

Bit No.	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Bit	0	0	0	1	0	1	0	1	1	1	0	1	1	0	0	0

• Idle word indication

The idle word recognized by the HT9580 is a standard POCSAG idle code-word as shown in the following table.

Bit No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bit	0	1	1	1	1	0	1	0	1	0	0	0	1	0	0	1

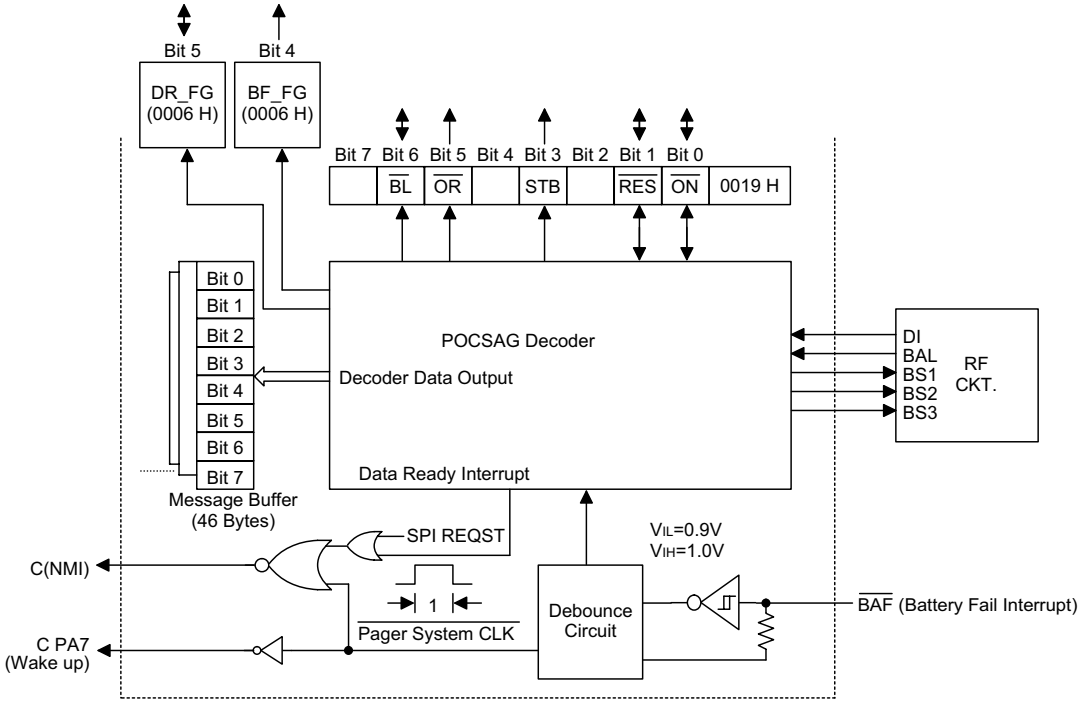
Bit No.	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Bit	1	1	0	0	0	0	0	1	1	0	0	1	0	1	1	1

• Error indication

After error correction, any code-word containing more than 2-bit random errors or 4-bit burst errors (option) in the address or message code-word may be indicated from the error flag position.

• Decoder and μC interface

The HT9580 has two μC interface available. One group is the pager control address (0019H), which controls the operation and configuration address of the decoder from 1AH~2EH. The other is the pager message buffer address (from 0040H to 006DH), which receives the message data of calls in the parallel mode. The data ready (DR_FG) and battery fail (BF_FG) interrupt flags are in the interrupt flag register (0006H).



The decoder control address (0019H) contains a battery low flag ($\overline{BL}$), an out of range flag ($\overline{OR}$), decoder standby flag (STB), a decoder software reset ($\overline{RES}$), and a decoder on/off control bit ($\overline{ON}$). The data ready and battery fail flag are in the interrupt flag register (0006H). It not only records the status information but controls the operation of the decoder.

If the flag status of the battery fail (BF_FG) changes from "0" to "1", the following conditions occur.

- The pager controller generates an interrupt if the value of the data ready interrupt is "0".
- The pager controller does not generate any interrupt if the value of the data ready interrupt is "1".

On the other hand, if the status of the battery fail flag (BF_FG) changes from "1" to "0", the internal node PA.7 of the pager controller will supply a wake-up function. After the decoder asserts the data ready request, the data ready interrupt is generated and the DR_FG bit (bit 5 of 0006H) is set high; then the data ready interrupt subroutine runs to process the call data on the message buffer and resets the DR_FG bit low.

The functional bits ($\overline{ON}$, $\overline{RES}$) and indication bits (STB, $\overline{OR}$, $\overline{BL}$, BF_FG and DR_FG) are all used to control the status of the decoder which is operated through the pager control address as described in the following table.

INT flag register (0006H)

Symbol	Bit	R/W	Description
BF_FG	4	R	Battery fail indication bit Once the decoder detects that the battery fail condition occurred, the BF_FG will go high.
DR_FG	5	R/W	Data ready interrupt indication bit When a valid call is detected, data transfers to the message buffer. The DR_FG bit goes high when the message is terminated within 46 bytes, one batch is at the end during the message receiving or the data buffer is full if the data length is more than 46 bytes. The μC software should read the data on the message buffer within one POCSAG message codeword (32-bit) time. The μC software has to clear the DR_FG bit low.

Decoder control register (0019H)

Symbol	Bit	R/W	Description
$\overline{ON}$	0	R/W	On/Off control bit This bit selects the $\overline{ON}$ or STANDBY state of the decoder 0: ON state 1: STANDBY RES If SPI circuit is enabled, it would be better if this bit is set high to reduce power consumption.
$\overline{RES}$	1	R/W	Resets the decoder core output The μC has to set the $\overline{RES}$ bit low and then high after the pager controller is turned on. The reset status must be released before writing data to the decoder configuration RAM.

Symbol	Bit	R/W	Description
STB	3	R	Standby indication bit When the value of the $\overline{ON}$ bit is 1, the system goes into the STANDBY state. The STANDBY state allows the μC to execute the configuration RAM setting.
$\overline{OR}$	5	R	Out-of-range indication bit Whenever the decoder detects an out-of-range condition, this bit is cleared to low after expiration of the programmed out-of-range hold time is selected by the configuration bits (SPF06 and SPF07). The out-of-range indication may be tested for an out-of-range condition whenever the interface enable of the decoder is active; otherwise $\overline{OR}$ is normally low. The out-of-range indication is set high by detection of valid data transmission. If the out-of-range indication bit changes the status from high to low or low to high, an interrupt will be generated and the out-of-range hold-off time-out counter starts to count. The bit is not valid when the SPI circuit is enabled.
$\overline{BL}$	6	R/W	Battery low indication bit The battery low indication is periodically tested for a battery low condition. If the decoder encounters a battery low condition, the battery low indication bit is cleared low. The μC can only set the $\overline{BL}$ bit high. Attempting to clear this bit has no effect. The bit is not valid when the SPI circuit is enabled.

- User address format

  A user address in the POCSAG code consists of 21 bits. Three of the 21 bits are coded in the frame number and are therefore not explicitly transmitted. In the decoder, the addresses A, B, C, D, E and F can use six different frames respectively. Every address has to be explicitly enabled by resetting the associated enable bit.

  Example:

  Address decimal value: RICA=10535

  Binary equivalent (14-bit): 10100100100111

  Binary equivalent (18+3-bit): 000000010100100100111

Register allocation

A00	A01	A02	A03	A04	A05	A06
0	0	0	0	0	0	0

A07	A08	A09	A10	A11	A12	A13
1	0	1	0	0	1	0

A14	A15	A16	A17	FA2	FA1	FA0
0	1	0	0	1	1	1

- Test mode

  The test mode of the decoder is selected by setting the $\overline{TS}$ pin low at any time. In the test mode, the RF control outputs BS1 and BS3 are constantly set high, but BS2 is set low. After the $\overline{TS}$ pin is set high the decoder exits the test mode.

- Message data transfer

  The decoder outputs a de-formatted address word and message words upon receipt of a valid call. The message data to be transferred is organized into 8-bit words and transferred to the message buffer address (0040H to 006DH). The data ready interrupt flag will be set high when the received data (including address codeword and message codeword) length is terminated within 46 bytes, one batch is over or if the 46 bytes data buffer is full if data length is more than 46 bytes. If the data in the message buffer is terminated, the "MSG_END" (0031H) bit will set high.

  The address word indicates call address, functional bit setting, and decoder flags. The message code-words are received and concatenated to a valid call address word. The message words are derived from un-corrected message code-words.

- Address word format

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Sync. State	Call Address			Dup. Call	Valid Address	Function Code	

Note: Bit0: Bit21 of the address code-word
Bit1: bit20 of the address code-word
Bit2: If this bit is "1", the address word is valid, otherwise the address word is not valid.
Bit3: 1 for a duplicate code-word
Bit7: 1 if an address code-word is received in the data fail mode

Bit6	Bit5	Bit4	Call Address
0	0	0	RIC A
0	0	1	RIC B
0	1	0	RIC C
0	1	1	RIC D
1	0	0	RIC E
1	0	1	RIC F
1	1	0	—
1	1	1	—

- Interrupt indication

  The HT9580 provides an internal data ready interrupt and a battery fail interrupt. The internal data ready interrupt and battery fail interrupt share the NMI location. Which interrupt occurred can be determined by checking the battery fail interrupt bit (BF_FG; bit 4 of 0006H) and the data ready interrupt bit (DR_FG; bit 5 of 0006H). Both interrupt bits are active high.

- Out-of-range indication

  The out-of-range condition occurs when the time interval defined by SPF06, SPF07 is unable to receive sync code words. If sync code words are detected, the timer counter defined by SPF06, SPF07 will reset. This signal will be seen as a loss of RF signal indication and the power on reset is in an out-of-range condition until the sync code word is detected.

- Duplicate call suppression

  The HT9580 provides a duplicate call suppression with time-out facility, to identify duplicate call reception. In the display pager mode, duplicate call indication is achieved only via the μC interface. A call is assumed to be a duplicate if its latest address and function bit setting is equal to the previous received call within the time interval defined by SPF06, SPF07.

- Receiver, Quick charge and PLL signal control

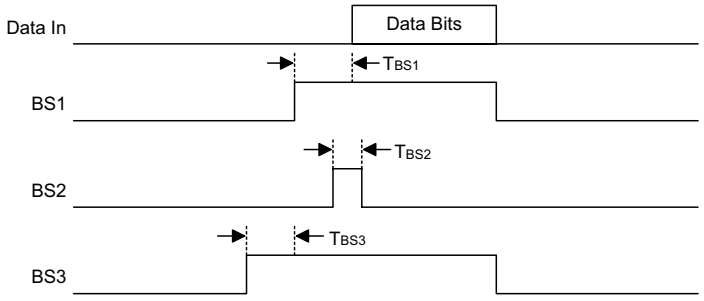
  Pager receiver, quick charge circuit, and RF PLL circuit can be controlled independently via enable outputs BS1, BS2, and BS3 respectively. Their operating period are optimized according to the synchronization mode of the decoder. Each enable signal has its own programmable establishment time.

Receiver Establishment Time T_{BS1}		Option	
512 bps	1200/2400 bps	SPF00	SPF01
7.81ms	53.33ms	0	0
15.63ms	6.67ms	0	1
31.25ms	13.33ms	1	0
62.50ms	26.67ms	1	1

Quick Charge Adjustment Time T_{BS2}		Option	
512 bps	1200/2400 bps	SPF02	SPF03
7.81ms	1.67ms	0	0
15.63ms	6.67ms	0	1
15.63ms	11.67ms	1	0
19.53ms	13.33ms	1	1

PLL Establishment Time T_{BS3}		Option	
512 bps	1200/2400 bps	SPF04	SPF05
0ms	0ms	0	0
31.25ms	26.67ms	0	1
46.87ms	40.00ms	1	0
62.50ms	53.33ms	1	1

R.F. timing chart



Decoder configuration RAM

The decoder contains a 21-byte RAM to store six user addresses, six frame numbers, and specially programmed function bits (SPF00~SPF19) for the decoder application configuration. The data memory is mapped to the address 001AH~002EH.

The configuration memory mapping table is shown below.

Address	Bit Definition							
	Bit7	**Bit6**	**Bit5**	**Bit4**	**Bit3**	**Bit2**	**Bit1**	**Bit0**
001AH	$\overline{\text{ENA}}$	A00	A01	A02	A03	A04	A05	A06
001BH	A07	A08	A09	A10	A11	A12	A13	A14
001CH	A15	A16	A17	FA2	FA1	FA0	X	X
001DH	$\overline{\text{ENB}}$	B00	B01	B02	B03	B04	B05	B06
001EH	B07	B08	B09	B10	B11	B12	B13	B14
001FH	B15	B16	B17	FB2	FB1	FB0	X	X
0020H	$\overline{\text{ENC}}$	C00	C01	C02	C03	C04	C05	C06
0021H	C07	C08	C09	C10	C11	C12	C13	C14
0022H	C15	C16	C17	FC2	FC1	FC0	X	X
0023H	$\overline{\text{END}}$	D00	D01	D02	D03	D04	D05	D06
0024H	D07	D08	D09	D10	D11	D12	D13	D14
0025H	D15	D16	D17	FD2	FD1	FD0	X	X
0026H	$\overline{\text{ENE}}$	E00	E01	E02	E03	E04	E05	E06
0027H	E07	E08	E09	E10	E11	E12	E13	E14
0028H	E15	E16	E17	FE2	FE1	FE0	X	X
0029H	$\overline{\text{ENF}}$	F00	F01	F02	F03	F04	F05	F06
002AH	F07	F08	F09	F10	F11	F12	F13	F14
002BH	F15	F16	F17	FF2	FF1	FF0	X	X
002CH	SPF00	SPF01	SPF02	SPF03	SPF04	SPF05	SPF06	SPF07
002DH	SPF08	SPF09	SPF10	SPF11	SPF12	SPF13	SPF14	SPF15
002EH	SPF16	SPF17	SPF18	SPF19	X	X	X	X

Description of the special programmed function bits (SPFn)

The following features can be selected by appropriate programming of the specially programmed function bits:

- SPF00~SPF01

  Receiver (BS1) establishment time (for the BS2~BS3 options, refer to SPF02~SPF05)

  00: 7.81ms/512 53.33ms/1200/2400

  01: 15.63ms/512 6.67ms/1200/2400

  10: 31.25ms/512 13.33ms/1200/2400

  11: 62.50ms/512 26.67ms/1200/2400

  Note: The recommendatory setting is 11.

- SPF02~SPF03

  RF dc level adjustment (BS2) enable time

  00: 7.81ms/512 1.67ms/1200/2400

  01: 11.71ms/512 6.67ms/1200/2400

  10: 15.63ms/512 11.67ms/1200/2400

  11: 19.53ms/512 13.33ms/1200/2400

- SPF04~SPF05

  PLL (BS3) establishment time

  00: 0ms/512 0ms/1200/2400

  01: 31.25ms/512 26.67ms/1200/2400

  10: 46.87ms/512 40.00ms/1200/2400

  11: 62.50ms/512 53.33ms/1200/2400

- SPF06~SPF07

  The duplicate call suppress time-out and out-of-range hold-off time-out

  00: 30s/512/1200 15s/2400

  01: 60s/512/1200 30s/2400

  10: 120s/512/1200 60s/2400

  11: 240s/512/1200 120s/2400

- SPF08~SPF09

  0x: Any two consecutive code-words or the code-word directly following the address code-word in error

  10: Any single code-word in error

  11: Any two consecutive code-words in error

- SPF10

  1: 4-bit burst error correction for address and message code-word

  0: 2-bit random error correction for address and message code-word

- SPF11

  1: Out-of-range Hold-off period according to SPF06 and SPF07

  0: Out-of-range Hold-off period is 0 regardless of SPF06 and SPF07

Baud rate selection bits(SPF12,SPF13,SPF14)

SPF12	SPF13	SPF14	Connected Crystal (Hz)	Baud Rate (bps)
0	0	0	32768	512
0	0	1	76.8k	512
0	1	0	76.8k	1200
0	1	1	76.8k	2400
1	0	0	153.6k	512
1	0	1	153.6k	1200
1	1	0	153.6k	2400

Note: The (SPF12, SPF13, SPF14) = (0, 1, 0) when power on reset

- SPF15

  Non-inverting or inverting data input selection

  1: Inverting input selected for DI from RF circuit, referring to DI

  0: Non-inverting input selected for DI from RF circuit reserved, should be 0

- SPF16~SPF19

  Message receiving mode selection depending on the function code

	0	1
SPF16	Function Code (0, 0) is a numeric message	Function Code (0, 0) is an alpha-numeric message
SPF17	Function Code (0, 1) is a numeric message	Function Code (0, 1) is an alpha-numeric message
SPF18	Function Code (1, 0) is a numeric message	Function Code (1, 0) is an alpha-numeric message
SPF19	Function Code (1, 1) is a numeric message	Function Code (1, 1) is an alpha-numeric message

CPU Core

The HT9580 is a high performance pager controller specifically designed for use in new generation radio pagers. It is based on the M6502 core. The 6502 Microprocessor offers complete hardware and software capability with existing 6500 series of products as well as significant enhancements.

Instruction register and decoder

Instructions fetched from memory are gated onto the internal bus. These instructions are latched into the instruction register then decoded, along with timing and interrupt signals, to generate control signals for the various registers.

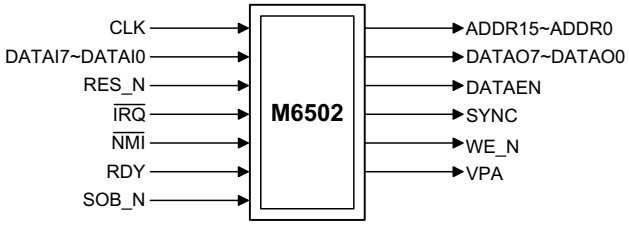
Timing control unit

The timing control unit keeps track of the instruction cycle being monitored. The unit is set to 0 each time an instruction fetch is executed and is advanced at the beginning of each input clock pulse for as many cycles as required to complete the instruction. Each data transfer between registers depends upon decoding the contents of both the Instruction Register and the Timing Control Unit.

All timing parameters in the M6502 are relative to the rising edge of the μC clock. All input signals are sampled at the rising edge of the μC clock, and all output signals transition after some delay relative to the rising edge of the μC clock. Input hold times are the amount of time after the rising edge of the clock that a signal must remain stable. Output hold times are the minimum delay that the signal will remain stable after the rising edge of the clock.

It should be noted that most of the outputs are decoded, and thus will glitch during the period from the hold time to the delay time.

The M6502 has 42 external signals (14 inputs and 28 outputs). These are illustrated in below:



M6502 I/O signals

The following table provides a description of M6502 input signals.

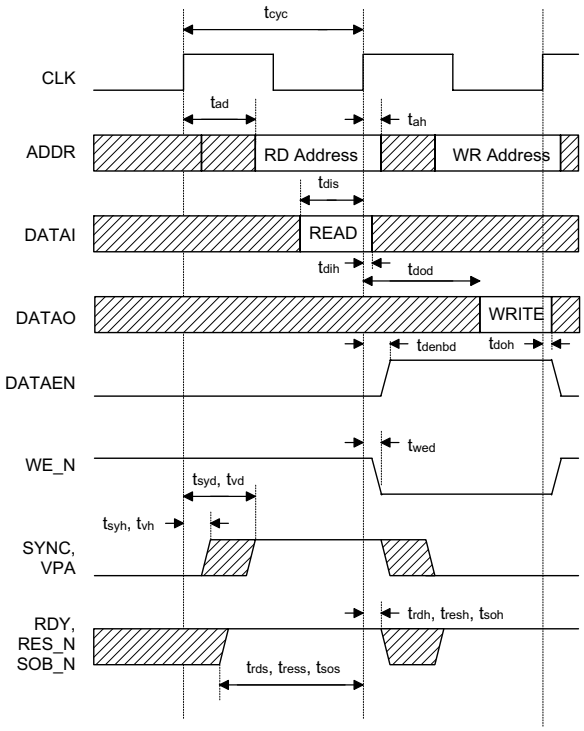
Signal	Type	Description
CLK	Input	CLOCK: External clock input.
DATAI7~ DATAI0	Input	Data bus in: This is the input half of the 8-bit data bus.
RES_N	Input	RESET: RES_N is used to reset the M6502. (RES_N must be held low for at least three clocks to insure proper initialization) RES_N is synchronized to the rising edge of the clock before it is used internally. Upon release of RES_N, the reset vector at FFFCH and FFFDH is fetched and loaded into the program counter (PC). Execution then begins at this new location. Note that the same logic is used here as with other interrupts ie. the state machine will go through the same steps of pushing PSR, PCH and PCL onto the stack (WE_N will remain high). So after RES_N is released, SREG will contain FDH (0 minus 3 pushes). All other registers will also be initialized, mostly to zero (a few of the temporary registers will be loaded with DATAI bus).
$\overline{\text{IRQ}}$	Input	MASKABLE INTERRUPT: IRQ is a maskable interrupt to the M6502 (active low). If IRQ=0 during an opcode fetch and the PSR I bit=0, this will initiate an interrupt sequence. The interrupt sequence begins by forcing a BRK opcode into the instruction register (I).
$\overline{\text{NMI}}$	Input	NON-MASKABLE INTERRUPT: NMI (active low) is a non-maskable interrupt to the M6502. A falling edge on NMI will initiate an interrupt sequence on the next opcode fetch.
RDY	Input	READY: RDY is used to hold the M6502 in the current state for multiple clock cycles. This is most often used to insert wait states when accessing slow peripherals. RDY=1 indicates that memory is ready and the processor can continue. When RDY=0, the processor stops execution at the current state and most of the register load enables are disabled in order to stop the flow of data. This signal can be used to reduce power consumption while waiting for an external stimulus to resume processing. When RDY is asserted, the internal signals of the M6502 will be static except for the clock. It should be noted that as RDY passes through the opcode decode block, it will typically have a significant amount of delay.
SOB_N	Input	SET OVERFLOW BIT: The V bit in the PSR is set when a falling edge is detected on this input.

M6502 input signals

The following table provides a description of M6502 output signals.

Signal	Type	Description
ADDR15~ ADDR0	Output	ADDRESS BUS: The 16-bit address bus is used to address data in memory.
DATAO7~ DATAO0	Output	DATA BUS OUT: This half of the data bus contains the output data.
DATAEN	Output	DATA OUT ENABLE: This signal is active (high) when data is to be driven out onto the bus for external purposes (indicated by WE_N low).
SYNC	Output	SYNCHRONIZE: SYNC=1 whenever an opcode is fetched from memory. It can be combined with RDY to create a single cycle execution for debugging purposes.
WE_N	Output	WRITE ENABLE: WE_N (active low) indicates whether the M6502 is reading from or writing to memory or an I/O device (1=read, 0=write). The DATAEN signal is active (high) whenever WE_N is also active (low). The rising edge of WE_N should be used by external devices to latch data on the DATAO bus. Note: Care must be taken to ensure there are no hold time issues between WE_N and the ADDR/DATA busses. All these signals will transition on the same edge of the CLK. WE_N is driven directly from a flip-flop, so it will typically have very little delay. The ADDR and DATA busses however pass through several layers of logic that make up the output multiplexors, and thus have significantly more delay than WE_N.
VPA	Output	VECTOR PULL ADDRESS: VPA (active high) signals that the M6502 is reading an interrupt vector from memory. This signal can be used in conjunction with a priority interrupt controller to provide rapid entry into numerous interrupt service routines. It should be noted that this signal is active for two clocks. The low byte of the vector address is provided during the first clock, and the high byte is provided on the second clock. The M6502 provides only a single maskable interrupt vector, which is also shared with the BRK opcode. When there are many sources of interrupts, software must poll each of these devices to determine which device needs servicing. VPA typically selects a bank of registers that provide a direct entry into an interrupt service routine based on the pending interrupts. This provides very rapid entry into the highest priority interrupt service routine.

M6502 output signals



M6502 read and write cycle

Read/write

This signal is normally in a high state indicating that the μC is reading data from the data bus memory. In the low state the data bus has valid data from the μC to be stored at the addressed memory location.

Parameter	Description
t_{cyc}	Clock cycle time (min)
t_{ad}	Address delay time
t_{ah}	Address hold time
t_{dis}	Read data in setup time
t_{dih}	Read data in hold time
t_{dod}	Write data out delay time
t_{doh}	Write data out hold time
t_{denbd}	DATAEN delay time
t_{wed}	WE_N delay time
t_{syd}	SYNC delay time

Parameter	Description
t_{syh}	SYNC hold time
t_{vd}	VPA delay time
t_{vh}	VPA hold time
t_{sos}	SOB_N setup time
t_{soh}	SOB_N hold time
t_{rds}	RDY setup time
t_{rdh}	RDY hold time
t_{ress}	RES_N setup time
t_{resh}	RES_N hold time

Timing parameter annotations

Arithmetic and logic unit

All arithmetic and logic operations take place within the ALU including incrementing and decrementing internal registers (except for the program counter). The ALU has no internal

memory and is used only to perform logical and transient numerical operations.

Accumulator

The Accumulator is a general purpose 8-bit register which stores the results of most arithmetic and logic operations. In addition, the accumulator usually contains one of the two data words used in these operations.

Index register

There are two 8-bit Index Register (X and Y) which may be used to count program steps or to provide an index value to be used in generating an effective address. When executing an instruction which specifies indexed addressing, the μC fetches the opcode and the base address, and modifies the address by adding the index register to it prior to performing the desired operation. Pre- or post-indexing of indirect addresses is possible.

Processor status register

The 8-bit processor status register contains seven status flags. Some of the flags are controlled by the program, others may be controlled both by the program and the μC. The HT9580 instruction set contains a number of conditional branch instructions which are designed to allow testing of these flags.

Program counter

The 16-bit program counter register provides the addresses which step the μC through sequential program instructions. Each time the HT9580 fetches an instruction from the program memory, the lower byte of the program counter (PCL) is placed on the low-order bits of the address bus and the higher byte of the program counter (PCH) is placed on the high-order 8 bits. The counter is incremented each time an instruction or data is fetched from the program memory.

Stack pointer

The stack pointer is an 8-bit register which is used to control the addressing of the variable-length stack. The stack pointer is automatically incremented and decremented under control of the microprocessor to perform stack manipulations under direction of either the program or interrupt (NMI and IRQ). The stack allows simple implementation of nested subroutines and multiple level interrupts. The stack pointer is initialized by the user's software.

Status register

N	V	E	B	D	I	Z	C

Note: C: Carry 1=true

Z: Zero 1=true

I: $\overline{\text{IRQ}}$ 1=disable

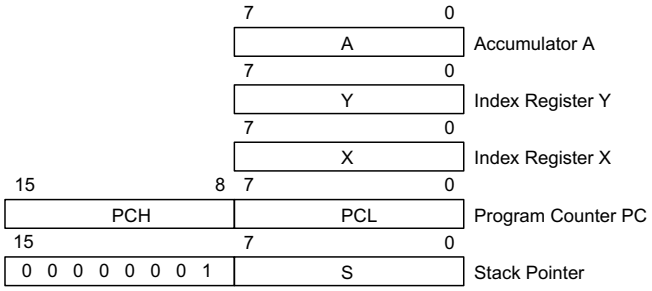
D: Decimal mode 1=true

B: BRK command 1=BRK, 0=$\overline{\text{IRQ}}$

E: Expansion bit (reserved)

V: Overflow 1=true

N: Negative 1=negative



The width of the corresponding registers

Interrupt System

The HT9580 is capable of directly addressing 64 Kbytes of memory. The address space has special significance within certain addressing modes, as follows:

Reset and interrupt vectors

The reset and interrupt vectors use the majority of the fixed addresses between FFFA and FFFF.

Stack

The M6502 stack may use memory from 0100 to 01FF. The effective address of stack and stack relative addressing modes will always be within this range.

Interrupt request – IRQ

This CMOS compatible signal requests that an interrupt sequence begin within the μC. The IRQ is sampled during PHI2 operation; if the interrupt flag in the processor status register is 0, the current instruction is completed and the interrupt sequence begins during PHI1. The program counter and processor status register are stored in the stack. The μC will then set the interrupt mask flag high so that no further interrupts may occur. At the end of this cycle, the PCL will be loaded from address FFFE, and PCH from location FFFF, transferring program control to the memory vector located at these addresses. The IRQ signal going low causes 3 bytes of information to be pushed onto the stack before jumping to the interrupt handler. The first byte is the high byte in the program counter. The second byte is the program counter low byte. The third byte is the status register value. These values are used to return the processor to its original state prior to the IRQ interrupt.

Non-maskable interrupt – NMI

A negative-going edge on this input requests that a non-maskable interrupt sequence be generated within the μC. The NMI is sampled during PHI2; the current instruction is completed and the interrupt sequence begins during PHI1. The Program Counter is loaded with the interrupt vector from locations FFFA (low byte) and FFFB (high byte), thereby transferring program control to the non-maskable interrupt routine. The NMI is generated from data ready interrupt or battery fail interrupt flag (0006H). However, it should be noted that this is an edge-sensitive input. As a result, another interrupt will occur if there is another negative-going transition and the program has not returned. Also, no interrupt will occur if NMI is low and a negative-going edge has not occurred since the last non-maskable interrupt. The NMI signal going low causes 3 bytes of information to be pushed onto the stack before jumping to the interrupt handler. The first byte is the high byte in the program counter. The second byte is the program counter low byte. The third byte is the status register value. These values are used to return to its original state prior to NMI interrupt.

Data address space

The μC internal address bus consists of A0~A15 forming a 16-bit address bus for memory and I/O exchanges on the data bus. The output of each address line is CMOS compatible. The Address output pins of HT9580 (A0~A15) derive from μC internal address pins A0~A15. The extended address pins (RA14~RA17) are the combination of bank point registers (0015H, 0016H) and internal address. The extended address pins are used to access internal/external SRAM or Mask ROM (Character ROM).

The data lines constitute 8-bit bidirectional data bus for use during exchanges between the μC and peripherals. The outputs are three-state buffers capable of driving CMOS load. The Program Address and Data Address space is continuous throughout the 64 Kbyte address space. Words, arrays, records, or any data structures may span the 64 Kbytes address space. The following addressing mode descriptions provide additional detail as to how effective addresses are calculated. Fifteen addressing modes are available for the HT9580 as illustrated on the next page.

Addressing modes

The M6502 supports fifteen (15) addressing modes, shown in table below. In interpreting this table you should note that:

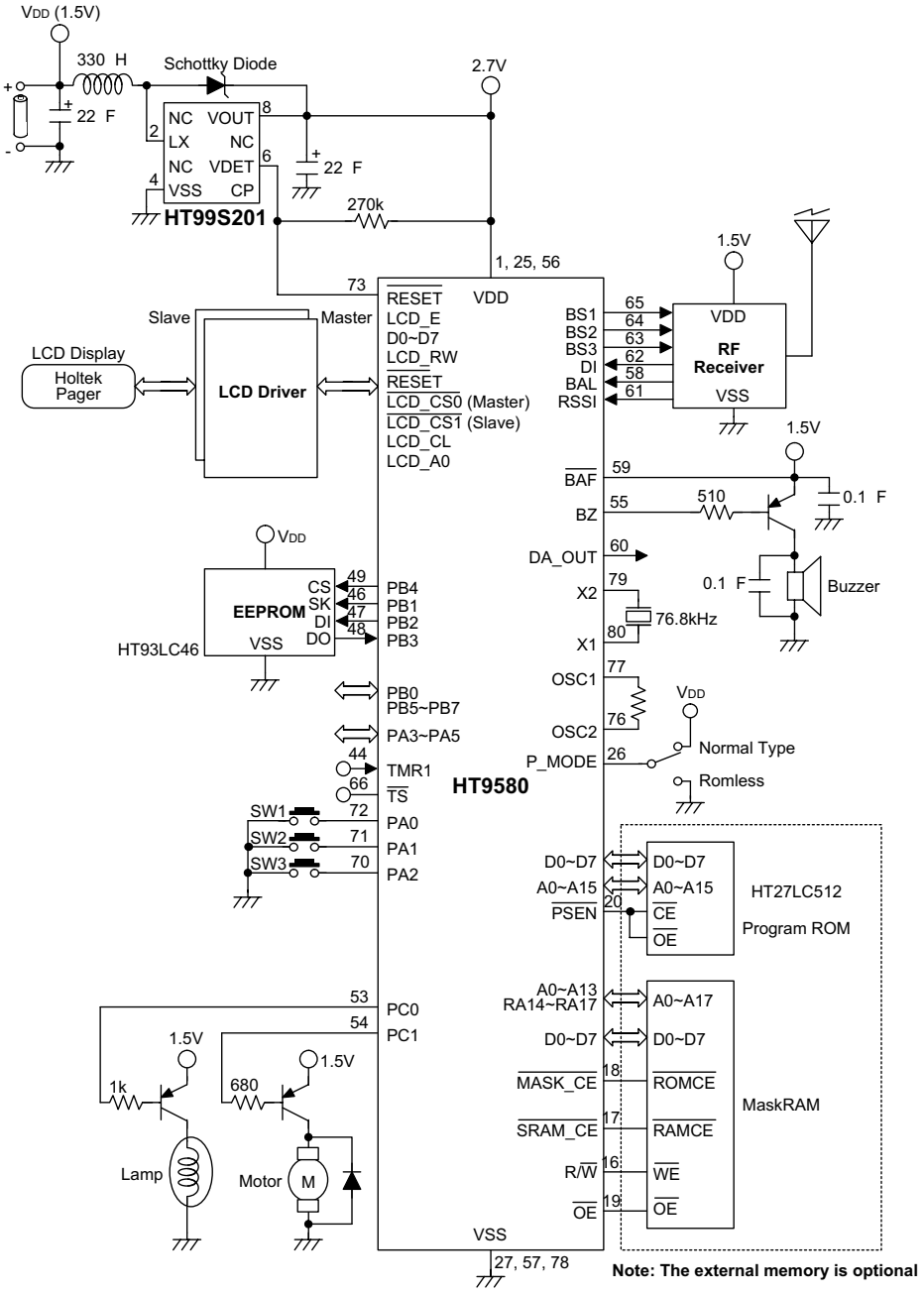
- The byte following a 2 byte opcode = IAL (typ.)
- The 2 bytes following a 3 byte opcode = BAL and BAH (typ.)
- Standard assembly notation is used
- A number in parenthesis indicates that the contents of the location pointed to by the number are to be used. For example (12H) indicates the contents of address 12H.
- A comma in the address is used to indicate the high and low byte of an address. For example (01H, AAH) indicates the contents of address 01AAH.

Mode	Description
IMP	IMPLIED: The data is implied in the opcode (example: CLC)
ACC	ACCUMULATOR: The accumulator is used as the source data. (data=AREG)
IMM	IMMEDIATE: The byte following the opcode is the data. (data=IAL)
ZPG	ZERO PAGE: The first 256 RAM locations (0000H~00FFH) are used for fast access and small code size. The upper 8-bit of the address are always zero. [data=(00, IAL)]
ZPX	ZERO PAGE INDIRECT X: The X register is added to the byte following the opcode to give a new zero page address. Note that the upper 8-bit of the address are always zero. [data=(00, IAL+X)]
ZPY	ZERO PAGE INDIRECT Y: The Y register is added to the byte following the opcode to give a new zero page address. Note that the upper 8-bit of the address are always zero. Only the LDX and the STX opcodes use this mode. [data=(00, IAL+Y)]
ABS	ABSOLUTE: The two bytes following the opcode give the absolute address of the data. [data=(BAH, BAL)]
ABX	ABSOLUTE X: The X register is added to the two bytes following the opcode to produce a new 16-bit address. [data=({BAH, BAL}+X)]
ABY	ABSOLUTE Y: The Y register is added to the two bytes following the opcode to produce a new 16-bit address. [data=({BAH, BAL}+Y)]
ABI	ABSOLUTE INDIRECT: The two bytes following the opcode are used as a pointer to memory. Only the JMP opcode uses this mode. [data=(BAH, BAL)]
AIX	INDEXED ABSOLUTE INDIRECT X: The two bytes following the opcode are added to the X register to yield a new 16-bit address. The contents of this address and the following one are used as an indirect address. Only the JMP opcode uses this mode. [data=({BAH, BAL+X+1},{BAH, BAL+X})]
IND	INDIRECT: The byte following the opcode is used as a pointer to the zero page. The contents of this address and the following one are used as the address to finally access the data. [data=({IAL+1}, {IAL})]

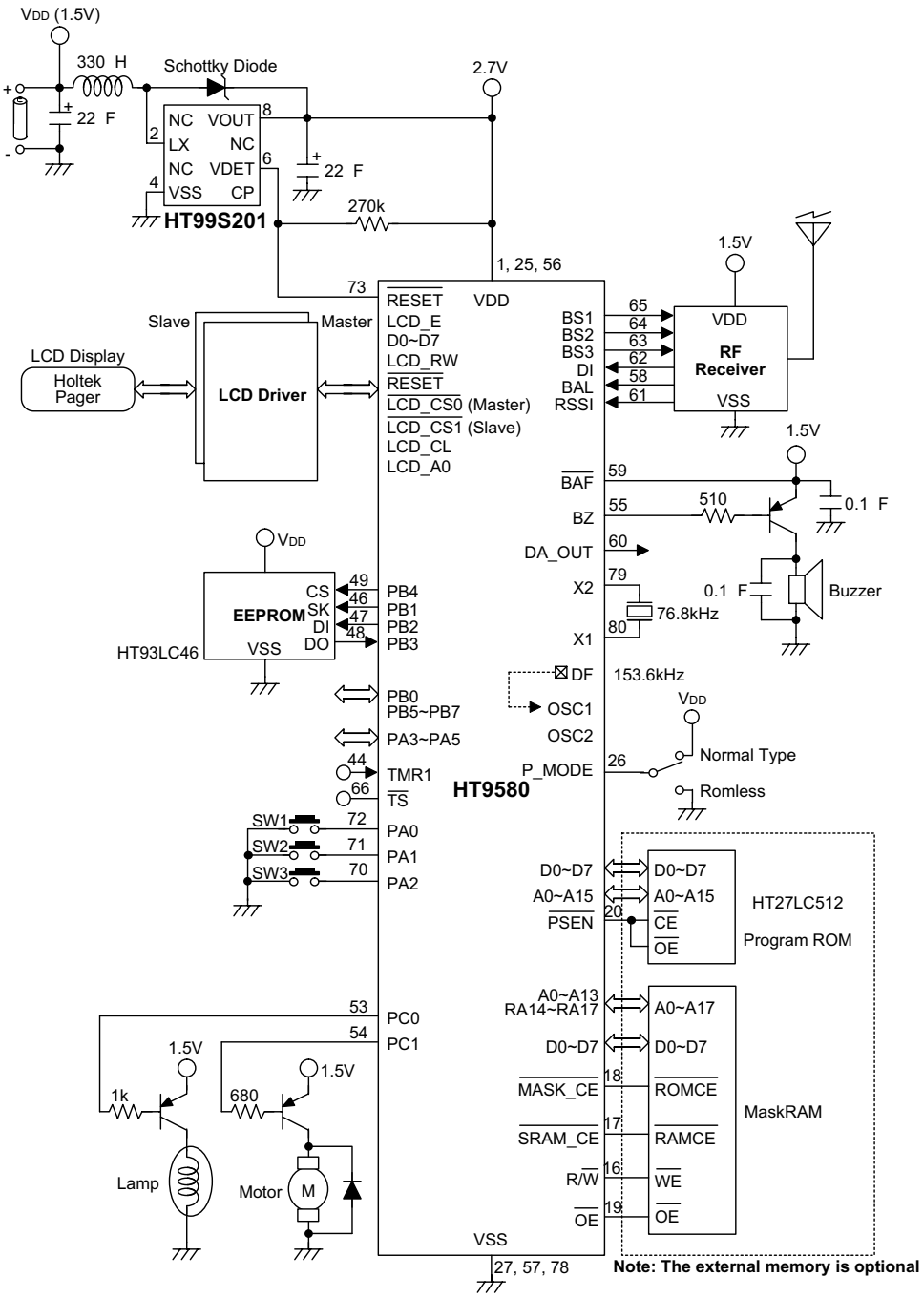
Mode	Description
INX	INDIRECT X: The byte following the opcode is added to the X register to produce a new zero page address. The contents of this address and the following one are used as the address to finally access the data. Note that when the X register is added to the byte following the opcode, the upper byte of the address is always zero. [data=({00, IAL+X+1}, {00, IAL+X})]
INY	INDIRECT Y: The byte following the opcode is a zero page address. The contents of this location and the next one produce a 16-bit address which is then added to the Y register to finally obtain the data. [data=(({00, IAL+1}, {00, IAL})+Y)]
REL	RELATIVE: The byte following the opcode is added in 2s complement fashion to the PC. The byte is sign extended. Used by branching instructions.

Application Circuits

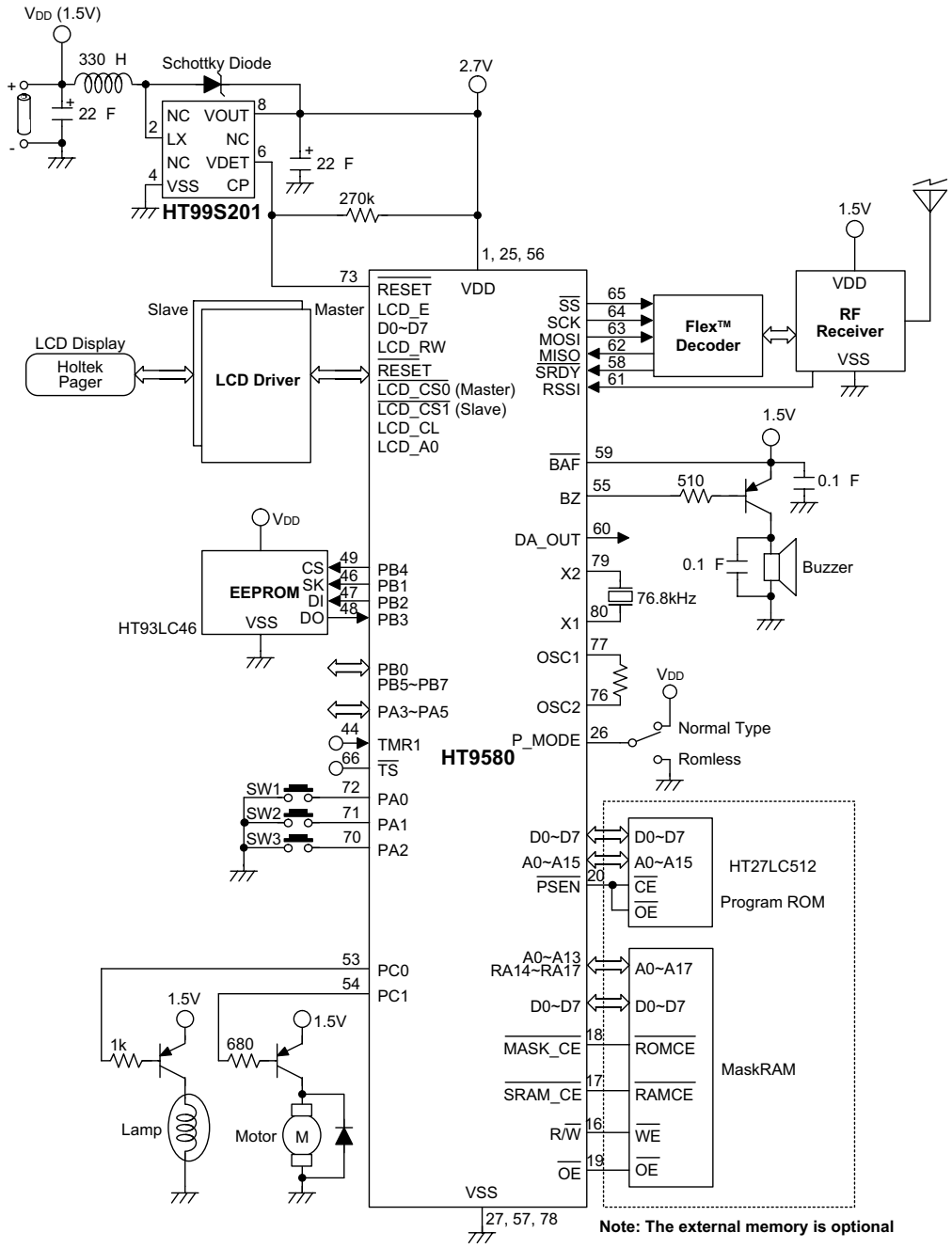
OSC1, OSC2 require an external resistor



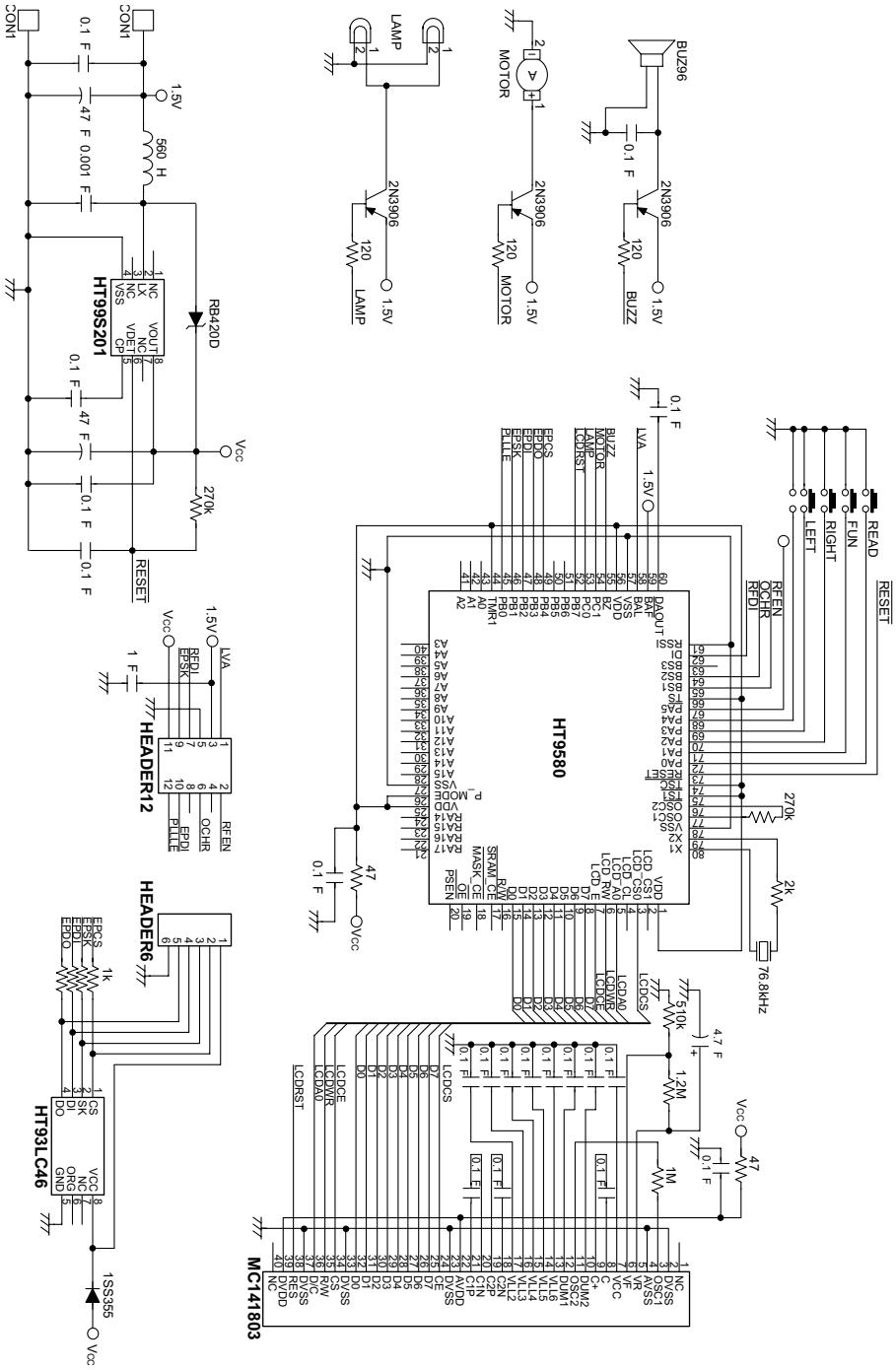
OSC1, OSC2 do not require a resistor. The OSC1 clock comes from an internal pad "DF" only



The 5 wire serial interface application circuits



Holtek standard pager circuits



Detailed Instruction Operation

The table below provides a brief description of each opcode.

The first column lists the name or the assembler mnemonic for the instruction.

The second column lists the opcode in hexadecimal.

The third column lists the address mode for the instruction.

The flags column indicates which of the 8-bit of flags are updated by the instruction.

Legend:

–	→ No change
+	→ Updated
6	→ From memory bit 6
7	→ From memory bit 7

The number of bytes column gives the number of bytes for the opcode.

The number of cycles column gives the number of clock cycles for the opcode. (A+b indicates one additional cycle when a branch is taken within the same page, or 2 cycles if the branch is to a different page.)

The last column are the description or brief descriptions of the opcode.

The operator notation is as follows:

=>	assignment
+	2′s complement add
–	2′s complement subtract
|	Bitwise OR
&	Bitwise AND
^	Bitwise exclusive OR
!	Bitwise invert (1′s complement)
<<	Left rotate
>>	Right rotate
<	Left shift
>	Right shift
A	Accumulator
C	Carry flag
X	X index register
Y	Y index register
S	Stack pointer
M	Memory

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
ADC	69	IMM	+	+	-	-	-	-	+	+	2	2	A+M+C=>A, C Add with carry
ADC	65	ZPG	+	+	-	-	-	-	+	+	2	3	A+M+C=>A, C Add with carry
ADC	75	ZPX	+	+	-	-	-	-	+	+	2	4	A+M+C=>A, C Add with carry
ADC	6D	ABS	+	+	-	-	-	-	+	+	3	4	A+M+C=>A, C Add with carry
ADC	7D	ABX	+	+	-	-	-	-	+	+	3	4	A+M+C=>A, C Add with carry
ADC	79	ABY	+	+	-	-	-	-	+	+	3	4	A+M+C=>A, C Add with carry
ADC	72	IND	+	+	-	-	-	-	+	+	2	5	A+M+C=>A, C Add with carry
ADC	61	IDX	+	+	-	-	-	-	+	+	2	6	A+M+C=>A, C Add with carry
ADC	71	IDY	+	+	-	-	-	-	+	+	2	5	A+M+C=>A, C Add with carry
AND	29	IMM	+	-	-	-	-	-	+	-	2	2	A&M=>A AND A with M
AND	25	ZPG	+	-	-	-	-	-	+	-	2	3	A&M=>A AND A with M
AND	35	ZPX	+	-	-	-	-	-	+	-	2	4	A&M=>A AND A with M
AND	2D	ABS	+	-	-	-	-	-	+	-	3	4	A&M=>A AND A with M
AND	3D	ABX	+	-	-	-	-	-	+	-	3	4	A&M=>A AND A with M
AND	39	ABY	+	-	-	-	-	-	+	-	3	4	A&M=>A AND A with M
AND	32	IND	+	-	-	-	-	-	+	-	2	5	A&M=>A AND A with M
AND	21	IDX	+	-	-	-	-	-	+	-	2	6	A&M=>A AND A with M
AND	31	IDY	+	-	-	-	-	-	+	-	2	5	A&M=>A AND A with M
ASL	0A	ACC	+	-	-	-	-	-	+	+	1	2	A<1=>A shift left 1, C<-7, 0<-zero
ASL	06	ZPG	+	-	-	-	-	-	+	+	2	5	M<1=>M shift left 1, C<-7, 0<-zero
ASL	16	ZPX	+	-	-	-	-	-	+	+	2	6	M<1=>M shift left 1, C<-7, 0<-zero
ASL	0E	ABS	+	-	-	-	-	-	+	+	3	6	M<1=>M shift left 1, C<-7, 0<-zero
ASL	1E	ABX	+	-	-	-	-	-	+	+	3	7	M<1=>M shift left 1, C<-7, 0<-zero
BBR0	0F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (0)=0, PC<=PC+Off (Off sign ext)
BBR1	1F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (1)=0, PC<=PC+Off (Off sign ext)
BBR2	2F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (2)=0, PC<=PC+Off (Off sign ext)
BBR3	3F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (3)=0, PC<=PC+Off (Off sign ext)
BBR4	4F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (4)=0, PC<=PC+Off (Off sign ext)
BBR5	5F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (5)=0, PC<=PC+Off (Off sign ext)
BBR6	6F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (6)=0, PC<=PC+Off (Off sign ext)
BBR7	7F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (7)=0, PC<=PC+Off (Off sign ext)
BBS0	8F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (0)=1, PC<=PC+Off (Off sign ext)
BBS1	9F	REL	-	-	-	-	-	-	-	-	3	5+b	If M (1)=1, PC<=PC+Off (Off sign ext)
BBS2	AF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (2)=1, PC<=PC+Off (Off sign ext)
BBS3	BF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (3)=1, PC<=PC+Off (Off sign ext)
BBS4	CF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (4)=1, PC<=PC+Off (Off sign ext)

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
BBS5	DF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (5)=1, PC<=PC+Off (Off sign ext)
BBS6	EF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (6)=1, PC<=PC+Off (Off sign ext)
BBS7	FF	REL	-	-	-	-	-	-	-	-	3	5+b	If M (7)=1, PC<=PC+Off (Off sign ext)
BCC	90	REL	-	-	-	-	-	-	-	-	2	2+b	If C=0, PC<=PC+M (Msign extended)
BCS	B0	REL	-	-	-	-	-	-	-	-	2	2+b	If C=1, PC<=PC+M (Msign extended)
BEQ	F0	REL	-	-	-	-	-	-	-	-	2	2+b	If Z=1, PC<=PC+M (Msign extended)
BIT	89	IMM	7	6	-	-	-	-	+	-	2	2	A&M=>Z, M7=>N, M6=>V
BIT	24	ZPG	7	6	-	-	-	-	+	-	2	3	A&M=>Z, M7=>N, M6=>V
BIT	34	ZPX	7	6	-	-	-	-	+	-	2	4	A&M=>Z, M7=>N, M6=>V
BIT	2C	ABS	7	6	-	-	-	-	+	-	3	4	A&M=>Z, M7=>N, M6=>V
BIT	3C	ABX	7	6	-	-	-	-	+	-	3	4	A&M=>Z, M7=>N, M6=>V
BMI	30	REL	-	-	-	-	-	-	-	-	2	2+b	If N=1, PC<=PC+M (Msign extended)
BNE	D0	REL	-	-	-	-	-	-	-	-	2	2+b	If Z=0, PC<=PC+M (Msign extended)
BPL	10	REL	-	-	-	-	-	-	-	-	2	2+b	If N=0, PC<=PC+M (Msign extended)
BRA	80	REL	-	-	-	-	-	-	-	-	2	2+b	PC<=PC+M (Msign extended)
BRK	00	IMP	-	-	-	-	-	1	-	-	1	7	Set B, push PC & PSR, PC<=(FFFE), Set I
BVC	50	REL	-	-	-	-	-	-	-	-	2	2+b	If V=0, PC<=PC+M (Msign extended)
BVS	70	REL	-	-	-	-	-	-	-	-	2	2+b	If V=1, PC<=PC+M (Msign extended)
CLC	18	IMP	-	-	-	-	-	-	-	0	1	2	C<=0
CLD	D8	IMP	-	-	-	-	0	-	-	-	1	2	D<=0
CLI	58	IMP	-	-	-	-	-	0	-	-	1	2	I<=0
CLV	B8	IMP	-	0	-	-	-	-	-	-	1	2	V<=0
CMP	C9	IMM	+	-	-	-	-	-	+	+	2	2	A-M=>N, Z, C
CMP	C5	ZPG	+	-	-	-	-	-	+	+	2	3	A-M=>N, Z, C
CMP	D5	ZPX	+	-	-	-	-	-	+	+	2	4	A-M=>N, Z, C
CMP	CD	ABS	+	-	-	-	-	-	+	+	3	4	A-M=>N, Z, C
CMP	DD	ABX	+	-	-	-	-	-	+	+	3	4	A-M=>N, Z, C
CMP	D9	ABY	+	-	-	-	-	-	+	+	3	4	A-M=>N, Z, C
CMP	D2	IND	+	-	-	-	-	-	+	+	2	5	A-M=>N, Z, C
CMP	C1	INX	+	-	-	-	-	-	+	+	2	6	A-M=>N, Z, C
CMP	D1	INY	+	-	-	-	-	-	+	+	2	5	A-M=>N, Z, C
CPX	E0	IMM	+	-	-	-	-	-	+	+	2	2	X-M=>N, Z, C
CPX	E4	ZPG	+	-	-	-	-	-	+	+	2	3	X-M=>N, Z, C
CPX	EC	ABS	+	-	-	-	-	-	+	+	3	4	X-M=>N, Z, C
CPY	C0	IMM	+	-	-	-	-	-	+	+	2	2	Y-M=>N, Z, C
CPY	C4	ZPG	+	-	-	-	-	-	+	+	2	3	Y-M=>N, Z, C

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
CPY	CC	ABS	+	-	-	-	-	-	+	+	3	4	Y-M=>N, Z, C
DEC	C6	ZPG	+	-	-	-	-	-	+	-	2	5	M<=M -1
DEC	D6	ZPX	+	-	-	-	-	-	+	-	2	6	M<=M -1
DEC	CE	ABS	+	-	-	-	-	-	+	-	3	6	M<=M -1
DEC	DE	ABX	+	-	-	-	-	-	+	-	3	7	M<=M -1
DEC	3A	ACC	+	-	-	-	-	-	+	-	1	2	A<=A -1
DEX	CA	IMP	+	-	-	-	-	-	+	-	1	2	X<=X -1
DEY	88	IMP	+	-	-	-	-	-	+	-	1	2	Y<=Y -1
EOR	49	IMM	+	-	-	-	-	-	+	-	2	2	A<=A^M
EOR	45	ZPG	+	-	-	-	-	-	+	-	2	3	A<=A^M
EOR	55	ZPX	+	-	-	-	-	-	+	-	2	4	A<=A^M
EOR	4D	ABS	+	-	-	-	-	-	+	-	3	4	A<=A^M
EOR	5D	ABX	+	-	-	-	-	-	+	-	3	4	A<=A^M
EOR	59	ABY	+	-	-	-	-	-	+	-	3	4	A<=A^M
EOR	52	IND	+	-	-	-	-	-	+	-	2	5	A<=A^M
EOR	41	INX	+	-	-	-	-	-	+	-	2	6	A<=A^M
EOR	51	INY	+	-	-	-	-	-	+	-	2	5	A<=A^M
INC	E6	ZPG	+	-	-	-	-	-	+	-	2	5	M<=M+1
INC	F6	ZPX	+	-	-	-	-	-	+	-	2	6	M<=M+1
INC	EE	ABS	+	-	-	-	-	-	+	-	3	6	M<=M+1
INC	FE	ABX	+	-	-	-	-	-	+	-	3	7	M<=M+1
INC	1A	ACC	+	-	-	-	-	-	+	-	1	2	A<=A+1
INX	E8	IMP	+	-	-	-	-	-	+	-	1	2	X<=X+1
INY	C8	IMP	+	-	-	-	-	-	+	-	1	2	Y<=Y+1
JMP	4C	ABS	-	-	-	-	-	-	-	-	3	3	PC<M
JMP	6C	ABI	-	-	-	-	-	-	-	-	3	5	PC<=(M)
JMP	7C	AIX	-	-	-	-	-	-	-	-	3	5	PC<=(M)
JSR	20	ABS	-	-	-	-	-	-	-	-	3	6	Push PC, PC<=M
LDA	A9	IMM	+	-	-	-	-	-	+	-	2	2	A<=M
LDA	A5	ZPG	+	-	-	-	-	-	+	-	2	3	A<=M
LDA	B5	ZPX	+	-	-	-	-	-	+	-	2	4	A<=M
LDA	AD	ABS	+	-	-	-	-	-	+	-	3	4	A<=M
LDA	BD	ABX	+	-	-	-	-	-	+	-	3	4	A<=M
LDA	B9	ABY	+	-	-	-	-	-	+	-	3	4	A<=M
LDA	B2	IND	+	-	-	-	-	-	+	-	2	5	A<=M
LDA	A1	INX	+	-	-	-	-	-	+	-	2	6	A<=M

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
LDA	B1	INY	+	-	-	-	-	-	+	-	2	5	A<=M
LDX	A2	IMM	+	-	-	-	-	-	+	-	2	2	X<=M
LDX	A6	ZPG	+	-	-	-	-	-	+	-	2	3	X<=M
LDX	B6	ZPY	+	-	-	-	-	-	+	-	2	4	X<=M
LDX	AE	ABS	+	-	-	-	-	-	+	-	3	4	X<=M
LDX	BE	ABY	+	-	-	-	-	-	+	-	3	4	X<=M
LDY	A0	IMM	+	-	-	-	-	-	+	-	2	2	Y<=M
LDY	A4	ZPG	+	-	-	-	-	-	+	-	2	3	Y<=M
LDY	B4	ZPX	+	-	-	-	-	-	+	-	2	4	Y<=M
LDY	AC	ABS	+	-	-	-	-	-	+	-	3	4	Y<=M
LDY	BC	ABX	+	-	-	-	-	-	+	-	3	4	Y<=M
LSR	4A	ACC	0	-	-	-	-	-	+	+	1	2	M<=M>1 shift right 1, zero ->7, 0->C
LSR	46	ZPG	0	-	-	-	-	-	+	+	2	5	M<=M>1 shift right 1, zero ->7, 0->C
LSR	56	ZPX	0	-	-	-	-	-	+	+	2	6	M<=M>1 shift right 1, zero ->7, 0->C
LSR	4E	ABS	0	-	-	-	-	-	+	+	3	6	M<=M>1 shift right 1, zero ->7, 0->C
LSR	5E	ABX	0	-	-	-	-	-	+	+	3	7	M<=M>1 shift right 1, zero ->7, 0->C
NOP	EA	IMP	-	-	-	-	-	-	-	-	1	2	No operation
ORA	09	IMM	+	-	-	-	-	-	+	-	2	2	A<=A|M
ORA	05	ZPG	+	-	-	-	-	-	+	-	2	3	A<=A|M
ORA	15	ZPX	+	-	-	-	-	-	+	-	2	4	A<=A|M
ORA	0D	ABS	+	-	-	-	-	-	+	-	3	4	A<=A|M
ORA	1D	ABX	+	-	-	-	-	-	+	-	3	4	A<=A|M
ORA	19	ABY	+	-	-	-	-	-	+	-	3	4	A<=A|M
ORA	12	IND	+	-	-	-	-	-	+	-	2	5	A<=A|M
ORA	01	INX	+	-	-	-	-	-	+	-	2	6	A<=A|M
ORA	11	INY	+	-	-	-	-	-	+	-	2	5	A<=A|M
PHA	48	IMP	-	-	-	-	-	-	-	-	1	3	Push A on stack
PHP	08	IMP	-	-	-	-	-	-	-	-	1	3	Push status on stack
PHX	DA	IMP	-	-	-	-	-	-	-	-	1	3	Push X on stack
PHY	5A	IMP	-	-	-	-	-	-	-	-	1	3	Push Y on stack
PLA	68	IMP	+	-	-	-	-	-	+	-	1	3	Pull A from stack
PLP	28	IMP	From Stack								1	3	Pull status from stack
PLX	FA	IMP	+	-	-	-	-	-	+	-	1	3	Pull X from stack
PLY	7A	IMP	+	-	-	-	-	-	+	-	1	3	Pull Y from stack
RMB0	07	ZPG	-	-	-	-	-	-	-	-	2	4	M(0) <=0 (RMW)
RMB1	17	ZPG	-	-	-	-	-	-	-	-	2	4	M(1) <=0 (RMW)

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
RMB2	27	ZPG	-	-	-	-	-	-	-	-	2	4	M(2) <=0 (RMW)
RMB3	37	ZPG	-	-	-	-	-	-	-	-	2	4	M(3) <=0 (RMW)
RMB4	47	ZPG	-	-	-	-	-	-	-	-	2	4	M(4) <=0 (RMW)
RMB5	57	ZPG	-	-	-	-	-	-	-	-	2	4	M(5) <=0 (RMW)
RMB6	67	ZPG	-	-	-	-	-	-	-	-	2	4	M(6) <=0 (RMW)
RMB7	77	ZPG	-	-	-	-	-	-	-	-	2	4	M(7) <=0 (RMW)
ROL	2A	ACC	+	-	-	-	-	-	+	+	1	2	M<=M<<1, rotate left 1, 0←c, c←7
ROL	26	ZPG	+	-	-	-	-	-	+	+	2	5	M<=M<<1, rotate left 1, 0←c, c←7
ROL	36	ZPX	+	-	-	-	-	-	+	+	2	6	M<=M<<1, rotate left 1, 0←c, c←7
ROL	2E	ABS	+	-	-	-	-	-	+	+	3	6	M<=M<<1, rotate left 1, 0←c, c←7
ROL	3E	ABX	+	-	-	-	-	-	+	+	3	7	M<=M<<1, rotate left 1, 0←c, c←7
ROR	6A	ACC	+	-	-	-	-	-	+	+	1	2	M<=M>>1, rotate right 1, 7←c, c←0
ROR	66	ZPG	+	-	-	-	-	-	+	+	2	5	M<=M>>1, rotate right 1, 7←c, c←0
ROR	76	ZPX	+	-	-	-	-	-	+	+	2	6	M<=M>>1, rotate right 1, 7←c, c←0
ROR	6E	ABS	+	-	-	-	-	-	+	+	3	6	M<=M>>1, rotate right 1, 7←c, c←0
ROR	7E	ABX	+	-	-	-	-	-	+	+	3	7	M<=M>>1, rotate right 1, 7←c, c←0
RTI	40	IMP	From Stack								1	5	PC<=from stack, B=0
RTS	60	IMP	-	-	-	-	-	-	-	-	1	5	PC<=from stack
SBC	E9	IMM	+	+	-	-	-	-	+	+	2	2	A<=A-M-C is a borrow)
SBC	E5	ZPG	+	+	-	-	-	-	+	+	2	3	A<=A-M-C is a borrow)
SBC	F5	ZPX	+	+	-	-	-	-	+	+	2	4	A<=A-M-C is a borrow)
SBC	ED	ABS	+	+	-	-	-	-	+	+	3	4	A<=A-M-C is a borrow)
SBC	FD	ABX	+	+	-	-	-	-	+	+	3	4	A<=A-M-C is a borrow)
SBC	F9	ABY	+	+	-	-	-	-	+	+	3	4	A<=A-M-C is a borrow)
SBC	F2	IND	+	+	-	-	-	-	+	+	3	5	A<=A-M-C is a borrow)
SBC	E1	INX	+	+	-	-	-	-	+	+	3	6	A<=A-M-C is a borrow)
SBC	F1	INY	+	+	-	-	-	-	+	+	3	5	A<=A-M-C is a borrow)
SEC	38	IMP	-	-	-	-	-	-	-	1	1	2	C<=1
SED	F8	IMP	-	-	-	-	1	-	-	-	1	2	D<=1
SEI	78	IMP	-	-	-	-	-	1	-	-	1	2	I<=1
SMB0	87	ZPG	-	-	-	-	-	-	-	-	2	4	M(0) <=1 (RMW)
SMB1	97	ZPG	-	-	-	-	-	-	-	-	2	4	M(1) <=1 (RMW)
SMB2	A7	ZPG	-	-	-	-	-	-	-	-	2	4	M(2) <=1 (RMW)
SMB3	B7	ZPG	-	-	-	-	-	-	-	-	2	4	M(3) <=1 (RMW)
SMB4	C7	ZPG	-	-	-	-	-	-	-	-	2	4	M(4) <=1 (RMW)
SMB5	D7	ZPG	-	-	-	-	-	-	-	-	2	4	M(5) <=1 (RMW)

Name	Opcode	Addr Mode	Flags								No. Bytes	No. Cyc.	Description
			N	V	E	B	D	I	Z	C			
SMB6	E7	ZPG	-	-	-	-	-	-	-	-	2	4	M(6) <=1 (RMW)
SMB7	F7	ZPG	-	-	-	-	-	-	-	-	2	4	M(7) <=1 (RMW)
STA	85	ZPG	-	-	-	-	-	-	-	-	2	3	M<=A
STA	95	ZPX	-	-	-	-	-	-	-	-	2	4	M<=A
STA	8D	ABS	-	-	-	-	-	-	-	-	3	4	M<=A
STA	9D	ABX	-	-	-	-	-	-	-	-	3	4	M<=A
STA	99	ABY	-	-	-	-	-	-	-	-	3	4	M<=A
STA	81	INX	-	-	-	-	-	-	-	-	2	6	M<=A
STA	91	INY	-	-	-	-	-	-	-	-	2	5	M<=A
STX	86	ZPG	-	-	-	-	-	-	-	-	2	3	M<=X
STX	96	ZPY	-	-	-	-	-	-	-	-	2	4	M<=X
STX	8E	ABS	-	-	-	-	-	-	-	-	3	4	M<=X
STY	84	ZPG	-	-	-	-	-	-	-	-	2	3	M<=Y
STY	94	ZPX	-	-	-	-	-	-	-	-	2	4	M<=Y
STY	8C	ABS	-	-	-	-	-	-	-	-	3	4	M<=Y
STZ	64	ZPG	-	-	-	-	-	-	-	-	2	3	M<=0
STZ	74	ZPX	-	-	-	-	-	-	-	-	2	4	M<=0
STZ	9C	ABS	-	-	-	-	-	-	-	-	3	4	M<=0
STZ	9E	ABX	-	-	-	-	-	-	-	-	3	5	M<=0
TAX	AA	IMP	+	-	-	-	-	-	+	-	1	2	X<=A
TAY	A8	IMP	+	-	-	-	-	-	+	-	1	2	Y<=A
TRB	14	ZPG	-	-	-	-	-	-	+	-	2	5	M<=!A&M, Z=A&M
TRB	1C	ABS	-	-	-	-	-	-	+	-	3	6	M<=!A&M, Z=A&M
TSB	04	ZPG	-	-	-	-	-	-	+	-	2	6	M<=A|M, Z=A&M
TSB	0C	ABS	-	-	-	-	-	-	+	-	3	7	M<=A|M, Z=A&M
TSX	BA	IMP	+	-	-	-	-	-	+	-	1	2	X<=S
TXA	8A	IMP	+	-	-	-	-	-	+	-	1	2	A<=X
TXS	9A	IMP	-	-	-	-	-	-	-	-	1	2	S<=X
TYA	98	IMP	+	-	-	-	-	-	+	-	1	2	A<=Y

Opcode Matrix

The table below shows the matrix of M6502 opcodes:

MSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	BRK imp	ORA inx			TSB zpg	ORA zpg	ASL zpg	RB0 zpg	PHP imp	ORA imm	ASL acc		TSB abs	ORA abs	ASL abs	BR0 zpg
1	BPL rel	ORA iny	ORA ind		TRB zpg	ORA zpx	ASL zpx	RB1 zpg	CLC imp	ORA aby	INC acc		TRB abs	ORA abx	ASL abx	BR1 zpg
2	JSR abs	AND inx			BIT zpg	AND zpg	ROL zpg	RB2 zpg	PLP imp	AND imm	ROL acc		BIT abs	AND abs	ROL abs	BR2 zpg
3	BMI rel	AND iny	AND ind		BIT zpx	AND zpx	ROL zpx	RB3 zpg	SEC imp	AND aby	DEC acc		BIT abx	AND abx	ROL abx	BR3 zpg
4	RTI imp	EOR inx				EOR zpg	LSR zpg	RB4 zpg	PHA imp	EOR imm	LSR acc		JMP abs	EOR abs	LSR abs	BR4 zpg
5	BVC rel	EOR iny	EOR ind			EOR zpx	LSR zpx	RB5 zpg	CLI imp	EOR aby	PHY imp			EOR abx	LSR abx	BR5 zpg
6	RTS imp	ADC inx			STZ zpg	ADC zpg	ROR zpg	RB6 zpg	PLA imp	ADC imm	ROR acc		JMP abi	ADC abs	ROR abs	BR6 zpg
7	BVS rel	ADC iny	ADC ind		STZ zpx	ADC zpx	ROR zpx	RB7 zpg	SEI imp	ADC aby	PLY imp		JMP aix	ADC abx	ROR abx	BR7 zpg
8	BRA rel	STA inx			STY zpg	STA zpg	STX zpg	SB0 zpg	DEY imp	BIT imm	TXA imp		STY abs	STA abs	STX abs	BS0 zpg
9	BCC rel	STA iny	STA ind		STY zpx	STA zpx	STX zpy	SB1 zpg	TYA imp	STA aby	TXS imp		STZ abs	STA abx	STZ abx	BS1 zpg
A	LDY imm	LDA inx	LDX imm		LDY zpg	LDA zpg	LDX zpg	SB2 zpg	TAY imp	LDA imm	TAX imp		LDY abs	LDA abs	LDX abs	BS2 zpg
B	BCS rel	LDA iny	LDA ind		LDY zpx	LDA zpx	LDX zpy	SB3 zpg	CLV imp	LDA aby	TSX imp		LDY abx	LDA abx	LDX aby	BS3 zpg
C	CPY imm	CMP inx			CPY zpg	CMP zpg	DEC zpg	SB4 zpg	INY imp	CMP imm	DEX imp		CPY abs	CMP abs	DEC abs	BS4 zpg
D	BNE rel	CMP iny	CMP ind			CMP zpx	DEC zpx	SB5 zpg	CLD imp	CMP aby	PHX imp			CMP abx	DEC abx	BS5 zpg
E	CPX imm	SBC inx			CPX zpg	SBC zpg	INC zpg	SB6 zpg	INX imp	SBC imm	NOP imp		CPX abs	SBC abs	INC abs	BS6 zpg
F	BEQ rel	SBC iny	SBC ind			SBC zpx	INC zpx	SB7 zpg	SED imp	SBC aby	PLX imp			SBC abx	INC abx	BS7 zpg

Application Note (I)

The LCD_CTRL and LCD_CMD registers are used to control the LCD Drivers. The following example shows how to initiate the "MC141803" LCD driver.

The following bit settings are used for the LCD_CTRL register.

```
; ************
; * LCD CONTROL *
; ************
chip1   SET   7          ; select HD66410 series LCD driver 1:HD66410
chip0   SET   6          ; select SED15X (KSX)/MC141X series LCD driver 0:SED, 1:MC
clk     SET   5          ; LCD clock output selection
                         ; Just for MC141X series
cmod    SET   4          ; enable/disable LCD_CL
cs1     SET   3          ; control master LCD driver chip select
cs0     SET   2          ; control slave LCD driver chip select
a0      SET   1          ; Data/Command select 1:display data on D0~D7
                         ; Data/Command select 0:display control data on D0~D7
rw      SET   0          ; LCD Read/Write input 0:WRITE 1:READ
LCDCT   EQU   17h        ; LCD Control register
LCDCM   EQU   18h        ; LCD Command register
```

The following three macros define three different modes including "LCD COMMAND WRITE", "LCD DATA WRITE" and "LCD DATA READ" modes.

```
; ***************************
; LCDM WRITE COMMAND MODE
; LCD_A0=0 (command mode)
; LCD_WRB=0 (write mode)
; ***************************
LCD_WC      MACRO       REG
            IFMA        1
            LDA         REG
            ENDIF
            RMB         a0, LCDCT
            RMB         rw, LCDCT
            STA         LCDCM
            SMB         rw, LCDCT
            ENDM
```

```
; ***************************
; LCDM WRITE DATA MODE
; LCD_A0=1 (data mode)
; LCD_WRB=0 (write mode)
; ***************************
LCD_WD      MACRO       REG
            IFMA        1
            LDA         REG
            ENDIF
            SMB         a0, LCDCT
            RMB         rw, LCDCT
            STA         LCDCM
            RMB         a0, LCDCT
            SMB         rw, LCDCT
            ENDM
; ***************************
; LCDM READ DATA MODE
; LCD_A0=1 (data mode)
: LCD_WRB=1 (read mode)
; ***************************
LCD_RD      MACRO
            SMB         a0, LCDCT
            SMB         rw, LCDCT
            LDA         LCDCM
            RMB         a0, LCDCT
            ENDM
```

The following subroutine will initiate the ″MC141803″ LCD driver.

```
; ***************************
; *initial LCDM*
; ***************************
INI_LCDM:
        LDA     #01011001B  ; MC141X series LCD driver
        STA     LCDCT       ; enable LCD_CL, LCD_CL=32kHz
                            ; LCD_CS0 (master) enable
        LCD_WC  #76H        ; normal operation
        LCD_WC  #7FH        ; set oscillator enable
        LCD_WC  #2BH        ; set DC/DC converter on
        LCD_WC  #2DH        ; set internal regulator on
        LCD_WC  #31H        ; set internal contrast control on
        LCD_WC  #2FH        ; set internal voltage divider on
        LCD_WC  #33H        ; set 50kHz to get frame frequency
        LCD_WC  #29H        ; set display on
        LCD_WC  #36H        ; master clear GDDRAM
        LCD_WD  #0H         ; dummy write data
        LCD_WC  #04H        ; change to page 5 if want to clear icon line
        LCD_WC  #37H        ; master clear icons
        LCD_WD  #0H         ; dummy write data
        LCD_WC  #3DH        ; set display with icon line
        LCD_WC  #0H         ; set page 0
        LCD_WC  #23H        ; set col0 to seg119
        LCD_WC  #83H        ; set  column address 3
        RTS
```

Application Note (II)

The 5 wire serial I/O could be used to communicate with a FLEX™ high speed pager decoder. The following example will show you how to initiate the serial bus and exchange data with FLEX™ decoder.

```
; ****************************************
; * 5 wire serial I/O registers *
; ****************************************
SPICONF     EQU     32H
SPISP       EQU     33H
SPIPOUT3    EQU     34H
SPIPOUT2    EQU     35H
SPIPOUT1    EQU     36H
SPIPOUT0    EQU     37H
SPIIN3      EQU     38H
SPIIN2      EQU     39H
SPIIN1      EQU     3AH
SPIIN0      EQU     3BH
SPIST       EQU     0
SPIEN       EQU     1
SCKEDG      EQU     2
SPIFG       EQU     3
SPIRT       EQU     4
SPILEN0     EQU     5
SPILEN1     EQU     6
SPIMD       EQU     7
F_MSK       EQU     6                   ; Flex data exchange complete 1:Mask, 0:Non-mask
R_MSK       EQU     5                   ; Request 1:Mask, 0:Non-mask
; ****************************************
gSecurity   DS      4                   ; Checksum
gPackWrite  DS      1                   ; Buffer write index
gPacketQ    DS      240                 ; Message buffer
; ****************************************
; * Initial SPI *
; ****************************************
```

```
Initial_SPI:
            RMB     SPIMD, SPICONF      ; Master mode
            SMB     SPILEN0, SPICONF    ; set SPI data length to 32 bits
            SMB     SPILEN1, SPICONF
            LDA     #1
            STA     SPISP               ; set SCK frequency is X1 clock
            SMB     SPIEN, SPICONF      ; enable SPI function
            RTS
; ****************************************
; Send 4 bytes to FLEX™ decoder
; ****************************************
Send_4_bytes:
            SMB     SPIST, SPICONF
Send_4_bytes0:
            BBS     SPIFG, SPICONF, Send_4_bytes0
            SMB     SPIFG, SPICONF
            LDA     SPIOUT2
            EOR     gSecurity+2
            STA     gSecurity+2
            LDA     SPIOUT1
            EOR     gSecurity+1
            STA     gSecurity+1
            LDA     SPIOUT0
            EOR     gSecurity+0
            STA     gSecurity+0
            RTS
; ****************************************
; Read serial Bus Data to Buffer "gPacketQ"
; ****************************************
Read_SPI_In:
            LDX     gPackWrite
            LDA     SPIIN3
            STA     gPacketQ,x
            INX
            LDA     SPIIN2
            STA     gPacketQ,x
            INX
```

```
                LDA     SPIIN1
                STA     gPacketQ,x
                INX
                LDA     SPIIN0
                STA     gPacketQ,x
                INX
                STA     gPackWrite
                RTS
; ****************************************
; Host Send 4 bytes to FLEX™ decoder
; ****************************************
HOST_SD     MACRO  BYTE3@BYTE2@BYTE1@BYTE0
                LDA     BYTE3
                STA     SPIOUT3
                LDA     BYTE2
                STA     SPIOUT2
                LDA     BYTE1
                STA     SPIOUT1
                LDA     BYTE0
                STA     SPIOUT0
                JSR     Send_4_bytes
                ENDM
; ****************************************
XMOV        MACRO  ARG1@ARG2
                LDA     ARG2
                STA     ARG1
                ENDM
; ****************************************
; Initial FLEX™ Decoder
; ****************************************
Initial_Decoder:
                RMB     RESET_FLEX, PB
                NOP
                NOP
                SMB     RESET_FLEX, PB
                STZ     SPIOUT3
                SPIOUT2
```

```
        STZ     SPIOUT1
        STZ     SPIOUT0
        SMB     SPIST, SPICONF
Initial_Decoder0:
        BBS     SPIFG, SPICONF, Initial_Decoder0
        SMB     SPIFG, SPICONF
        XMOV    gSecurity+0@SPIIN0
        XMOV    gSecurity+1@SPIIN1
        XMOV    gSecurity+2@SPIIN2
        XMOV    gSecurity+3@SPIIN3
        HOST_SD #10H@#00H@#00H@#7FH
        HOST_SD #11H@#80H@#00H@#35H
        HOST_SD #12H@#80H@#00H@#01H
        HOST_SD #13H@#80H@#06H@#3DH
        HOST_SD #14H@#80H@#0EH@#05H
        HOST_SD #15H@#88H@#2EH@#01H
        HOST_SD #16H@#00H@#2AH@#7FH
        HOST_SD #17H@#00H@#2EH@#00H
        HOST_SD #18H@#00H@#22H@#00H
        HOST_SD #19H@#00H@#26H@#00H
        HOST_SD #1AH@#80H@#00H@#01H
        HOST_SD #1BH@#00H@#00H@#01H
        HOST_SD #24@#00H@#04H@#00H; Frame 58
        HOST_SD #80H@#05H@#2BH@#AAH; Capcode 306090
        HOST_SD #78@#00H@#00H@#01H; Enable user addresses 0
        HOST_SD #01H@#00H@#01H@#D8H
        HOST_SD #0FH@#00H@#00H@#00H
        HOST_SD #10H@#00H@#00H@#7FH
        HOST_SD #11H@#80H@#00H@#35H; Warm up 1
        HOST_SD #12H@#80H@#00H@#01H; Warm up 2
        HOST_SD #13H@#80H@#06H@#3DH; Warm up 3
        HOST_SD #14H@#80H@#0EH@#05H; Warm up 4
        HOST_SD #15H@#88H@#2EH@#01H; Warm up 5
        HOST_SD #16H@#00H@#2AH@#7FH; 3200 sps Sync Setting Packet
        HOST_SD #17H@#00H@#2EH@#00H; 1600 sps Sync Setting Packet
        HOST_SD #18H@#00H@#22H@#00H; 3200 sps Data Setting Packet
        HOST_SD #1BH@#00H@#00H@#01H; 1600 sps Data Setting Packet
```

```
        HOST_SD  #02H@#00H@#00H@#41H; Turn on decoder
        STZ      SPIOUT3
        XMOV     SPIOUT2@gSecurity+2
        XMOV     SPIOUT1@gSecurity+1
        XMOV     SPIOUT0@gSecurity+0
        SMB      SPIST, SPICONF
Initial_Decoder1:
        BBS      SPIFG, SPICONF, Initial_Decoder1
        SMB      SPIFG, SPICONF
        RMB      F_MSK, INTC
        RMB      R_MSK, INTC
        RTS
```

Holtek Semiconductor Inc. (Headquarters)
No.3, Creation Rd. II, Science-based Industrial Park, Hsinchu, Taiwan, R.O.C.
Tel: 886-3-563-1999
Fax: 886-3-563-1189

Holtek Semiconductor Inc. (Taipei Office)
11F, No.576, Sec.7 Chung Hsiao E. Rd., Taipei, Taiwan, R.O.C.
Tel: 886-2-2782-9635
Fax: 886-2-2782-9636
Fax: 886-2-2782-7128 (International sales hotline)

Holtek Semiconductor (Hong Kong) Ltd.
RM.711, Tower 2, Cheung Sha Wan Plaza, 833 Cheung Sha Wan Rd., Kowloon, Hong Kong
Tel: 852-2-745-8288
Fax: 852-2-742-8657

Holtek Semiconductor (Shanghai) Ltd.
7th Floor, Building 2, No.889, Yi Shan Rd., Shanghai, China
Tel: 021-6485-5560
Fax: 021-6485-0313

Holmate Technology Corp.
48531 Warm Springs Boulevard, Suite 413, Fremont, CA 94539
Tel: 510-252-9880
Fax: 510-252-9885