HITACHI MICROCOMPUTER SYSTEM

HMCS6800

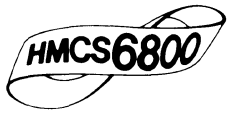
CMOS 8-BIT SINGLE CHIP MICROCOMPUTER UNIT(MCU)

HD63L05

USER'S MANUAL

H3L5MCU-EM

HITACHI



HITACHI MICROCOMPUTER SYSTEM

CMOS 8-BIT SINGLE CHIP MICROCOMPUTER UNIT(MCU)

HD63L05

USER'S MANUAL

—— PRELIMINARY ——

HITACHI

68-1-93

September 1982 68-1-93

Examples of circuit and examples of characteristics described in this manual are designed to explain typical application examples of the Hitachi Semiconductors.

Pay attention to the following points in using this manual.

1. The contents of this manual may be changed without prior notice.

2. No part or all of this manual may be reproduced or republished in any form without prior permission of the company.

3. The company is not responsible for any damage that may be caused by an accident on the users' side.

4. This manual does not provide any guarantee of implementation of industrial ownership on other right or approval of implementation right.

Foreward

The HD63L05 is a 3V CMOS single-chip microcomputer for battery-operated systems with the same instruction set as the HD6805S and abundant on-chip functions with extremely low power consumption.

The HD63L05 has architecture that is suitable for the controller field in which bit input / output and status flag processing on software are important and it also includes the LCD drivers, analog-to-digital converter, etc. which are effective in the reduction of the number of system parts.

Table of Contents

1. General

1.1 Features of the HD63L05

The HD63L05 is a 3V CMOS single-chip microcomputer for battery-operated systems with the same instruction set as the HD6805S and abundant on chip functions with extremely low power consumption.

This microcomputer operates from a 3V power supply with extremely low power consumption and has two lower power consumption mode; a software-controlled halt mode and a standby mode which is controlled through the input terminal.

The HD63L05 contains two on chip oscillators, an 8-bit timer with a programmable 7-bit prescaler, 20 input/output ports, 4 kbyte ROMs, 96 byte RAMs, LCD drivers and an 8-bit analog-to-digital converter. Due to the mask-option, the functions of the limited terminals are substitutable by other functions such as LCD drivers, analog inputs, or digital outputs.

The HD63L05 instruction set is compatible with Hitachi's NMOS 8-bit single-chip microcomputer HD6805S in the operation code level. The processor keeps the advanced features of the HD6805 family's instruction set, such as powerful bit manipulation.

Table 1-1 Features of the HD63L05

Function		HD63L05F1
Package		FP-80
ROM (Bytes)		3760
RAM (Bytes)		96
Input/Output (I/O)	I/O	20
	0	19 (mask-option)
LCD driver		static, 1/3 bias 1/3 duty max 17 segments
Analog-to-digital converter		8-bit A/D converter max 8 channels.
Timer		8-bit timer (7 bit prescaler)
Interrupt		External 1 Timer 1 A/D 1 Time Base 1
Register	Accumulator A	8 bits × 1
	Index X	8 bits × 1
	Stack pointer SP	5 bits × 1
	Program counter PC	12 bits × 1
Oscillator	OSC 1	RC, crystal
	OSC 2	crystal

Package

FP-80

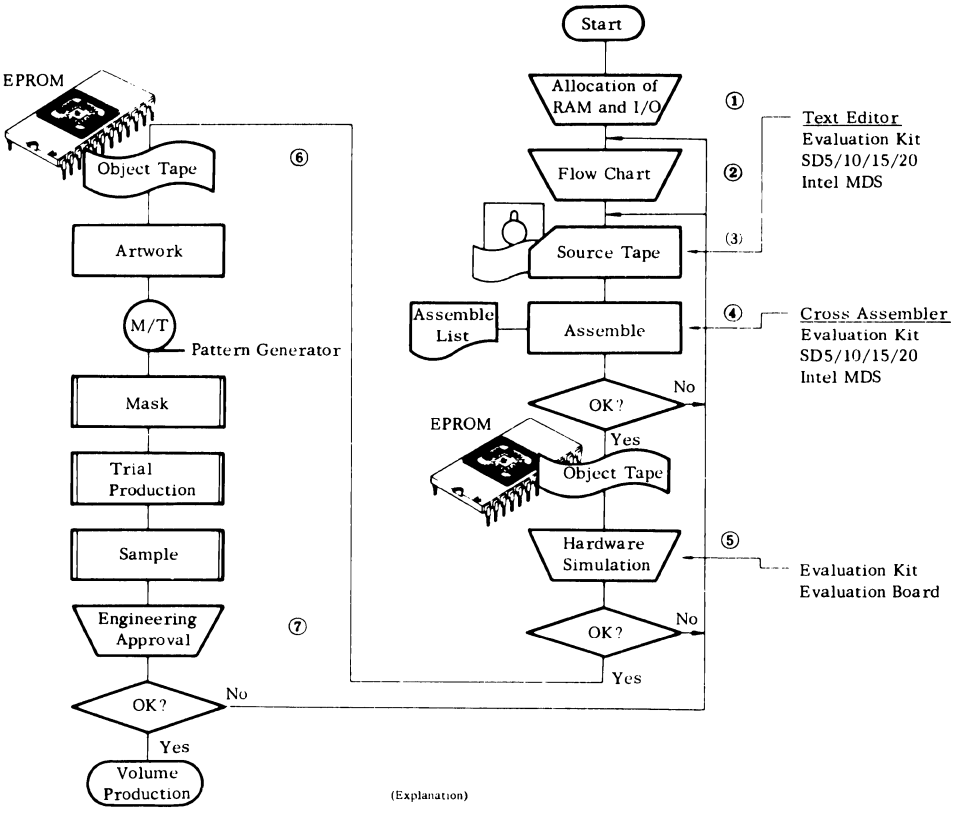
* DP-64S,FP-64 are under development.

1. 2 Design Procedure and Supporting Tool

The cross assembler and the hardware simulator using various types of computer are prepared by the company as supporting systems to develop users' programs.

Users' programs are mask programed into the ROM and delivered as the LSI by the company.

Fig.1-1 shows the typical program design procedure and Table 1-2 shows the system development supporting tool for the HD63L05 which are used in these processes.



(Explanation)

1. When the user programs the system, the predetermined functions are assigned to the I/O pin and the RAM before the programming.
2. A flow chart is designed to achieve the predetermined functions and the flow chart is coded by using the prenumeric code.
3. The coded flow chart is punched into the card or the paper tape or written into the floppy disk, to generate a source program.
4. The source program is assembled by the resident system (evaluation kit) or the cross system, to generate the object program. In this case, errors during the assembling are also detected.
5. Hardware simulation is performed to confirm the program. Either the evaluation kit or the evaluation board is used for hardware simulation.
6. The completed program is sent to the company in the form of EPROM or the object tape.
7. Options such as ROM is masked by the company, LSI is testatively produced and the sample is handed in to the user. After the user has evaluated the sample and confirmed that the program is correct, mass production is started.

Figure 1-1 Program Design Procedure

Table 1-2 System Development Supporting Tool

Type Name	Resident System		Cross System
	Evaluation Kit	Evaluation Board	
HD63L05F1	H3L5EVT1 (Hardware) + S3L5MIX1-R (Software)	H3L5EV00	Emulator for SD5 Cross assembler editor for SD5/10/15/20 Cross assembler editor for Intel MDS

1.3 Block Diagram

Input signals and output signals of the MCU are described below.

- Vcc, Vss
  power is supplied to the MCU by using these two terminals. Vcc has a voltage of 3.0V±0.8V, and Vss is grounded.

- $\overline{INT}$
  This terminal is used to enboke an external interruption to the MCU. See Section "2.8Interrupts" for details.

- XTAL, EXTAL
  These terminals are control input terminals to the built-in clock circuit. A crystal(400 kHz typ.) or a resister is connected to these terminals in accordance with stability of internal oscillation. See Section"2.7 Internal Oscillator Options" for how to use these terminals.

- TIMER
  This terminal is an external input terminal to count down the internal timer circuit. See "2.4 Timer" for the detailed description of the timer circuit.

- $\overline{RES}$
  Used to reset the MCU. See "2.5 Resets" for details.

- STANDBY
  An external input terminal used to stop the MCU and hold data. For details, see "2.7 Internal Oscillator Options".

- A/D Input Terminals (CH1~CH8)
  Input terminals for analog voltages needed for A/D conversion. These may also be used as level check inputs under program control. See "2.10A/D Converter" for details.

- VRH, VRL
  Reference voltages for A/D conversion are applied to these two terminals. For details, see "2.10 A/D Converter".

- CC1, CC2
  An offset compensating capacitor (300pF typ.) is connected between CC1 and CC2. For details, see "2.10A/D Converter".

- NUM
  This terminal is not used for user application. Connect it to Vcc.

- Input/Output Terminals (A0 A7,B0 B7,C0 C3)
  These 20 terminals consist of two 8-bit ports and one 4-bit port. Each of them may be used as an input or output under program control of the data direction register. For details, see "2.9 Input/Output".

●Liquid Crystal Driver Terminals (COM1~COM3, SEG1~SEG17)
COM1∿COM3 are for driving common electrodes, while SEG1 SEG17 are for driving segments. SEG1∿SEG17 can be used as outputs by mask-option and SEG13∿SEG17 can be used as analog inputs for A/D converter by mask-option.

●V1, V2
These are terminals for LCD driver. V1 and V2 are connected to Vcc via capacitors (0.1uF each). These two terminals can be used as outputs or analog inputs by mask-option.

●VCH
Output termianl from internal voltage regulator. A capacitor (0.5 μF) is connected between VCH and Vcc.

●E
System clock output (cycle clock 100 kHz typ.)
This NMOS open-drain output stays at "Low" level when the MCU is in halt status or standby.

HD63L05 Block Diagram is shown in Figure 1-2.

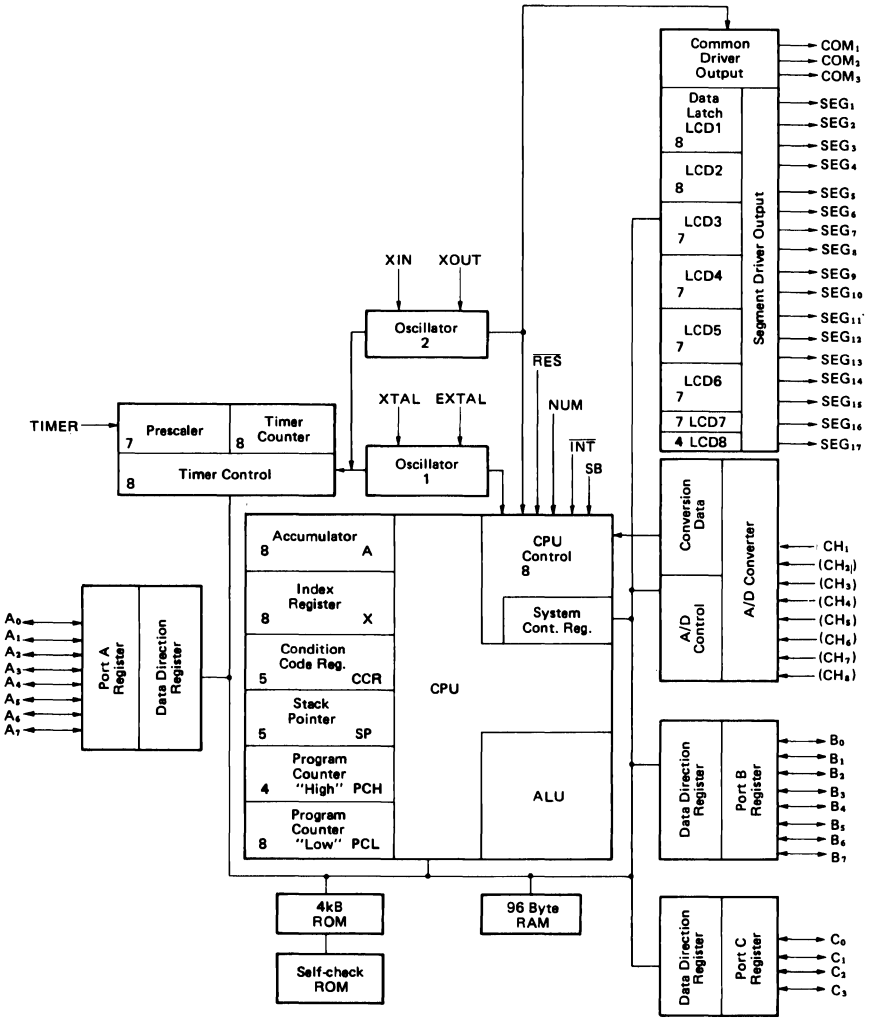


Figure 1-2 HD63L05 Block Diagram

2. Architecture

2.1 Memory

The MCU memory is configured as shown in Fig.2-1. During the processing of an interrupt, the contents of the MCU resisters are pushed onto the stack in the order shown in Fig.2-2. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

Cautions
It is not possible to change the contents of the Write Only Register (For example, the Data Direction Register of the I/O port) of the HD63L05 by applying the Read/Modify/Write instructions, BSET, or BCLR.

For preventing the system from wild running, don't address the Not Used area of the memory map.

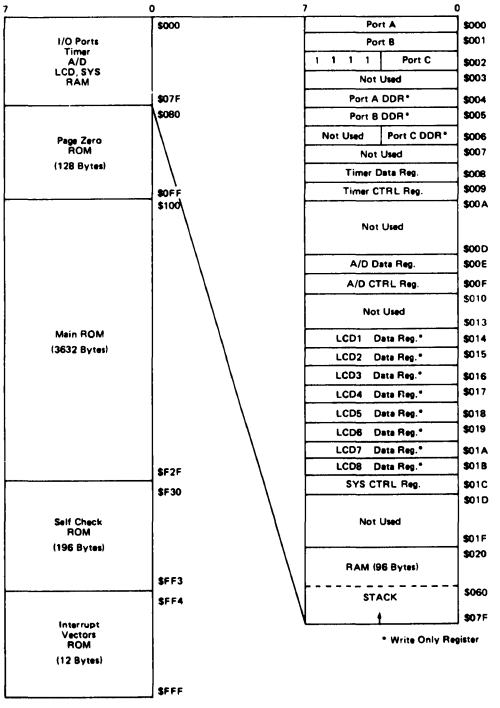


Figure 2-1 Memory Map

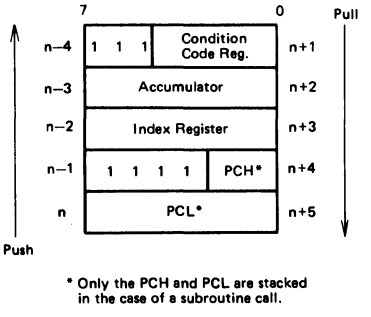


Figure 2-2 Interrupt Stacking Order

2.2 Registers

The MCU has five registers available to the programmer. They are shown in Figure 2-3 and are explained in the following paragraphs.

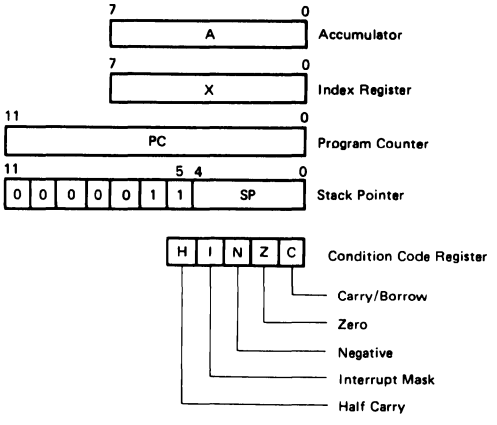


Figure 2-3 Programming Model

Accumulator (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

Index Register (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/ modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage register.

Program Counter (PC)

The program counter is a 12-bit register that contains the address of the next instruction to be executed.

Stack Pointer (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The most significant bits of the stack pointer are permanently set to 0000011. During a MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 which allows the programmer to use 15 levels of subroutine calls.

Condition Code Register (CC)

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

Half Carry (H)

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bit3 and bit4.

Interrupt (I)

This bit is set to mask the internal interrupts and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set,it is latched and will be processed as soon as the interrupt bit is reset.

Note

CLI(clear interrupt mask bit) is used to allow the interruption from the instruction after next. SEI(set interrupt mask bit) masks the interruption from next instruction.

Negative (N)

Used to indicate that the result of the last arithmetic, logical of data manipulation was negative (bit7 in result equal to logical one).

Zero (Z)

Used to indicate that the result of the last arithmetic, logical of data manipulation was zero.

Carry/Borrow (C)

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch insuructions, shifts, and rotates.

2.3 System Control Register

Apart from the registers for program operation explained above, there is a register that controls system operation. Its configuration is shown in Figure 2-4.

- Time Base Instruction Request Flag (TB INT)
  Stores an interruption request from the time base which is selected by the TB select bit and is cleared by system reset or by program. If the TB MASK bit or I (Interrupt bit in the CCR) is set, the interruption request is not acknowledged. Only logical "0" can be written into this bit by program.

- Time Base Instruction Mask (TB MASK)
  If this bit is set, any interrupt request from the time base is not acknowledged.

- Time Base Select Bit (TB SELECT)
  This bit selects the time base. In logical "1", an interruption from the 1-second cycle time base is acknowledged. In logical "0", 1/16 second cycle time base is acknowledged.

- Time Base Reset Bit (TB RESET)
  This bit resets the frequency divider behind the 32 kHz oscillator. When this bit is set, one shot reset pulse is generated by the hardware. Then it resets the frequency divider and after that, the frequency divider restarts. As this bit has not a register, the CPU always reads this bit as logical "0".
  The frequency divider also provides the system clocks to the A/D converter and LCD drivers. So, it is needed to pay an attention when "TB RESET" is used.

- Halt (HALT)
  Used to halt the CPU. When this bit is set, the registers are saved onto the stack in the same sequence as interruption processing. After all registers have been saved, the CPU halts and is wait-for-interrupt state.
  If this bit is reset by an external interruption or an internal interruption, the CPU restarts operating. By using the Halt function with Time Base Interruption, the CPU can operate intermittently itself.

- EXT
  When the form of output port is selected by DUTY selecting bit, ϕWRITE can be got every time data is written into LCD register in the case that this bit is "1". ϕWRITE can be used with the designation of pin location as the clock for writing in the case of transferring data of LCD register to the outside. Normally, EXT is reset.

- Duty Select Bit (DUTY)
  The LCD drive signal is based on 1/3 bias - 1/3 duty. However, there are switching circuits built in for static drive signal and output ports. For details, see the information given in "LCD Circuit".

Note : The EXT bit and the DUTY bits have to be initiallized in a 1 m second from the beginning of the system reset when the static drive signal or output port is selected.

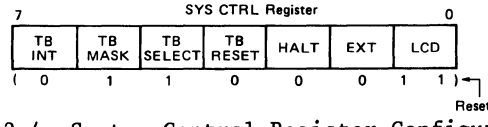


Figure 2-4 System Control Register Configuration

2.4 Timer

The MCU timer circuitry is shown in Figure 2-5. The 8-bit counter is loaded under program control and counts down toward zero as soon as the colck input is applied. When the timer reaches zero the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present MCU state in the stack, fetching the timer interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the timer control register. The interrupt bit (I bit) in the condition code register will also prevent a timer interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal signal (ϕ_2 or $\phi_{32}k$). When the internal clock signal is used as the source, the clock input is gated by the input applied to the timer input terminal; this permits easy measurement of its pulse width. There are two types of internal clock signals (ϕ_2 and $\phi_{32}k$) to allow timer operation when the CPU is halted. (ϕ_2 is active when OSC1 is not stopped.) These clock signals are under program control. Note that the timer operation is asynchronous to the CPU when the clock signal is from external source or $\phi_{32}k$.

A 7-bit prescaler is provided to extend the timing interval up to a maximam of 128 counts before being applied to the timer. The number of prescaling counts can be program controlled by the lower 3 bits within the timer control register. The timer continues to count past zero and its present count can be monitored at any time by monitoring the timer data register. This allows a program to determine the length of time since a timer interrupt has occurred and not disturb the counting process.

At the time of resetting, the prescaler and the counter are all initialized to logical "1". The timer interruption request bit is cleared and the timer interruption mask bit is set. The timer interrupt request bit (bit 7 of Timer Control Register) is set to logical "1" when timer count reaches zero, and is cleared by program or by system reset. Only logical "0" can be written into this bit by program. The bit 6 of Timer Control Register is writable by program. Both of these bits can be read by CPU.

2.5 Resets

The MCU can be reset either by initial powerup or by the external reset input ($\overline{RES}$). All the I/O ports are initialized to Input mode (DDRs are cleared) during RESET.

Upon power up, a minimum of 150 milliseconds is needed before allowing the reset input to go "High". This time allows the internal oscillator (OSC1) to stabilize. Connecting a capacitor to the $\overline{RES}$ input as shown in Fig. 2-8 will provide sufficient delay.

2.6 Self Check

The self check capability of the MCU provides an internal check to determine if the port is functional. Connect the MCU as shown in Figure 2-9 and monitor the output of port C bit 3 for an oscillation of approximately 0.5 Hz. This self check capabibity also provides the internal state of the MCU to measure the LSI current. After a system reset, the MCU goes into each current measurement mode by the combination of the control switches. The LSI current can be measured when the NUM is returned to Vcc after setting of the current mode.

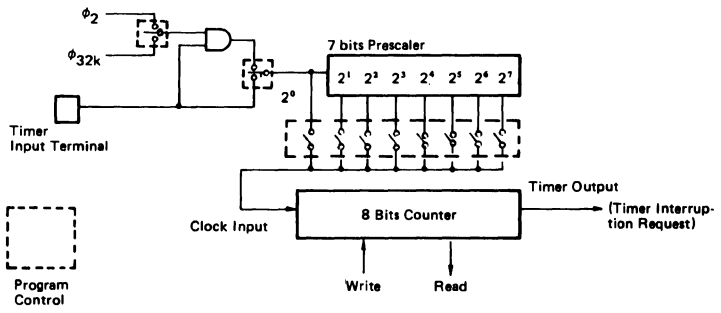


Figure 2-5 Timer Block Diagram

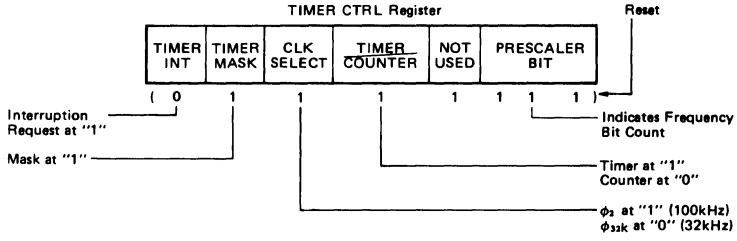


Figure 2-6 Timer Control Register Configuration

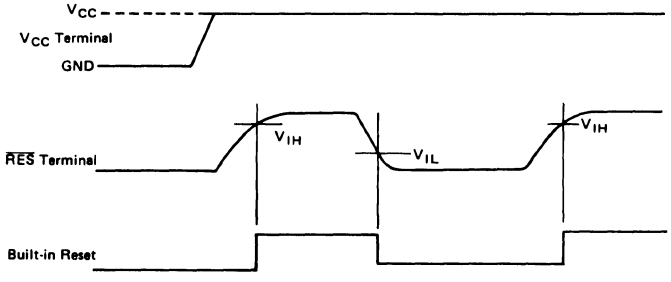


Figure 2-7 Power and Rest Timing

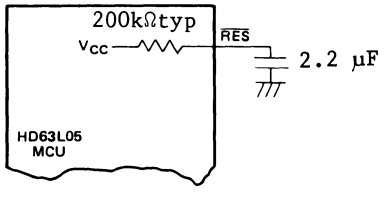
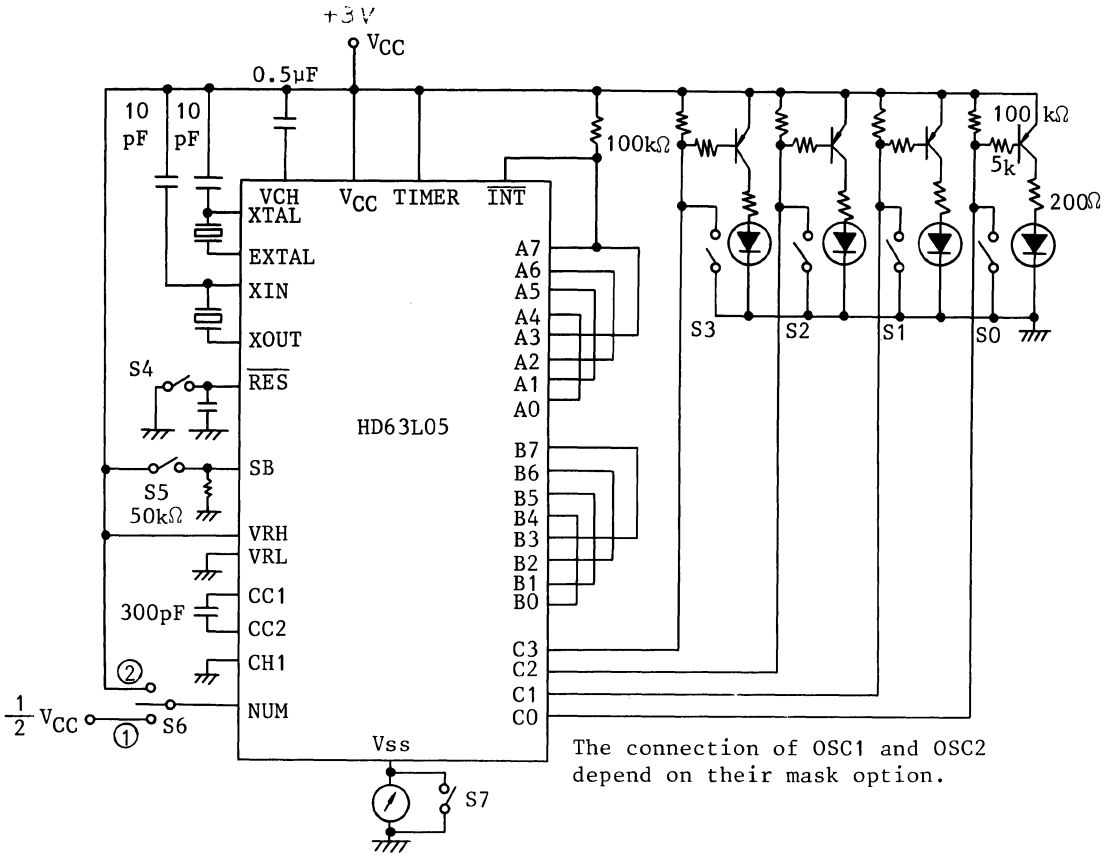


Figure 2-8
Input Reset Delay Circuit



		Selection of Switch							
		S0	S1	S2	S3	S4	S5	S6	S7
LSI Function		×	×	×	×	×	×	①	○
LSI Current	During operation	○	×	×	×	○→×	×	①→②	×
	Halt	○	○	○	×	○→×	×	①→②	×
	A/D	○	○	×	×	○→×	×	①→②	×
	Stand-by	○	○	○	×	○→×	×→○	①→②	×

× :OFF ○ :ON → : Change the state

Figure 2-9 Self Check Connections

2.7 Internal Oscillator Options

The MCU incorporates two oscillators : oscillator 1 for system clock supply and oscillator 2 for time base, analog to digital converter, and LCD drivers.

- Oscillator 1 (OSC1; XTAL, EXTAL)
  The internal oscillator circuit can be driven by an external crystal or resistor depending on the stability. A manufacturing mask option is avilable to provide better matching between the external components and the internal oscillator. The oscillator 1 can stop when power is applied in either Halt or Standby status. Figure 2-10 shows the connection. A resistor selection graph is given in Figure 2-11.

- Oscillator 2 (OSC2; XIN, XOUT)
  Clocks for time base, LCD drivers, an analog-to-digital converter, and a timer can be supplied by the OSC2 (32.768kHz crystal) or by the OSC1 through the frequency divider. In Halt status, oscillator 2 operates and permits the operation of the peripheral modules with low power consumption. In Standby status, this oscillator stops when power is applied. Fugure 2-12 shows the connection and the relation between oscillator 1 and oscillator 2 is shown Figure 2-13 and Figure 2-14.

Note

When OSC2 is not available or OSC1 is the crystal option, OSC1 is not allowed to stop at Halt. The accuracy of the time base is kept when OSC2 is 32.768 kHz crystal oscillator.

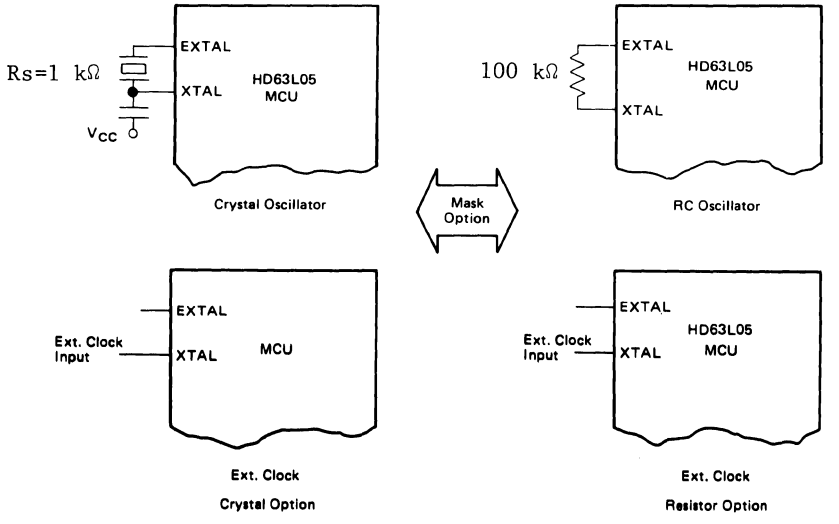


Figure 2- 10 Mask Option for Oscillator1

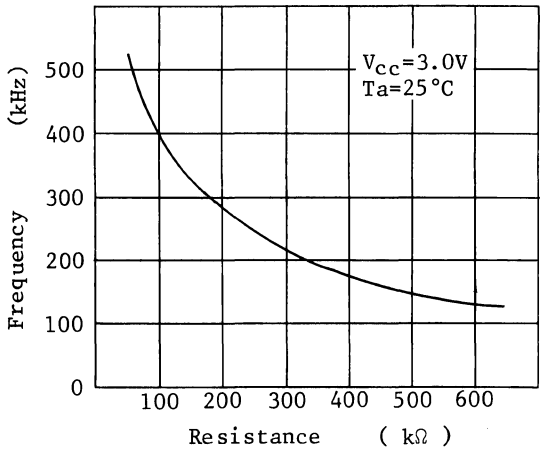


Figure 2-11 Typical Resistor Selection Graph

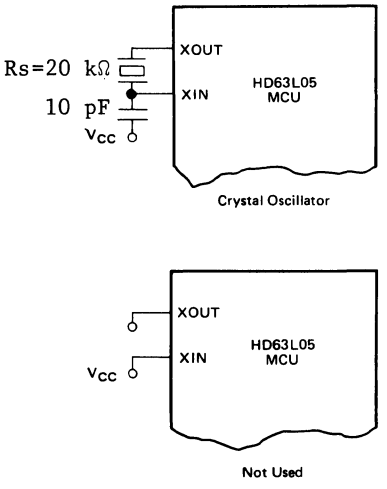


Figure 2-12 Connection of Oscillator2

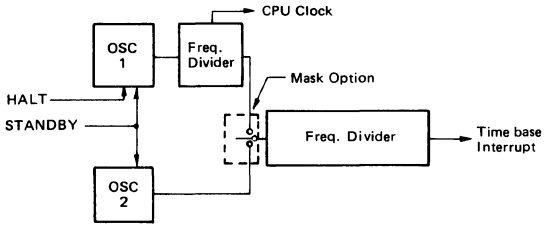


Figure 2-13 Relation between Oscillator1 and Oscillator2

Mask Option	When OSC1 is X-TAL						When OSC1 is RC					
	OSC2 Not Available			OSC2 Available			OSC2 Not Available			OSC2 Available		
System / State	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral	OSC1	CPU	Peripheral
During System Operation	O	O	O	O	O	O	O	O	O	O	O	O
At Halt	O	X	O	O	X	O	O	X	O	X	X	O
At Standby	X	X	X	X	X	X	X	X	X	X	X	X

(NOTE) O ..... run X ..... stop

Figure 2-14 Oscillator2 Mask-option and System Operation

2.8 Interrupts

There are six different interruptions to the MCU : external interruption via external interrupt terminal ($\overline{INT}$), internal timer interruption, interruption by termination of A/D conversion, time base interruption (2 types), and software interruption by an instruction (SWI).

When any interrupt occurs, processing is suspended, the present MCU state is pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The interrupt service routines normally end with a return from interrupt instruction (RTI) which allows the MCU to resume processing of the program prior to the interrupt. Table 2-1 provides a listing of the interrupts, their priority, and the vector address that contains the starting address of the appropriate interrupt routine.

Figure 2-15 shows the system operation flow, in which the portion surrounded with dot-dash lined contains interruption execution sequence.

Note : A clear interrupt bit instruction (CLI) allows to suspend the processing of the program by an interruption after execution of the next instruction while a set interrupt bit instruction (SEI) inhibits any interrupts before execution of the next instruction. When a mask bit of a control register is cleared by an instruction, interruption is allowed before execution of the next instruction.

- Acknowledging interrupts in HALT Status
  In HALT status, the CPU is stopped but the peripherals are operating. When an interrupt is acknowledged, the CPU is activated and executes interruption service matching the interruption condition by means of vectoring.

- Acknowledging interrupts in Standby Status
  In Standby status, the system is all stopped with power supplied to it. Therefore, any interruption request (including $\overline{RES}$) is not acknowledged.

Table 2-1 Interruption Priority

Interruption	Priority	Vector Address
$\overline{RES}$	1	$FFFE, $FFFF
SWI	2	$FFFC, $FFFD
$\overline{INT}$	3	$FFFA, $FFFB
TIMER	4	$FFF8, $FFF9
A/D	5	$FFF6, $FFF7
TIME BASE	6	$FFF4, $FFF5

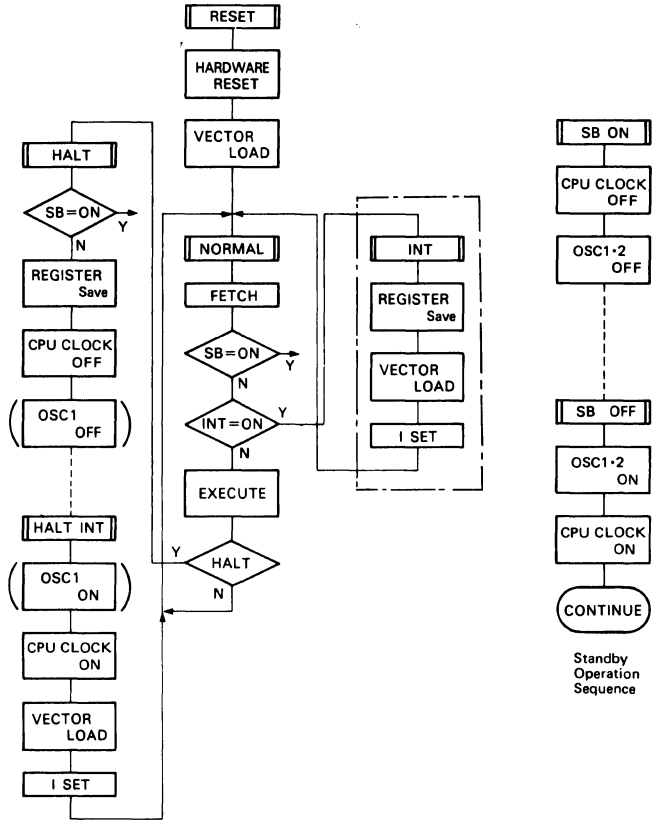


Figure 2-15 System Operation Flowchart

2.9 Input/Output (Port A,B,C)

There are 20 input/output terminals. All pins are programmable as either inputs or outputs under software control of the data direction register. When programmed as outputs, all I/O pins read latched output data regardless of the logic level at the output pin due to output pin due to output loading (see Figure 2-16).

Manufacturing mask-options are available to provide better matching between the external components and the I/O ports. Figure 2-17 shows the configuration of I/O ports. As the output is on/off controlled by a data direction register, an I/O port may directly be applied as an input terminal. No problem is involved with the input if both "High" and "Low" levels are applied. For one level input, the user must specify the use of a pull-up PMOS for "Open/Low" input application.

Two types of the output PMOS drivability (I_{OH}=-100μA or I_{OH}=-10μA) or NMOS open drain can be selected by mask option for output application All I/O Pins are initialized as inputs by system reset.

2.10 A/D Converter

The MCU incorporates an 8 bits A/D converter based on the resistor ladder system. Figure 2-18 shows its block diagram.

The "High" side of reference voltage is applied to V_{RH}, while the "Low" side of reference voltage is applied to V_{RL}. The reference voltage is divided by resistors into voltages matching each bit, which is compared with analog input voltage for A/D conversion. As the analog input voltage is applied to the MOS gate of the comparator through the analog multiplexer, this voltage comparison system achieves high input impedance. Offset of the comparator are compensated for each bit by external capacitor which is connected between CC1 and CC2.

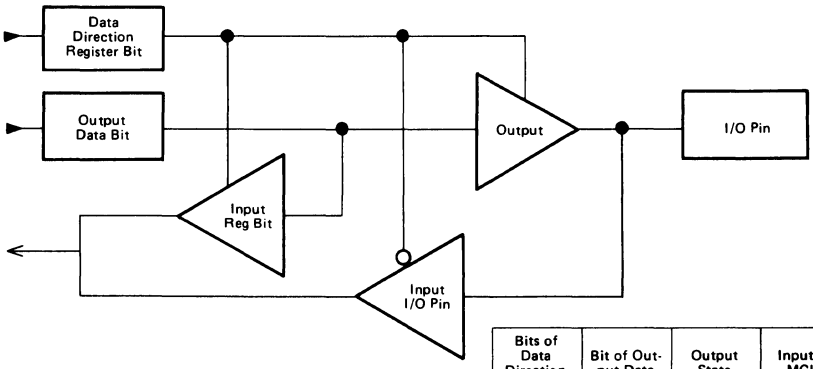
The A/D DATA Register stores the results of an A/D conversion or can be set 8 bit data for programmed comparator. These functions are controlled by software-controlled A/D CTRL Register. Figure 2-19 shows the configuration of the A/D control register.

- A/D INT
  The A/D INT bit is set to logical "1" after completion of A/D conversion and is cleared by program or by system reset. Only logical "0" can be written into this bit by program.

- A/D MASK
  If this bit is set, interrupt from the A/D converter is not acknowledged. This bit can be written by program.

- CNV
  To start A/D conversion, set this bit to logical "1". During conversion, data of this bit stays at "1". The bit is automatically reset to "0" when the A/D conversion ends. In A/D conversion, supply voltage is applied to the comparator only when CNV="1". The digital data which is obtained by the A/D conversion is held in the A/D data register. This data is reset when the CNV is set to "1" again.

- Auto/Program
  Used to select either auto-run 8 bits A/D conversion or 8 bit programmed comparator operation (Auto 8 bits A/D conversion at "0").



Bits of Data Direction Register	Bit of Output Data	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 2-16 Port I/O Circuit

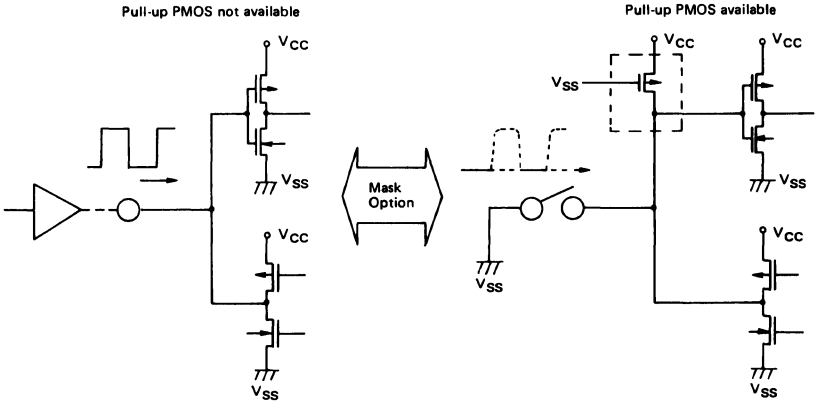
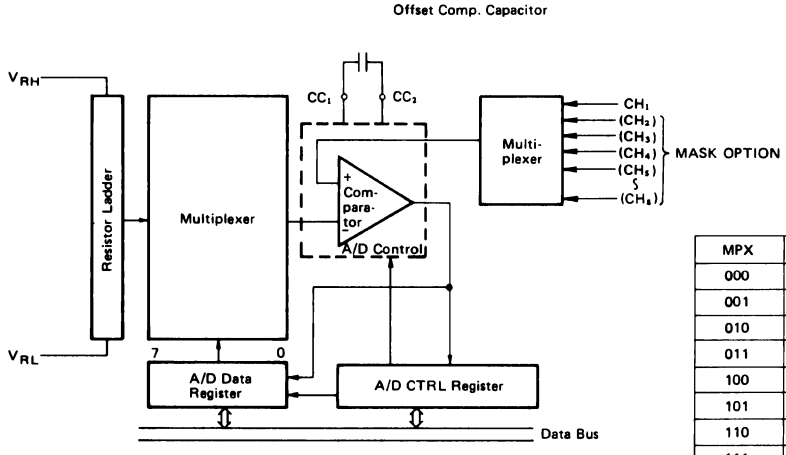


Figure 2-17 Selection of the Input Configuration for I/O Port



MPX	Channel
000	CH_1
001	(CH_2)
010	(CH_3)
011	(CH_4)
100	(CH_5)
101	(CH_6)
110	(CH_7)
111	(CH_8)

Figure 2-18 8 Bits A/D Converter Block Diagram

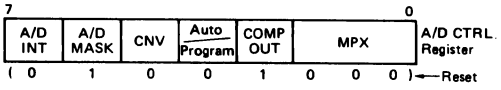


Figure 2-19 A/D Control Register Configuration

●COMP OUT

The result of comparator operation under program control can be read from this bit (Logical "1" means that input voltage is higher than programmed reference voltage).

●MPX

Used to select 8-channel analog inputs. The multiplexer is an analog switch based on CMOS. Note that the analog inputs from CH2 to CH8 are mask option which CH1 is CH7 and CH8 are not available because these two terminals are used for LCD power supply.

2.11 LCD Circuit

The system configuration of the LCD circuits is shown in Figure 2-20 Segment data for display are stored in data registers LCD1 to LCD8. Since the circuits are connected to the output terminals via pin location block, the user may specify a combination of data to be multiplexed to the segment output terminals.

The bit data of the LCD register is combined with the timing clock (ϕ_1, ϕ_2 or ϕ_3) and three combined bit data are gathered to make a segment output data for 1/3 bias - 1/3 duty driving in the pin location block. In case of static LCD drive of output port, timing is always fixed at ϕ_1 (always "High") and one bit data of the LCD register is transferred for an output terminal.

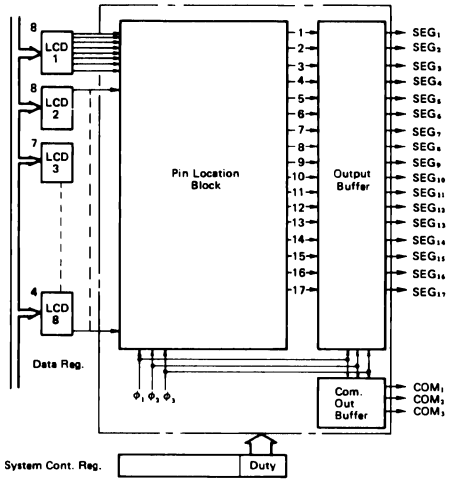
Note that the output terminals from SEG13 to SFG17 are mask option while the others (SEG1 to SEG12) are always available when the Duty bits are "01" or "11".

When the form of output port is selected by Duty bit ("00"), ϕWRITE can be got every time data is written into LCD1 register in the case that EXT bit is "1". As LCD1 register has 8 bits latches, it is easy to transfer the internal 8 bits data to external devices via output ports, with automatically generated write clock ϕWRITE. The cycle clock pulse can be also available as an internal data source for the output terminal when output port is selected.

Assignment of segment terminals to the bits of the LCD data register, including the case where they are used as output terminals, is to be specified by the user when he orders masks.

2.12 Liquid Crystal Driver Wave Forms

The LCD circuit is based on 1/3 bias - 1/3 duty driving. Figure 2-21 shows the common electrode output signal waveforms (COM1, COM2, COM3), segment signal waveforms (SEG1 to SEG17) and LCD bias waveforms (between COM and SEGMENT).



System Cont. Duty	Contents of $SEG_1 \sim SEG_{17}$
00	OUTPUT PORT
01	STATIC LCD
10	———
11	1/3 Bias 1/3 Duty LCD

Figure 2-20 LCD Circuit System Configuration

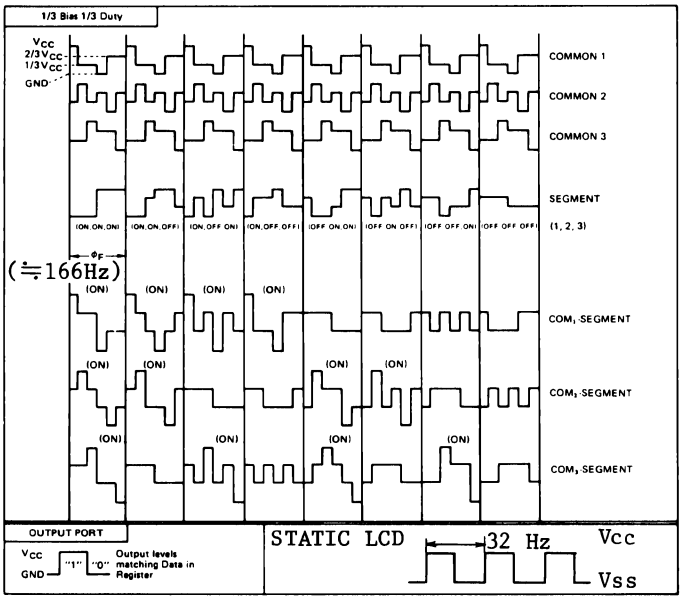


Figure 2-21 LCD Driving Waveforms

2.13 Bit Manipulation

The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET,BCLR). Any bit in the page zero read only memoly can be tested,using the BRSET and BRCLR instructions,and the program branches as a result of its state. This capability to work with any bit in RAM,ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines.

NOTE

It is needed to pay attention to the system control register, the timer control register,and A/D control register when BSET,BCLR,or Read/Modify/Write instructions are applied to them. If own interrupt request occured onto the interrupt request bit (bit7) of the control register between read cycle and write cycle of these instructions,the bit7 might be cleared in the write cycle and not acknowledged by CPU.

2.14 Addressing Modes

The MCU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the fillowing paragraphs.

●Immediate
Refer to Figure 2-22. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

●Direct
Refer to Figure 2-23. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing.

●Extended
Refer to Figure 2-24 . Extended addressing is used to reference any location in memory space. The EA is the contents of the two byte following the opcode. Extended addressing instructions are three bytes long.

●Relative
Refer to Figure 2-25. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken, Rel=0. When a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

●Indexed (No Offset)
Refer to Figure 2-26 . This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

●Indexed (8-bit Offset)
Refer to Figure 2-27. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 lowest memory locations are accessable. These instructions occupy two bytes.

●Indexed (16-bit Offset)
Refer to Figure 2-28 . This addressing mode calculates the EA by adding the contents of two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

●Bit Set/Bit Clear
Refer to Figure 2-29 . This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the addresss in page zero.

● Bit Test and Branch

Refer to Figure 2-30 . This mode of addressing applies to instructios which can test any bit in first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode . The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

● Implied

Refer to Figure 2-31 . The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on accumulator and the index register are included in this mode of addressing. In addition,contril instructions such as SWI,RTI belong to this group. All implied addressing instructions are one byte long.

2.15 Instruction Set

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**
  Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 2-2.

- **Read/Modify/Write Instructions**
  These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 2-3.

- **Branch Instructions**
  The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 2-4.

- **Bit Manipulation Instructions**
  These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 2-5.

- **Control Instructions**
  The control instructions control the MCU operations during program execution. Refer to Table 2-6.

- **Alphabetical Listing**
  The complete instruction set is given in alphabetical order in Table 2-7.

- **Opcode Map**
  Table 2-8 is an opcode map for the instructions used on the MCU.

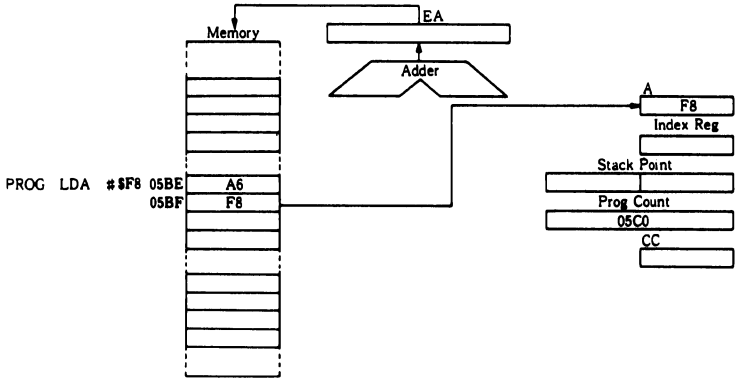


Figure 2-22 Immediate Addressing Example

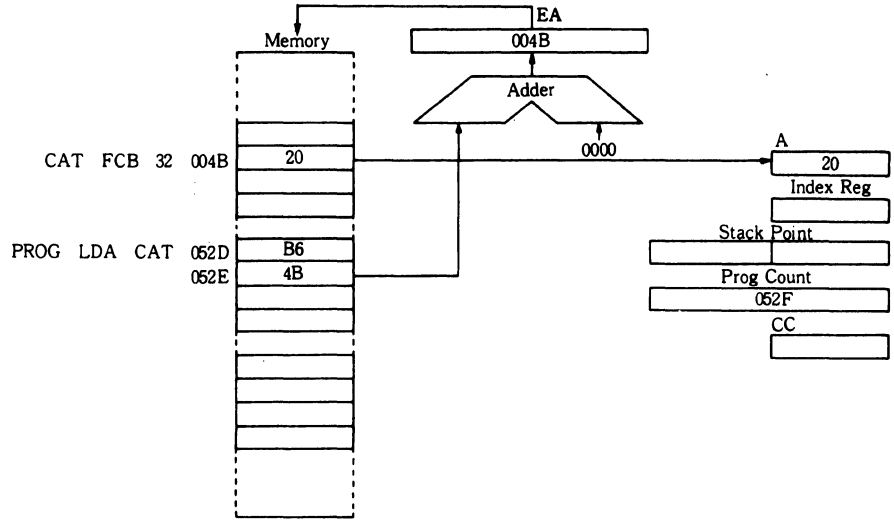


Figure 2-23 Direct Addressing Example

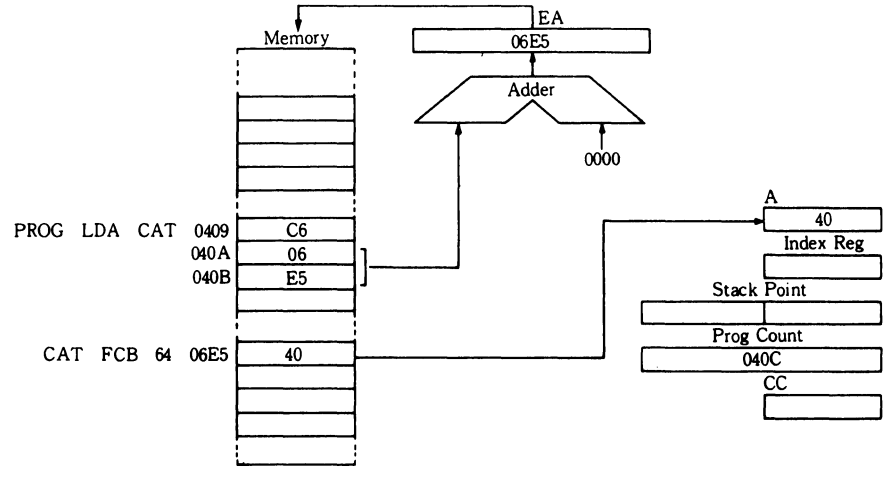


Figure 2-24 Extended Addressing Example

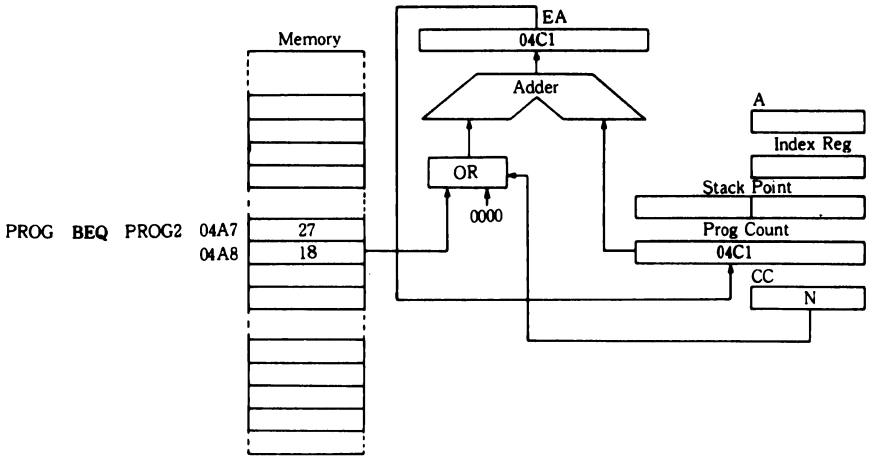


Figure 2-25 Relative Addressing Example

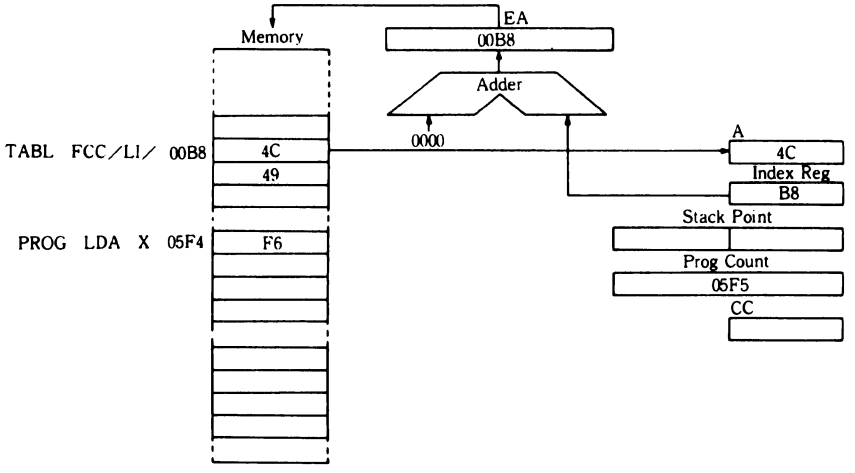


Figure 2-26 Indexed (No Offset) Addressing Example

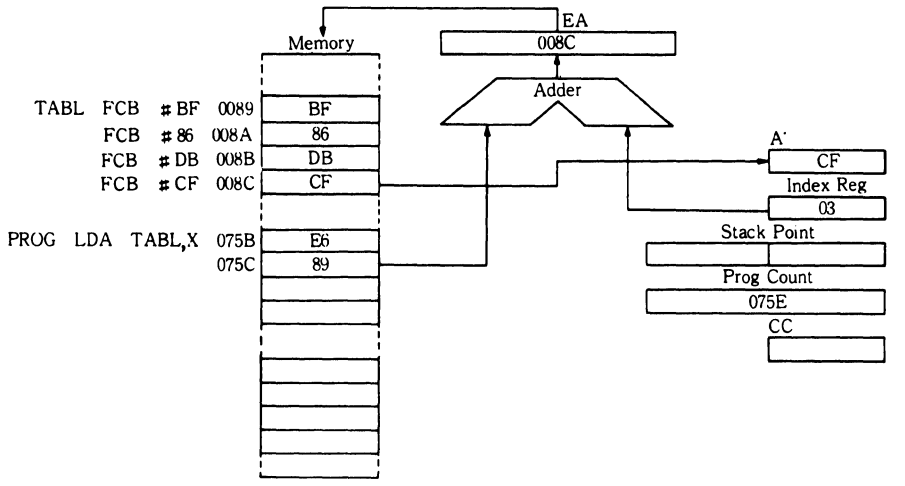


Figure 2-27 Indexed (8-Bit Offset) Addressing Example

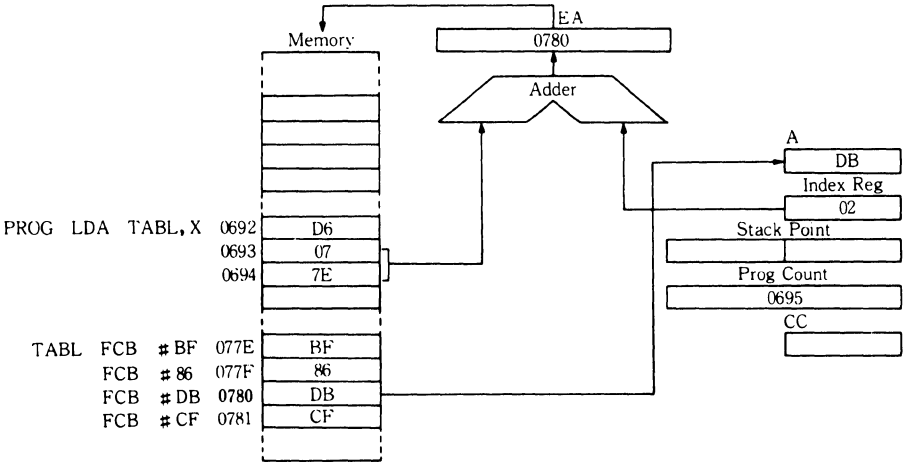


Figure 2-28 Indexed (16-Bit Offset) Addressing Example

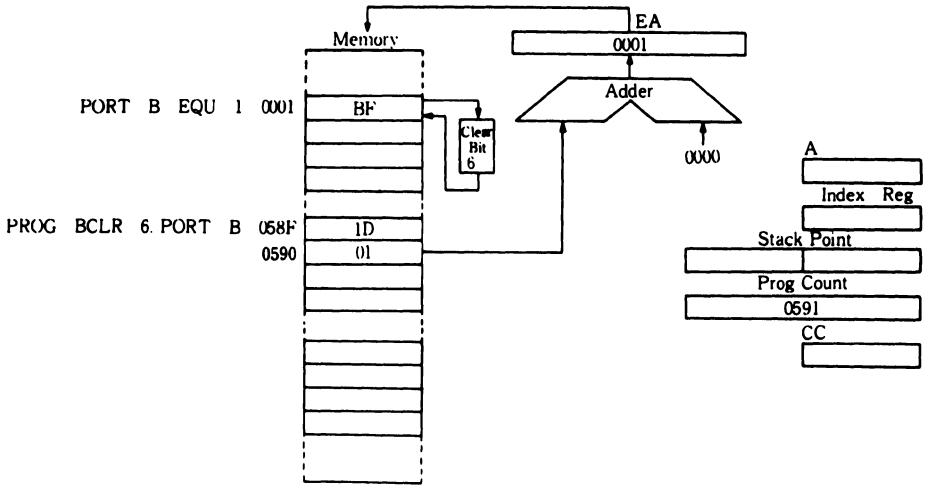


Figure 2-29 Bit Set/Clear Addressing Example

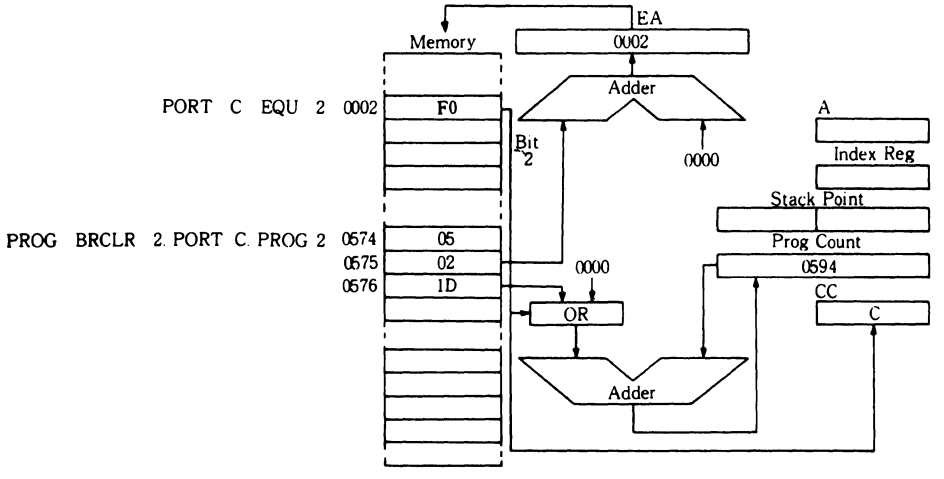


Figure 2-30 Bit Test and Branch Addressing Example

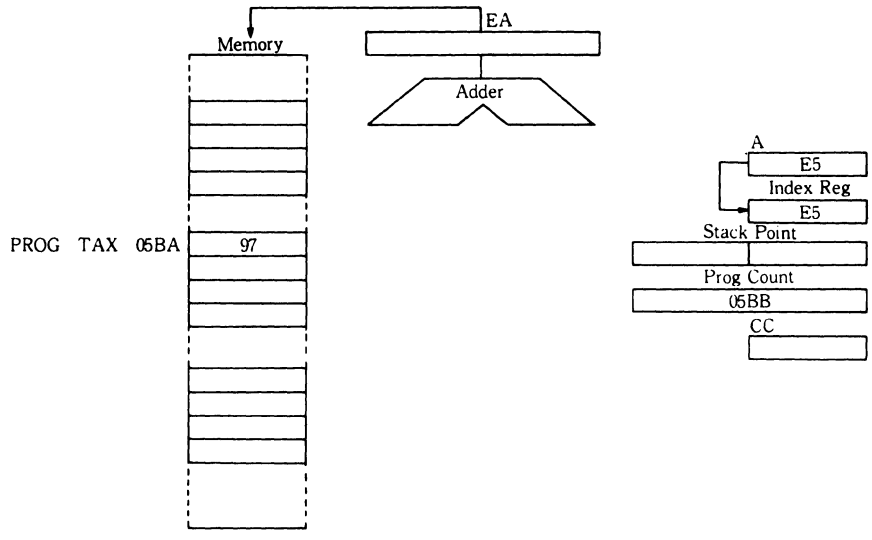


Figure 2-31 Implied Addressing Example

Table 2-2 Register/Memory Instructions

		Addressing Mode																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Operation	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	2	E6	2	4	D6	3	5
Load X from Memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	2	EE	2	4	DE	3	5
Store A in Memory	STA	–	–	–	B7	2	4	C7	3	5	F7	1	3	E7	2	5	D7	3	6
Store X in Memory	STX	–	–	–	BF	2	4	CF	3	5	FF	1	3	EF	2	5	DF	3	6
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	2	EB	2	4	DB	3	5
Add Memory and Carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	2	E9	2	4	D9	3	5
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	2	E0	2	4	D0	3	5
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	2	E2	2	4	D2	3	5
AND Memory to A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	2	E4	2	4	D4	3	5
OR Memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	2	EA	2	4	DA	3	5
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	3	C8	3	4	F8	1	2	E8	2	4	D8	3	5
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	2	E1	2	4	D1	3	5
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	2	E3	2	4	D3	3	5
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	2	E5	2	4	D5	3	5
Jump Unconditional	JMP	–	–	–	BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to Subroutine	JSR	–	–	–	BD	2	4	CD	3	5	FD	1	3	ED	2	4	DD	3	5

Symbols: Op = Operation # = Instruction

Table 2-3 Read/Modify/Write Instructions

		Addressing Mode														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
Operation	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	1	5C	1	1	3C	2	4	7C	1	3	6C	2	5
Decrement	DEC	4A	1	1	5A	1	1	3A	2	4	7A	1	3	6A	2	5
Clear	CLR	4F	1	1	5F	1	1	3F	2	4	7F	1	3	6F	2	5
Complement	COM	43	1	1	53	1	1	33	2	4	73	1	3	63	2	5
Negate (2's Complement)	NEG	40	1	1	50	1	1	30	2	4	70	1	3	60	2	5
Rotate Left Thru Carry	ROL	49	1	1	59	1	1	39	2	4	79	1	3	69	2	5
Rotate Right Thru Carry	ROR	46	1	1	56	1	1	36	2	4	76	1	3	66	2	5
Logical Shift Left	LSL	48	1	1	58	1	1	38	2	4	78	1	3	68	2	5
Logical Shift Right	LSR	44	1	1	54	1	1	34	2	4	74	1	3	64	2	5
Arithmetic Shift Right	ASR	47	1	1	57	1	1	37	2	4	77	1	3	67	2	5
Arithmetic Shift Left	ASL	48	1	1	58	1	1	38	2	4	78	1	3	68	2	5
Test for Negative or Zero	TST	4D	1	1	5D	1	1	3D	2	4	7D	1	3	6D	2	5

Symbols: Op = Operation # = Instruction

Table 2-4 Branch Instructions

Operation	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	3
Branch Never	BRN	21	2	2 or 3 *
Branch IF Higher	BHI	22	2	2 or 3 *
Branch IF Lower or Same	BLS	23	2	2 or 3 *
Branch IF Carry Clear	BCC	24	2	2 or 3 *
(Branch IF Higher or Same)	(BHS)	24	2	2 or 3 *
Branch IF Carry Set	BCS	25	2	2 or 3 *
(Branch IF Lower)	(BLO)	25	2	2 or 3 *
Branch IF Not Equal	BNE	26	2	2 or 3 *
Branch IF Equal	BEQ	27	2	2 or 3 *
Branch IF Half Carry Clear	BHCC	28	2	2 or 3 *
Branch IF Half Carry Set	BHCS	29	2	2 or 3 *
Branch IF Plus	BPL	2A	2	2 or 3 *
Branch IF Minus	BMI	2B	2	2 or 3 *
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	2 or 3 *
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	2 or 3 *
Branch IF Interrupt Line is Low	BIL	2E	2	2 or 3 *
Branch IF Interrupt Line is High	BIH	2F	2	2 or 3 *
Branch to Subroutine	BSR	AD	2	4

Symbol: Op = Operation # = Instruction
* If branched, each instruction will be a 3-cycle instruction.

Table 2-5 Bit Processing Instructions

Operations	Mnemonic	Addressing Mode					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is Set	BRSET n (n = 0.....7)	–	–	–	2 • n	3	4 or 5 *
Branch IF Bit n is Clear	BRCLR n (n = 0.....7)	–	–	–	01 + 2 • n	3	4 or 5 *
Set Bit n	BSET n (n = 0.....7)	10 + 2 • n	2	4	–	–	–
Clear Bit n	BCLR n (n = 0.....7)	11 + 2 • n	2	4	–	–	–

Symbol: Op = Operation # = Instruction
* If Branched, each instruction will be a 5-cycle instruction.

Table 2-6 Control Instructions

Operation	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	1
Transfer X to A	TXA	9F	1	1
Set Carry Bit	SEC	99	1	1
Clear Carry Bit	CLC	98	1	1
Set Interrupt Mask Bit	SEI	9B	1	1
Clear Interrupt Mask Bit	CLI	9A	1	1
Software Interrupt	SWI	83	1	9
Return from Subroutine	RTS	81	1	4
Return from Interrupt	RTI	80	1	7
Reset Stack Pointer	RSP	9C	1	1
No-Operation	NOP	9D	1	1

Symbol: Op = Operation # = Instruction

Table 2-7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			∧	•	∧	∧	∧
ADD		x	x	x		x	x	x			∧	•	∧	∧	∧
AND		x	x	x		x	x	x			•	•	∧	∧	•
ASL	x		x			x	x				•	•	∧	∧	∧
ASR	x		x			x	x				•	•	∧	∧	∧
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	∧	∧	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•

(Continued)

Symbols for condition code:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 2-7 Instruction Set (Continued)

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
BRN					x						●	●	●	●	●
BRCLR										x	●	●	●	●	∧
BRSET										x	●	●	●	●	∧
BSET									x		●	●	●	●	●
BSR					x						●	●	●	●	●
CLC	x										●	●	●	●	0
CLI	x										●	0	●	●	●
CLR	x		x			x	x				●	●	0	1	●
CMP		x	x	x		x	x	x			●	●	∧	∧	∧
COM	x		x			x	x				●	●	∧	∧	1
CPX		x	x	x		x	x	x			●	●	∧	∧	∧
DEC	x		x			x	x				●	●	∧	∧	●
EOR		x	x	x		x	x	x			●	●	∧	∧	●
INC	x		x			x	x				●	●	∧	∧	●
JMP			x	x		x	x	x			●	●	●	●	●
JSR			x	x		x	x	x			●	●	●	●	●
LDA		x	x	x		x	x	x			●	●	∧	∧	●
LDX		x	x	x		x	x	x			●	●	∧	∧	●
LSL	x		x			x	x				●	●	∧	∧	∧
LSR	x		x			x	x				●	●	0	∧	∧
NEG	x		x			x	x				●	●	∧	∧	∧
NOP	x										●	●	●	●	●
ORA		x	x	x		x	x	x			●	●	∧	∧	●
ROL	x		x			x	x				●	●	∧	∧	∧
ROR	x		x			x	x				●	●	∧	∧	∧
RSP	x										●	●	●	●	●
RTI	x										?	?	?	?	?
RTS	x										●	●	●	●	●
SBC		x	x	x		x	x	x			●	●	∧	∧	∧
SEC	x										●	●	●	●	1
SEI	x										●	1	●	●	●
STA			x	x		x	x	x			●	●	∧	∧	●
STX			x	x		x	x	x			●	●	∧	∧	●
SUB		x	x	x		x	x	x			●	●	∧	∧	∧
SWI	x										●	1	●	●	●
TAX	x										●	●	●	●	●
TST	x		x			x	x				●	●	∧	∧	●
TXA	x										●	●	●	●	●

Symbols for condition code:

H Half Carry (From Bit 3)
I Interrupt Mask
N Negative (Sign Bit)
Z Zero
C Carry/Borrow
∧ Test and Set if True, Cleared Otherwise
● Not Affected
? Load CC Register From Stack

Table 2-8 OP Code Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	←HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA (+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(–1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(+1)		JSR		JSR(+1)	D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/4 or 5	2/4	2/2 or 3	2/4	1/1	1/1	2/5	1/3	1/*	1/1	2/2	2/3	3/4	3/5	2/4	1/2	

(NOTES) 1. "–" is an undefined operation code.
2. The figure in the lowest row of each column gives the number of bytes and the cycles needed for the instruction.
The number of cycles for the asterisked (*) mnemonics is a follows:

RTI	7
RTS	4
SWI	9
BSR	4

3. The parenthesized figure must be added to the cycle count of the associated instruction.
4. If the instruction is branched, the cycle count is the larger figure.

3. Executable Instruction

Shown below are the meanings of symbols and abbreviations.

(1) Operation

( ): contents

←: movement direction

+: addition

-: subtraction

∧: AND

∨: OR

⊕: Exclusive OR

$\overline{x}$: NOT

(2) Register symbols in MPU

ACCA: accumulator A

CC: condition codes register

IX: index register, 8 bits

PC: program counter, 12 bits

PCH: upper three bits of program counter

PCL: lower eight bits of program counter

SP: stack pointer, 5 bits

(3) Memory and addressing codes

M: stored address

MH: upper eight bits of stored address

ML: lower eight bits of stored address

M+1: stored address M plus 1

Msp: stored address indicated by stack pointer

Imm: immediate value

Disp: displacement value = M - (IX)

D: displacement value = M - (IX)

DH: displacement value = upper eight bits

DL: displacement value = lower eight bits

Rel: relative value

IMPLIED: implied addressing

RELATIVE: relative addressing

ACCUMULATOR: accumulator addressing

INDEX REG.: index register addressing

IMMEDIATE: immediate addressing

DIRECT: direct addressing

EXTENDED: extended addressing

INDEXED 0 BYTE OFFSET: indexed addressing 0 byte offset

INDEXED 1 BYTE OFFSET: indexed addressing 1 byte offset

INDEXED 2 BYTE OFFSET: indexed addressing 2 byte offset

EA: effective address

(4) Contents of bits 0 through 4 of condition codes register

C: carry - borrow bit 0

Z: zero bit 1

N: negative bit 2

I: interrupt mask bit 3

H: half carry from bit 3 to bit 4 bit 4

(5) Status of each bit before execution of instruction

An: bit n of ACCA (n = 7, 6, 5, ..., 0)

Mn: bit n of M (n = 7, 6, 5, ..., 0)

Xn: bit n of IX (n= 7, 6, 5, ..., 0)

(6) Status of each bit on result after execution of instruction

Rn: bit n of result (n = 7, 6, 5, ..., 0)

(7) Symbols on instruction's format

P: each addressing mode on Immediate, Direct, Extended and index of 0, 1 and 2 byte offset

Q: each addressing mode on Direct and index of 0 and 1 byte offset

A: accumulator addressing mode

X: index register addressing mode

DR: direct addressing mode

dd: relative operand (8 bits)

n: bit n of memory (n = 7, 6, 5, ..., 0)

(8) Status of HD63L05'S interrupt pin

INT: status of interrupt pin (high, low)

ADC (ADd with Carry)

Format

ADC P

Operation

ACCA <- (ACCA) + (M) + (C)

Condition Codes

H: Set if there was a carry from bit 3, cleared otherwise.
I: Not affected.
N: Set if the most significant bit of the result is set; cleared otherwise.
Z: Set if the result is 0; otherwise cleared.
C: Set if there was a carry from the most significant bit of the result; cleared otherwise.

Description

Adds the contents of the carry bit C to the sum of the contents of ACCA and M, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	ADC	#Imm	A9	Imm		2	2
DIRECT	ADC	M	B9	M		2	3
EXTENDED	ADC	M	C9	MH	ML	3	4
INDEXED 0 BYTE OFFSET	ADC	,X	F9			1	2
INDEXED 1 BYTE OFFSET	ADC	Disp,X	E9	D		2	4
INDEXED 2 BYTE OFFSET	ADC	Disp,X	D9	DH	DL	3	5

Example

```
LDA   VAL2      (EXVAL5,EXVAL6)+(VAL1,VAL2)
ADD   EXVAL6    *        =(EXVAL5,EXVAL6)
STA   EXVAL6    *
LDA   VAL1      *
ADC   EXVAL5    ***
STA   EXVAL5    *
```

ADD (ADD without carry)

Format

ADD P

Operation

ACCA <- (ACCA) + (M)

Condition Codes

H: Set if there was a carry from bit 3; cleared otherwise.
I: Not affected.
N: Set if the significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if there was a carry from the most significant bit of the result; otherwise cleared.

Description

Adds the contents of ACCA and the contents of M, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	ADD	#Imm	AB	Imm		2	2
DIRECT	ADD	M	BB	M		2	3
EXTENDED	ADD	M	CB	MH	ML	3	4
INDEXED 0 BYTE OFFSET	ADD	,X	FB			1	2
INDEXED 1 BYTE OFFSET	ADD	Disp,X	EB	D		2	4
INDEXED 2 BYTE OFFSET	ADD	Disp,X	DB	DH	DL	3	5

Example

```
LDA     VAL1        (VAL1)+(WORK)=(RESULT)
ADD     WORK        *
STA     RESULT      *
```

AND (logical AND)

Format

AND P

Operation

ACCA <- (ACCA) ∧ (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Not affected.

Description

Performs logical AND between the contents of ACCA and the contents of M, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	AND	#Imm	A4	Imm		2	2
DIRECT	AND	M	B4	M		2	3
EXTENDED	AND	M	C4	MH	ML	3	4
INDEXED 0 BYTE OFFSET	AND	,X	F4			1	2
INDEXED 1 BYTE OFFSET	AND	Disp,X	E4	D		2	4
INDEXED 2 BYTE OFFSET	AND	Disp,X	D4	DH	DL	3	5

Example

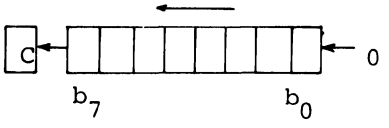
```
LDA   0,X     ERASE UPPER 4 BITS
AND   #$0F    *
STA   0,X     (RESTORE)
INC   X       *
BRA   LOOP    *
```

ASL (Arithmetic Shift Left)

Format

ASL Q
ASL A
ASL X

Operation



Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if, before a shift, the most significant bit is 1, otherwise cleared.

Description

Shifts the contents of ACCA, IX or M one place to the left. The bit 0 is loaded with a zero. The carry bit C is loaded with the bit 7 of ACCA, IX or M.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	ASL	A	48			1	1
INDEX REG.	ASL	X	58			1	1
DIRECT	ASL	M	38	M		2	4
INDEXED 0 BYTE OFFSET	ASL	,X	78			1	3
INDEXED 1 BYTE OFFSET	ASL	Disp,X	68	D		2	5

Example

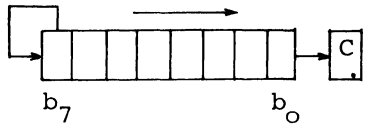
```
        LDA   WORK
CHECK   ASL   A       BRANCH FOLLOWING BIT
        BCS   BITON   *       7-6-5-4-3-2-1-0
BITOFF  EQU   *
        LDX   #100
```

ASR (Arithmetic Shift Right)

Format

ASR Q
ASR A
ASR X

Operation



Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if, before a shift, the least significant bit is 1; otherwise cleared.

Description

Shifts the contents of ACCA, IX or M one place to the right. The bit 7 is held constant. The bit 0 is loaded into the carry bit C.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	ASR	A	47			1	1
INDEX REG.	ASR	X	57			1	1
DIRECT	ASR	M	37	M		2	4
INDEXED 0 BYTE OFFSET	ASR	,X	77			1	3
INDEXED 1 BYTE OFFSET	ASR	Disp,X	67	D		2	5

Example

```
ASR WORK     BRANCH OPTION (KEEPING BIT7)
BCS OPT0
ASR WORK
BCS OPT1
ASR WORK
BCS OPT2
*
```

BCC (Branch if Carry Clear)

Format

BCC dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C)=0

Description

Tests the state of the C bit and causes a branch if C is 0.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BCC	Rel	24	Rel		2	2 or 3

Example

```
        LDA    VAL2
        ADD    EXVAL6
        BCC    NORMAL   KETA AGARI NASHI
*
        INC    X        KETA AGARI
```

BCLR (Bit CLear bit n)

Format

BCLR n,DR

Condition Codes

Not affected.

Operation

Mn <- 0

Description

Clears the bit n (n = 0 through 7) of M. The other bits are unaffected.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
DIRECT	BCLR	0,M	11	M		2	4
DIRECT	BCLR	1,M	13	M		2	4
DIRECT	BCLR	2,M	15	M		2	4
DIRECT	BCLR	3,M	17	M		2	4
DIRECT	BCLR	4,M	19	M		2	4
DIRECT	BCLR	5,M	1B	M		2	4
DIRECT	BCLR	6,M	1D	M		2	4
DIRECT	BCLR	7,M	1F	M		2	4

Example

```
LDA    CNTRL     ** MAKE CONTROL CODE **
AND    #$F0      *
ORA    WORK      *
STA    CNTRL     *
BCLR   0,CNTRL   CLEAR BIT 0,6,7 ABSOLUTELY
BCLR   6,CNTRL
BCLR   7,CNTRL
```

BCS (Branch if Carry Set)

Format

BCS dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C)=1

Description

Tests the state of the C bit and causes a branch if C is 1.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BCS	Rel	25	Rel		2	2 or 3

Example

```
        LDA     VAL1
        ADD     EXVAL6
        BCS     ABNML       KETA AGARI
*
        STA     EXVAL6      KETA AGARI NASHI
```

BEQ (Branch of EQual)

Format

BEQ dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (Z)=1

Description

Tests the state of the Z bit and causes a branch if Z is 1.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BEQ	Rel	27	Rel		2	2 or 3

Example

```
LDA    WORK
BEQ    AAAA      WORK = 0
CMP    RESULT
BEQ    BBBB      WORK = RESULT
```

BHCC (Branch if Half Carry Clear)

Format

BHCC dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (H)=0

Description

Tests the state of the H bit and causes a branch if H is 0.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BHCC	Rel	28	Rel		2	2 or 3

Example

```
        CMP    #$9
        BLS    DAALOW   $99 ---> INPUT
DAAH6   LDX    #$60     HIGH NYBLE NEEDS CORRECTION
*
DAALOW  BHCC   DAAL9
        TXA
```

BHCS (Branch if Half Carry Set)

Format

BHCS dd

Condition Codes

Not affected

Operation

PC <- (PC)+0002+Rel if (H)=1

Description

Tests the state of the H bit and causes a brach if H is 1.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BHCS	Rel	29	Rel		2	2 or 3

Example

```
        CMP    #$9
        BLS    DAALW1   $99 ---> INPUT
DAAH7   LDX    #$60     HIGH NYBLE NEEDS CORRECTION
*
DAALW1  BHCS   DAAL6
        NAD    #$F
```

BHI (Branch if HIgher)

Format

BHI dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C V Z)=0
i.e. if (ACCA)>(M)
(unsigned binary numbers)

Description

Causes a brach if C is 0 and Z is 0.
If the BHI instruction is executed immediately after execution of either of the instructions CMP or SUB, the branch will occur if and only the unsigned binary number represented by the minuend (i.e. ACCA) was greater than the unsigned binary number represented by the subtrahend (i.e. M).

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BHI	Rel	22	Rel		2	2 or 3

Example

```
        LDA     VAL1
        CMP     VAL2
        BHI     ZIP25   VAL1 > VAL2  (IGNORE SIGN BIT)
*
        STA     WORK    VAL1 --> WORK (LOWER OR SAME)
```

BHS (Branch if Higher or Same)

Format

BHS dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C)=0

Description

Following an unsigned compare or subtract, BHS will cause a branch if the register was higher than or the same as the location in memory.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BHS	Rel	24	Rel		2	2 or 3

Example

```
        LDA    VAL1
        CMP    VAL2
        BHS    ZIP26     VAL1 >= VAL2  IGNORE SIGN BIT
*
        STA    WORK      VAL1 ---> WORK (LOWER)
```

BIH (Branch if Interrupt line is High)

Format

BIH dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel
if INT=1 (high)

Description

Tests the state of the external interrupt pin (INT) and causes a branch if it is high.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BIH	Rel	2F	Rel		2	2 or 3

Example

```
        BIH   INTH0   INT LINE CHECK
INTL1   LDA   #$28    OUTPUT DATA = $28
        BRA   NEXT2
INTH0   LDA   #$FF    OUTPUT DATA = $FF
NEXT2   STA   PIA     OUTPUT
```

BIL (Branch if Interrupt line is Low)

Format

BIL dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel
if INT=0 (low)

Description

Tests the state of the external interrupt pin and causes a branch if it is low.

If branched, this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BIL	Rel	2E	Rel		2	2 or 3

Example

```
        BIL    INTL2    INT LINE CHECK
INTH3   LDA    #$45     OUTPUT DATA = $45
        BRA    NEXT4
INTL2   LDA    #$0      OUTPUT DATA = $00
NEXT4   STA    PIA      OUTPUT
```

BIT (BIt Test)

Format

BIT P

Operation

(ACCA) Λ (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result of the AND is 1; otherwise cleared.
Z: Set if all the bits of the result of the AND are 0; otherwise cleared.
C: Not affected.

Description

Performs the logical AND operation of the contents of ACCA and the contents of M, and modifies the condition codes accordingly. The contents of ACCA and M are held

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	BIT	#Imm	A5	Imm		2	2
DIRECT	BIT	M	B5	M		2	3
EXTENDED	BIT	M	C5	MH	ML	3	4
INDEXED 0 BYTE OFFSET	BIT	,X	F5			1	2
INDEXED 1 BYTE OFFSET	BIT	Disp,X	E5	D		2	4
INDEXED 2 BYTE OFFSET	BIT	Disp,X	D5	DH	DL	3	5

Example

```
EVBIT   LDA     VAL1
        BIT     #$F8
        BEQ     OK          0 <= BIT ASSIGN (VAL1) <= 7
*
NG      LDA     #227        SET ERROR NUMBER
        JMP     ERROR
```

BLO (Branch if LOwer)

Format

BLO dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C)=1

Description

Following a compare, BLO will branch if the register was lower than the memory location.
Equivalent to the BCS executable instruction.

If branched,this instrucion will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BLO	Rel	25	Rel		2	2 or 3

Example

```
        LDA    VAL1
        CMP    VAL2
        BLO    ZIP27     VAL1 < VAL2 (IGNORE SIGN BIT)
*
        STA    WORK      VAL1 --> WORK  HIGHER OR SAME
```

BLS (Branch if Lower or Same)

Format

BLS dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (C V Z)=1
i.e. if (ACCA)≤(M)

Description

Causes a branch if C is 1 or Z is 1.
If the BLS instruction is executed immediately after execution of either of the instructions CMP or SUB, the branch will occur if and only the unsigned binary number represented by the minuend (i.e. ACCA) was less than or equal to the unsigned binary number represented by the subtrahend (i.e. M).

If branched, this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BLS	Rel	23	Rel		2	2 or 3

Example

```
        LDA     VAL1
        CMP     VAL2
        BLS     ZIP28     VAL1<= VAL2 IGNORE SIGN BIT
*
        STA     WORK      VAL1 ---> WORK (HIGHER)
```

BMC (Branch if interrupt Mask is Clear)

Format

BMC dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (I)=0

Description

Tests the state of the I bit and causes a branch if I is 0.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BMC	Rel	2C	Rel		2	2 or 3

Example

```
        BMC    MSKOFF   INTMSK OFF?
        BIL    MSKOFF   INT LINE LOW?
        LDA    PIA      READ DATA
        STA    WORK
MSKOFF  RTS
```

BMI (Branch if MInus)

Format

BMI dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (N)=1

Description

Tests the state of the N bit, and causes a branch if N is 1.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
RELATIVE	BMI	Rel	2B	Rel		2	2 or 3

Example

```
        LDA    VAL1
        BMI    ZIP29     VAL1 < 0
*
        STA    WORK      VAL1 ---> WORK (PLUS)
```

BMS (Branch if interrupt Mask is Set)

Format

BMS dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (I)=1

Description

Tests the state of the I bit and causes a branch if I is 1.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BMS	Rel	2D	Rel		2	2 or 3

Example

```
        BMS     MSKON1   INTMSK ON?
MSKOF1  RTS              NO
MSKON1  BIL     MSKOF1   INT LINE LOW?
        LDA     PIA      DATA
        STA     WORK
        RTS
```

BNE (Branch if Not Equal)

Format

BNE dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (Z)=0

Description

Tests the state of the Z bit and causes a branch if Z is 0.
Following a compare or subtract instruction, BNE will cause a branch if the arguments were different.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
RELATIVE	BNE	Rel	26	Rel		2	2 or 3

Example

```
LDA    WORK
BNE    CCCC    WORK NOT = 0
CMP    RESULT
BNE    DDDD    WORK NOT = RESULT
```

BPL (Branch if PLus)

Format

BPL dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel if (N)=0

Description

Tests the state of the N bit, and causes a branch if N is 0.

If branched,this instruction will be a 3-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BPL	Rel	2A	Rel		2	2 or 3

Example

```
        LDA     VAL1
        BPL     ZIP31     VAL >= 0
*
        STA     WORK      VAL1 ---> WORK (MINUS)
```

BRA (BRanch always)

Format

BRA dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002+Rel

Description

Causes an unconditional branch to the address given by the above expression.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
RELATIVE	BRA	Rel	20	Rel		2	3

Example

```
        LDA    EXVAL5
        STA    RESULT
        BRA    END01    BRANCH TO END01 ALWAYS
*
CHECK8 EQU    *
```

BRCLR (BRanch if bit n is CLeaR)

Format

BRCLR n,DR,dd

Operation

PC ← (PC) + 0003 + Rel if (Mn) = 0

Condition Codes

H: Not affected.
I: Not affected.
N: Not affected.
Z: Not affected.
C: Set if (Mn) = 1; otherwise cleared.

Description

Tests the bit n (n = 0 through 7) of M and causes a branch if the contents of Mn are 0.

If branched, each instruction will be a 5-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BRCLR	0,M,Rel	01	M	Rel	3	4 or 5
RELATIVE	BRCLR	1,M,Rel	03	M	Rel	3	4 or 5
RELATIVE	BRCLR	2,M,Rel	05	M	Rel	3	4 or 5
RELATIVE	BRCLR	3,M,Rel	07	M	Rel	3	4 or 5
RELATIVE	BRCLR	4,M,Rel	09	M	Rel	3	4 or 5
RELATIVE	BRCLR	5,M,Rel	0B	M	Rel	3	4 or 5
RELATIVE	BRCLR	6,M,Rel	0D	M	Rel	3	4 or 5
RELATIVE	BRCLR	7,M,Rel	0F	M	Rel	3	4 or 5

Example

```
LDA    CNTRL     ** SET CONTROL CODE **
AND    #$0F
ORA    WORK
STA    CNTRL
*           ** ACTION **
BRCLR  4,CNTRL,ENGINE
BRCLR  7,CNTRL,GASCHK
```

Unconditional Branch
BRN

BRN (BRanch Never)

Format

BRN dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002

Description

BRN is included here to demonstrate the nature of branches on the 63L05. Each branch is matched with an inverse that varies only in the least significant bit of the opcode. BRN is the inverse of BRA. This instruction may have some use during program debugging.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
RELATIVE	BRN	Rel	21	Rel		2	2

Example

```
        STX   4,X
*
        BRN   *          ** DELAY **
        BRN   *
        BRN   *
        BRN   *
```

BRSET (BRanch if bit n is SET)

Format

BRSET n,DR,dd

Operation

PC <-- (PC)+0003+Rel if (Mn)=1

Condition Codes

H: Not affected.
I: Not affected.
N: Not affected.
Z: Not affected.
C: Set if (Mn)=1; otherwise cleared.

Description

Tests the bit n (n = 0 through 7) of M, and causes a branch if the contents of Mn are 1.

If branched,each instruction will be a 5-cycle instruction.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BRSET	0,M,Rel	00	M	Rel	3	4 or 5
RELATIVE	BRSET	1,M,Rel	02	M	Rel	3	4 or 5
RELATIVE	BRSET	2,M,Rel	04	M	Rel	3	4 or 5
RELATIVE	BRSET	3,M,Rel	06	M	Rel	3	4 or 5
RELATIVE	BRSET	4,M,Rel	08	M	Rel	3	4 or 5
RELATIVE	BRSET	5,M,Rel	0A	M	Rel	3	4 or 5
RELATIVE	BRSET	6,M,Rel	0C	M	Rel	3	4 or 5
RELATIVE	BRSET	7,M,Rel	0E	M	Rel	3	4 or 5

Example

```
        LDA    CNTRL     ** SET CONTROL CODE **
        AND    #$8E
        ORA    WORK
        STA    CNTRL
*                        ** ACTION **
PROC1   BRSET  0,CNTRL,OIL
PROC2   BRSET  7,CNTRL,GAS
```

BSET (Bit SET bin n)

Format

BSET n,DR

Condition Codes

Not affected.

Operation

Mn <- 1

Description

Sets the bit n (n = 0 through 7) of M. All other bits are unaffected.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
DIRECT	BSET	0,M	10	M		2	4
DIRECT	BSET	1,M	12	M		2	4
DIRECT	BSET	2,M	14	M		2	4
DIRECT	BSET	3,M	16	M		2	4
DIRECT	BSET	4,M	18	M		2	4
DIRECT	BSET	5,M	1A	M		2	4
DIRECT	BSET	6,M	1C	M		2	4
DIRECT	BSET	7,M	1E	M		2	4

Example

```
        LDA    RESULT
        BPL    PLUS
*                      (MINUS)
        BSET   2,CNTRL
        BSET   3,WORK
PLUS    EQU    *
        LDA    VAL2
```

BSR (Branch to SubRoutine)

Format

BSR dd

Condition Codes

Not affected.

Operation

PC <- (PC)+0002
Msp <- (PCL), SP <- (SP)-0001
Msp <- (PCH), SP <- (SP)-0001
PC <- (PC)+Rel

Description

The program counter is incremented by 2. The less significant byte of the contents of the program counter is pushed onto the stack. The stack pointer is then decremented by one. The more significant byte of the contents of the program counter is then pushed onto the stack. Unused bits in the Program Counter high byte are stored as 1's on the stack. The stack pointer is again decremented by one. A branch then occurs to the address specified by the program counter.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
RELATIVE	BSR	Rel	AD	Rel		2	4

Example

```
        LDA     #$3B    ACCA = INTERFACE(0011 1011)
        BSR     HAND
*
        LDA     #$1E    ACCA = INTERFACE(0001 1110)
        BSR     FING
```

CLC (CLear Carry)

Format

CLC

Condition Codes

H: Not affected.
I: Not affected.
N: Not affected.
Z: Not affected.
C: Cleared.

Operation

C <- 0

Description

Clears the carry bit C in the condition code register.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	CLC		98			1	1

Example

```
        BNE     CHK83
        STA     RESULT
        CLC             RETURN CODE SET 'OK'
        RTS
*
*
```

CLI (CLear Interrupt mask)

Format

CLI

Operation

I <- 0

Condition Codes

H: Not affected.
I: Cleared.
N: Not affected.
Z: Not affected.
C: Not affected.

Description

Clears the interrupt mask bit in the processor condition code register. This enables the microprocessor to service interrupts. Interrupts that were pending while the I bit was set will now begin to have effect.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	CLI		9A			1	1

Example

```
SEI             INTERRUPT DISABLE
RSP             RESET STACK POINTER
JSR    SYSINZ   SYSTEM INITIALIZE
CLI             INTERRUPT ENABLE
```

CLR (CLeaR)

Format

```
CLR Q
CLR A
CLR X
```

Operation

IX <- 0
or
ACCA <- 0
or
M <- 0

Condition Codes

H: Not affected.
I: Not affected.
N: Cleared.
Z: Set.
C: Not affected.

Description

The contents of IX, ACCA or M are replaced with zeroes.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	CLR A		4F			1	1
INDEX REG.	CLR X		5F			1	1
DIRECT	CLR	M	3F	M		2	4
INDEXED 0 BYTE OFFSET	CLR	,X	7F			1	3
INDEXED 1 BYTE OFFSET	CLR	Disp,X	6F	D		2	5

Example

```
*                    ** INITIALIZE **
          CLR    PNTR
          CLR    PNTR+1
          CLR    0,X
          CLR    A
```

CMP (CoMPare)

Format

CMP P

Operation

(ACCA) - (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result of the subtraction is 1; otherwise cleared.
Z: Set if the result of the subtraction is 0; otherwise cleared.
C: Set if the absolute value of memory is greater than the abosolute value of the accumulator; otherwise cleared.

Description

Compares the contents of ACCA and the contents of M and sets the condition codes which may then be used for controlling the conditional branches. Both operands are unaffected.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	CMP	#Imm	A1	Imm		2	2
DIRECT	CMP	M	B1	M		2	3
EXTENDED	CMP	M	C1	MH	ML	3	4
INDEXED 0 BYTE OFFSET	CMP	,X	F1			1	2
INDEXED 1 BYTE OFFSET	CMP	Disp,X	E1	D		2	4
INDEXED 2 BYTE OFFSET	CMP	Disp,X	D1	DH	DL	3	5

Example

```
        LDA     PNTR,X
*
        CMP     #'A
        BEQ     SECTA     ACCA = 'A'
        CMP     #'B
        BEQ     SECTB     ACCA = 'B'
        BRA     INPUT
```

COM (COMplement)

Format

```
COM Q
COM A
COM X
```

Operation

IX <- $\overline{(IX)}$ =$FF-(IX)
or
ACCA <- $\overline{(ACCA)}$ = $FF-(ACCA)
or
M <- $\overline{(M)}$ = $FF-(M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set

Description

Replaces the contents of ACCA, IX or M with its one's complement.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
ACCUMULATOR	COM	A	43			1	1
INDEX REG.	COM	X	53			1	1
DIRECT	COM	M	33	M		2	4
INDEXED 0 BYTE OFFSET	COM	,X	73			1	3
INDEXED 1 BYTE OFFSET	COM	Disp,X	63	D		2	5

Example

```
*
SUBIN   EQU     *
        INC     X
        LDA     PNTR,X
        COM     A       MODIFY DATA (REVERSE)
        RTS
*
```

CPX (ComPare indeX register)

Format

CPX P

Operation

(IX)-(M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if the absolute value of the contents of the memory is greater than the absolute value of the contents of IX; otherwise cleared.

Description

Compares the contents of IX with those of the memory. The condition code can be collated by means of the next conditional branch instruction. Both operands are unaffected.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	CPX	#Imm	A3	Imm		2	2
DIRECT	CPX	M	B3	M		2	3
EXTENDED	CPX	M	C3	MH	ML	3	4
INDEXED 0 BYTE OFFSET	CPX	,X	F3			1	2
INDEXED 1 BYTE OFFSET	CPX	Disp,X	E3	D		2	4
INDEXED 2 BYTE OFFSET	CPX	Disp,X	D3	DH	DL	3	5

Example

```
LDA     #$CC    ACCA = INTERFACE TO CR OR LF
LDX     PNTR
CPX     #$0D
BEQ     CR      CARRIAGE RETURN
CPX     #$0A
BEQ     LF      LINE FEED
```

DEC (DECrement)

Format

```
DEC Q
DEC A
DEC X
```

Operation

IX <- (IX)-01
or
ACCA <- (ACCA)-01
or
M <- (M)-01

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Not affected.

Description

Subtracts one from the contents of ACCA, IX or M.
N and Z bits are set and reset according to the result of this operation.
The C bit is not affected by this operation.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	DEC	A	4A			1	1
INDEX REG.	DEC	X	5A			1	1
DIRECT	DEC	M	3A	M		2	4
INDEXED 0 BYTE OFFSET	DEC	,X	7A			1	3
INDEXED 1 BYTE OFFSET	DEC	Disp,X	6A	D		2	5

Example

```
*              ** MOVE **
LOOP23 DEC     A
       BMI     NEXT
       LDX     0,X      *
       STX     $100,X   *
       INC     X        *
       BRA     LOOP23
*
NEXT   EQU     *
```

Logical Operation
EOR

EOR (Exclusive OR)

Format

EOR P

Operation

ACCA <- (ACCA) ⊕ (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Not affected.

Description

Performs the logical EXCLUSIVE OR between the contents of ACCA and those of M, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
IMMEDIATE	EOR	#Imm	A8	Imm		2	2
DIRECT	EOR	M	B8	M		2	3
EXTENDED	EOR	M	C8	MH	ML	3	4
INDEXED 0 BYTE OFFSET	EOR	,X	F8			1	2
INDEXED 1 BYTE OFFSET	EOR	Disp,X	E8	D		2	4
INDEXED 2 BYTE OFFSET	EOR	Disp,X	D8	DH	DL	3	5

Example

```
*                    ** ARRANGE CONTROL CODE **
        LDA    CNTRL     XXXX XXXX
        EOR    #$99      1001 1001
        STA    CNTRL
        BRA    ACT01
```

INC (INCrement)

Format

```
INC Q
INC A
INC X
```

Operation

```
IX <- (IX)+01
   or
ACCA <- (ACCA)+01
   or
M <- (M)+01
```

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Not affected.

Description

Adds one to the contents of ACCA, IX or M.
N and Z bits are set or reset according to the result of this operation.
The C bit is not affected by this operation.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	INC	A	4C			1	1
INDEX REG.	INC	X	5C			1	1
DIRECT	INC	M	3C	M		2	4
INDEXED 0 BYTE OFFSET	INC	,X	7C			1	3
INDEXED 1 BYTE OFFSET	INC	Disp,X	6C	D		2	5
INDEXED 2 BYTE OFFSET							

Example

```
LOOP3   INC     A           *
        CMP     #100
        BHI     EXIT        CHECK COUNTER (100 TIMES)
        LDX     0,X
        STX     $300,X      MOVE
        INC     X           *
```

JMP (JuMP)

Format

JMP P

Condition Codes

Not affected.

Operation

PC <- EA

Description

A jump occurs to the instruction stored at the effective address. The effective address is obtained according to the rules for EXTended, DIRect or INDexed addressing.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
DIRECT	JMP	M	BC	M		2	2
EXTENDED	JMP	M	CC	MH	ML	3	3
INDEXED 0 BYTE OFFSET	JMP	,X	FC			1	1
INDEXED 1 BYTE OFFSET	JMP	Disp,X	EC	D		2	3
INDEXED 2 BYTE OFFSET	JMP	Disp,X	DC	DH	DL	3	4

Example

```
LDA     VAL1
STA     EXVAL5
LDA     VAL2
STA     EXVAL6
JMP     END90       DO TO END-ROUTINE
```

JSR (Jump to SubRoutine)

Format

JSR P

Condition Codes

Not affected.

Operation

PC <- (PC)+n [Note)]
Msp <- (PCL), SP <- (SP)-0001
Msp <- (PCH), SP <- (SP)-0001
PC <- EA

Description

The program counter is incremented by n [Note)] depending on the addressing mode, and is then pushed onto the 2-byte stack. And the stack point is updated. A jump occurs to the instruction stored at the effective address. The effective address is obtained according to the rules for EXTended, DIRect or INDexed addressing.

Note) n is equal to 1, 2 or 3, depending on the number of bytes in the instruction code. Refer to the addressing code and the number of MPU cycles shown below.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
DIRECT	JSR	M	BD	M		2	4
EXTENDED	JSR	M	CD	MH	ML	3	5
INDEXED 0 BYTE OFFSET	JSR	,X	FD			1	3
INDEXED 1 BYTE OFFSET	JSR	Disp,X	ED	D		2	4
INDEXED 2 BYTE OFFSET	JSR	Disp,X	DD	DH	DL	3	5

Example

```
*                    ** MAIN ROUTINE **
START   EQU     *
        JSR     INTRTN      INITIALIZE
        JSR     KBRTN       INPUT FROM KEY-BOARD
        JSR     ANARTN      ANALYSE
        JSR     PRCRTN      PROCESS
        JMP     ENDRTN      END
```

LDA (LoaD Accumulator)

Format

LDA P

Operation

ACCA <- (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Not affected.

Description

Loads the contents of memory into the accumulator.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	LDA	#Imm	A6	Imm		2	2
DIRECT	LDA	M	B6	M		2	3
EXTENDED	LDA	M	C6	MH	ML	3	4
INDEXED 0 BYTE OFFSET	LDA	,X	F6			1	2
INDEXED 1 BYTE OFFSET	LDA	Disp,X	E6	D		2	4
INDEXED 2 BYTE OFFSET	LDA	Disp,X	D6	DH	DL	3	5

Example

```
LDA    VAL1
STA    WORK
LDA    0,X
STA    RESULT
LDA    #$FF
```

LDX (LoaD indeX register)

Format

LDX P

Operation

IX <- (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of IX is 1; otherwise cleared.
Z: Set if all the bits of IX of the result are 0; otherwise cleared.
C: Not affected.

Description

Loads the contents of memory into IX. The condition code is set according to data.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMMEDIATE	LDX	# Imm	AE	Imm		2	2
DIRECT	LDX	M	BE	M		2	3
EXTENDED	LDX	M	CE	MH	ML	3	4
INDEXED 0 BYTE OFFSET	LDX	,X	FE			1	2
INDEXED 1 BYTE OFFSET	LDX	Disp,X	EE	D		2	4
INDEXED 2 BYTE OFFSET	LDX	Disp,X	DE	DH	DL	3	5

Example

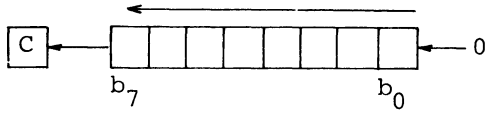
```
LDX    VAL1
STX    WORK
LDX    0,X
STX    RESULT
LDX    #$FF
```

LSL (Logical Shift Left)

Format

LSL Q
LSL A
LSL X

Operation



Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if, before the execution of an instruction, the most significant bit of ACCA, IX or M was 1; otherwise cleared.

Description

Shifts the contents of ACCA, IX or M one place to the left. The bit 0 is loaded with a zero. The carry bit C is loaded with the most significant bit of ACCA, IX or M.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	LSL	A	48			1	1
INDEX REG.	LSL	X	58			1	1
DIRECT	LSL	M	38	M		2	4
INDEXED 0 BYTE OFFSET	LSL	,X	78			1	3
INDEXED 1 BYTE OFFSET	LSL	Disp,X	68	D		2	5

Example

```
LSL    WORK    ** MULTIPLY   X 8 **
LSL    WORK
LSL    WORK
```

LSR (Logical Shift Right)

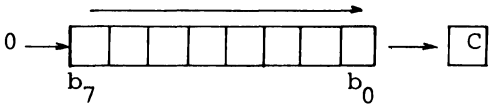
Format

LSR Q
LSR A
LSR X

Condition Codes

H: Not affected.
I: Not affected.
N: Cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if, before execution of an instruction, the least significant bit of ACCA, IX or M was 1; otherwise cleared.

Operation



Description

Shifts the contents of ACCA, IX or M one place to the right. The bit 7 is loaded with a zero. The carry bit C is loaded with the least significant bit of ACCA, IX or M.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
ACCUMULATOR	LSR	A	44			1	1
INDEX REG.	LSR	X	54			1	1
DIRECT	LSR	M	34	M		2	4
INDEXED 0 BYTE OFFSET	LSR	,X	74			1	3
INDEXED 1 BYTE OFFSET	LSR	Disp,X	64	D		2	5

Example

```
LSR     WORK    ** DIVIDE / 16 **
LSR     WORK
LSR     WORK
LSR     WORK
```

NEG (NEGate)

Format

```
NEG Q
NEG A
NEG X
```

Operation

IX <- (IX)=00-(IX)
or
ACCA <- (ACCA)=00-(ACCA)
or
M <- (M)=00-(M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
V: Set if there would be a borrow; otherwise cleared. Set if the contents of ACCA, IX or M are other than 0.

Description

Replaces the contents of ACCA, IX or M with its two's complement. Note that $80 (-128) is left unchanged.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
ACCUMULATOR	NEG	A	40			1	1
INDEX REG.	NEG	X	50			1	1
DIRECT	NEG	M	30	M		2	4
INDEXED 0 BYTE OFFSET	NEG	,X	70			1	3
INDEXED 1 BYTE OFFSET	NEG	Disp,X	60	D		2	5

Example

```
*          CHECK RANGE (RELATIVE ADDRESSING)
           CMP   #129     CHECK RANGE
           BCC   BERROR   *     BRANCH ERROR
           NEG   A        OFFSET
           BRA   SET
*
```

NOP (No OPeration)

Format

NOP

Condition Codes

Not affected.

Operation

Description

This is a single-byte instruction which causes only the program counter to be incremented. No other registers are changed.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
IMPLIED	NOP		9D			1	1

Example

```
NOP          ** DELAY **
NOP
NOP
NOP
NOP
NOP
```

Logical Operation
ORA

ORA (inclusive OR)

Format

ORA

Operation

ACCA <- (ACCA) v (M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if all the bits of the result are 0; otherwise cleared.

Description

Performs logical OR between the contents of ACCA and those of M, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
IMMEDIATE	ORA	# Imm	AA	Imm		2	2
DIRECT	ORA	M	BA	M		2	3
EXTENDED	ORA	M	CA	MH	ML	3	4
INDEXED 0 BYTE OFFSET	ORA	,X	FA			1	2
INDEXED 1 BYTE OFFSET	ORA	Disp,X	EA	D		2	4
INDEXED 2 BYTE OFFSET	ORA	Disp,X	DA	DH	DL	3	5

Example

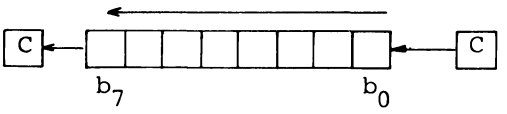
```
        BCS     SKIP
*
ADCN    EQU     *           ** ADDITION CONTROL BIT **
        LDA     #$14        0001 0100
        ORA     CNTRL
        STA     CNTRL
SKIP    EQU     *
```

ROL (ROtate Left)

Format

```
ROL Q
ROL A
ROL X
```

Operation



Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set of all the bits of the result are 0; otherwise cleared.
C: Set if, before the execution of an instruction, the most significant bit of ACCA, IX or M was 1; otherwise cleared.

Description

Shifts the contents of ACCA, IX or M one place to the left. The bit 0 is loaded with the carry bit C, while the carry bit C is loaded with the most significant bit of ACCA, IX or M.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	ROL	A	49			1	1
INDEX REG.	ROL	X	59			1	1
DIRECT	ROL	M	39	M		2	4
INDEXED 0 BYTE OFFSET	ROL	,X	79			1	3
INDEXED 1 BYTE OFFSET	ROL	Disp,X	69	D		2	5

Example

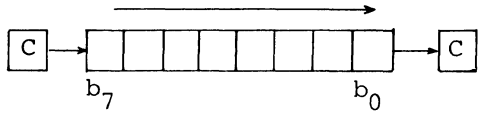
```
*               ** REPEAT ACTION FOLLOWING CNTRL **
        CLC
REPEAT  EQU     *
        ROL     CNTRL
        BCS     ACTION      ACTION & REPEAT OR ESCAPE
        NOP                 *
        NOP                 ** DELAY
        NOP                 *
        BRA     REPEAT
```

ROR (ROtate Right)

Format

ROR Q
ROR A
ROR X

Operation



Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if all the bits of the result are 0; otherwise cleared.
C: Set if, before the execution of an instruction, the least significant bit of ACCA, IX or M was 1; otherwise cleared.

Description

Shifts the contents of ACCA, IX or M one place to the right. The bit 7 is loaded with the carry bit C, while the bit 0 is loaded with the carry bit C.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
ACCUMULATOR	ROR	A	46			1	1
INDEX REG.	ROR	X	56			1	1
DIRECT	ROR	M	36	M		2	4
INDEXED 0 BYTE OFFSET	ROR	,X	76			1	3
INDEXED 1 BYTE OFFSET	ROR	Disp,X	66	D		2	5

Example

```
*                ** REPEAT ACTION FOLLOWING CNTRL **
                 CLC
REPT1            EQU   *
                 ROR   CNTRL
                 BCS   ACTN1      ACTION & REPEAT OR ESCAPE
                 NOP              *
                 NOP              ** DELAY **
                 NOP              *
                 BRA   REPT1
```

Stack Pointer Operation
RSP

RSP (Reset Stack Pointer)

Format

RSP

Condition Codes

Not affected.

Operation

SP <- $7F

Description

Resets the stack pointer to the top ($7F) of the stack.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	RSP		9C			1	1

Example

```
SEI             INTERRUPT DISABLE
RSP             RESET STACK POINTER
JSR     SYSINZ  SYSTEM INITIALIZE
CLI             INTERRUPT ENABLE
```

RTI (Return from Interrupt)

Format

RTI

Condition Codes

Set or cleared according to the first byte pulled from the stack.

Operation

SP <- (SP)+0001, CC <- (SP)
SP <- (SP)+0001, ACCA <- (SP)
SP <- (SP)+0001, IX <- (SP)
SP <- (SP)+0001, PCH <- (SP)
SP <- (SP)+0001, PCL <- (SP)

Description

The Condition Codes, Accumulator, Index Register and the Program Counter are restored according to the state previously saved on the stack. Note that the interrupt mask bit (I bit) will be reset if and only if the corresponding bit stored on the stack is zero.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	RTI		80			1	7

Example

```
JSR    KEYSCN   KEY INPUT
STA    INKEY    STORE KEY CODE
JSR    EXSWIN   INPUT EXTERNAL SW
STA    INSW     STORE SW CONDITION
RTI             RETURN TO INTERRUPT
```

RTS (Return from Subroutine)

Format

RTS

Condition Codes

Not affected.

Operation

SP <- (SP)+0001, PCH <- (SP)
SP <- (SP)+0001, PCL <- (SP)

Description

The stack pointer is incremented by one. The contents of the byte of memory, pointed to by the stack pointer, are loaded into the high byte of the program counter. The stack pointer is again incremented by one. The byte pointed to by the stack pointer is loaded into the low byte of the program counter.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	RTS		81			1	4

Example

```
        STA    WORK
        LDA    EXVAL5
        STA    RESULT
        CLC                RETURN CODE SET : OK
        RTS                *
    *
```

Arithmetic Opeartion
SBC

SBC (SuBtract with Carry)

Format

SBC P

Operation

ACCA <- (ACCA)-(M)-(C)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the result is 0; otherwise cleared.
C: Set if the absolute value of the contents of memory plus the carry bit C is greater than the absolute value of the contents of ACCA; otherwise cleared.

Description

Subtracts the contents of the memory and the carry bit C from the contents of ACCA, and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code				Number of MPU cycles
			Byte 1	Byte 2	Byte 3	No. of bytes	
IMMEDIATE	SBC	# Imm	A2	Imm		2	2
DIRECT	SBC	M	B2	M		2	3
EXTENDED	SBC	M	C2	MM	ML	3	4
INDEXED 0 BYTE OFFSET	SBC	,X	F2			1	2
INDEXED 1 BYTE OFFSET	SBC	Disp,X	E2	D		2	4
INDEXED 2 BYTE OFFSET	SBC	Disp,X	D2	DH	DL	3	5

Example

```
*
*                    (VAL1,VAL1+1)-(EXVAL5,EXVAL5+1)
*                                  = (EXVAL5,EXVAL5+1)

        LDA     VAL1+1    *
        SUB     EXVAL5+1  *
        STA     EXVAL5+1  *
        LDA     VAL1      *
        SBC     EXVAL5    *
        STA     EXVAL5    *
```

SEC (SEt Carry)

Format

SEC

Operation

C bit <- 1

Condition Codes

H: Not affected.
I: Not affected.
N: Not affected.
Z: Not affected.
C: Set.

Description

Sets the carry bit C in the condition code register.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
IMPLIED	SEC		99			1	1

Example

```
        BEQ     CHK84
        STA     RESULT
        SEC                     RETURN CODE SET : NG
        RTS
*
*
```

SEI (SEt Interrupt mask)

Format

SEI

Condition Codes

H: Not affected.
I: Set.
N: Not affected.
Z: Not affected.
C: Not affected.

Operation

I bit <- 1

Description

Sets the interrupt mask bit in the processor condition code register. The microprocessor is inhibited from servicing interrupts, and will continue with execution of the instructions of the program until the interrupt mask bit is cleared.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	SEI		9B			1	1

Example

```
SEI             INTERRUPT DISABLE
RSP             RESET STACK POINTER
JSR     SYSINZ  SYSTEM INITIALIZE
CLI             INTERRUPT ENABLE
```

STA (STore Accumulator)

Format

STA P

Operation

M <- (ACCA)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of ACCA is 1; otherwise cleared.
Z: Set if the contents of ACCA are 0; otherwise cleared.
C: Not affected.

Description

Stores the contents of ACCA in memory. The contents of ACCA remain the same.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
DIRECT	STA	M	B7	M		2	4
EXTENDED	STA	M	C7	MH	ML	3	5
INDEXED 0 BYTE OFFSET	STA	,X	F7			1	3
INDEXED 1 BYTE OFFSET	STA	Disp,X	E7	D		2	5
INDEXED 2 BYTE OFFSET	STA	Disp,X	D7	DH	DL	3	6

Example

```
LDA    VAL1
STA    WORK
LDA    RESULT
STA    0,X
LDA    #$FF
STA    EXVAL5,X
```

STX (STore indeX register)

Format

STX P

Operation

M <- (IX)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of IX is 1; otherwise cleared.
Z: Set if the contents of IX are 0; otherwise cleared.
C: Not affected.

Description

Stores the contents of IX in memory. The contents of IX remain the same.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
DIRECT	STX	M	BF	M		2	4
EXTENDED	STX	M	CF	MH	ML	3	5
INDEXED 0 BYTE OFFSET	STX	,X	FF			1	3
INDEXED 1 BYTE OFFSET	STX	Disp,X	EF	D		2	5
INDEXED 2 BYTE OFFSET	STX	Disp,X	DF	DH	DL	3	6

Example

```
LDX    VAL1
STX    WORK
LDX    RESULT
STX    0,X
LDX    #$FF
STX    EXVAL5,X
```

SUB (SUBtract)

Format

SUB P

Operation

ACCA <- (ACCA)-(M)

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of the result is 1; otherwise cleared.
Z: Set if the contents of the result are 0; otherwise cleared.
C: Set if the absolute value of the contents of memory is greater than the absolute value of the contents of ACCA; otherwise cleared.

Description

Subtracts the contents of memory from those of ACCA and places the result in ACCA.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code Byte 1	Byte 2	Byte 3	No. of bytes	Number of MPU cycles
IMMEDIATE	SUB	# Imm	A0	Imm		2	2
DIRECT	SUB	M	B0	M'		2	3
EXTENDED	SUB	M	C0	MH	ML	3	4
INDEXED 0 BYTE OFFSET	SUB	,X	F0			1	2
INDEXED 1 BYTE OFFSET	SUB	Disp,X	E0	D		2	4
INDEXED 2 BYTE OFFSET	SUB	Disp,X	D0	DH	DL	3	5

Example

```
LDA    VAL1      (VAL1)-(WORK)=(RESULT)
SUB    WORK
STA    RESULT
```

Interrupt Control
SWI

SWI (SoftWare Interrupt)

Format

SWI

Condition Codes

H: Not affected.
I: Set.
N: Not affected.
Z: Not affected.
C: Not affected.

Operation

PC <- (PC)+0001

Msp <- (PCL), SP <- (SP)-0001
Msp <- (PCH), SP <- (SP)-0001
Msp <- (IX), SP <- (SP)-0001
Msp <- (ACCA), SP <- (SP)-0001
Msp <- (CC), SP <- (SP)-0001
I bit <- 1
PC <- (SWI interrupt vector address)

Description

All the registers other than the stack pointer (SP) are pushed onto the stack. The interrupt mask bit is then set. Performs vectoring to the address indicated by the contents of the SWI interrupt vector address Note)

Note) $7FC and $7FD for HD6805S

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	SWI		83			1	9

Example

```
LDA    #$FF       *
STA    TIMER+1    * TIMER COUNTER SET
LDA    #$3F       *
STA    TIMER      *
LDA    #3         TIMER CODE SET
SWI               MONITOR SERVICE CALL
```

TAX (Transfer Accumulator to indeX register)

Format

TAX

Condition Codes

Not affected.

Operation

IX <- (ACCA)

Description

Transfers the contents of ACCA to IX. The contents of ACCA are unchanged.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	TAX		97			1	1

Example

```
TAX             SAVE ACCUMULATOR
LDA     #4      *
ADD     RESULT  ** ADD (RESULT+4)
STA     RESULT  *
TXA             REVIVE ACCUMULATOR
```

TST (TeST)

Format

TST Q
TST A
TST X

Operation

(IX) - 00
or
(ACCA) - 00
or
(M) - 00

Condition Codes

H: Not affected.
I: Not affected.
N: Set if the most significant bit of ACCA, IX or M is 1; otherwise cleared.
Z: Set if the contents of ACCA, IX or M are 0; otherwise cleared.
C: Not affected.

Description

Sets N and Z bits of the condition code register according to the contents of ACCA, IX or M.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
ACCUMULATOR	TST	A	4D			1	1
INDEX REG.	TST	X	5D			1	1
DIRECT	TST	M	3D	M		2	4
INDEXED 0 BYTE OFFSET	TST	,X	7D			1	3
INDEXED 1 BYTE OFFSET	TST	Disp,X	6D	D		2	5

Example

```
        TST     CNTRL
        BEQ     INIT00      CNTRL=$00
*
        TST     WORK
        BMI     MINS00      WORK=(1XXX XXXX)
*
```

TXA (Transfer indeX register to Accumulator)

Format

TXA

Condition Codes

Not affected.

Operation

ACCA <- (IX)

Description

Transfers the contents of IX to ACCA. The contents of IX are unchanged.

Addressing Mode and Number of MPU Cycles

Addressing Mode	Mnemonic	Operand type	Instruction code			No. of bytes	Number of MPU cycles
			Byte 1	Byte 2	Byte 3		
IMPLIED	TXA		9F			1	1

Example

```
TAX             SAVE ACCUMULATOR
LDA    #4       *
ADD    RESULT   ** ADD (RESULT+4)
STA    RESULT   *
TXA             REVIVE ACCUMULATOR
```

4. Pin Assignment and Dimensional Outline

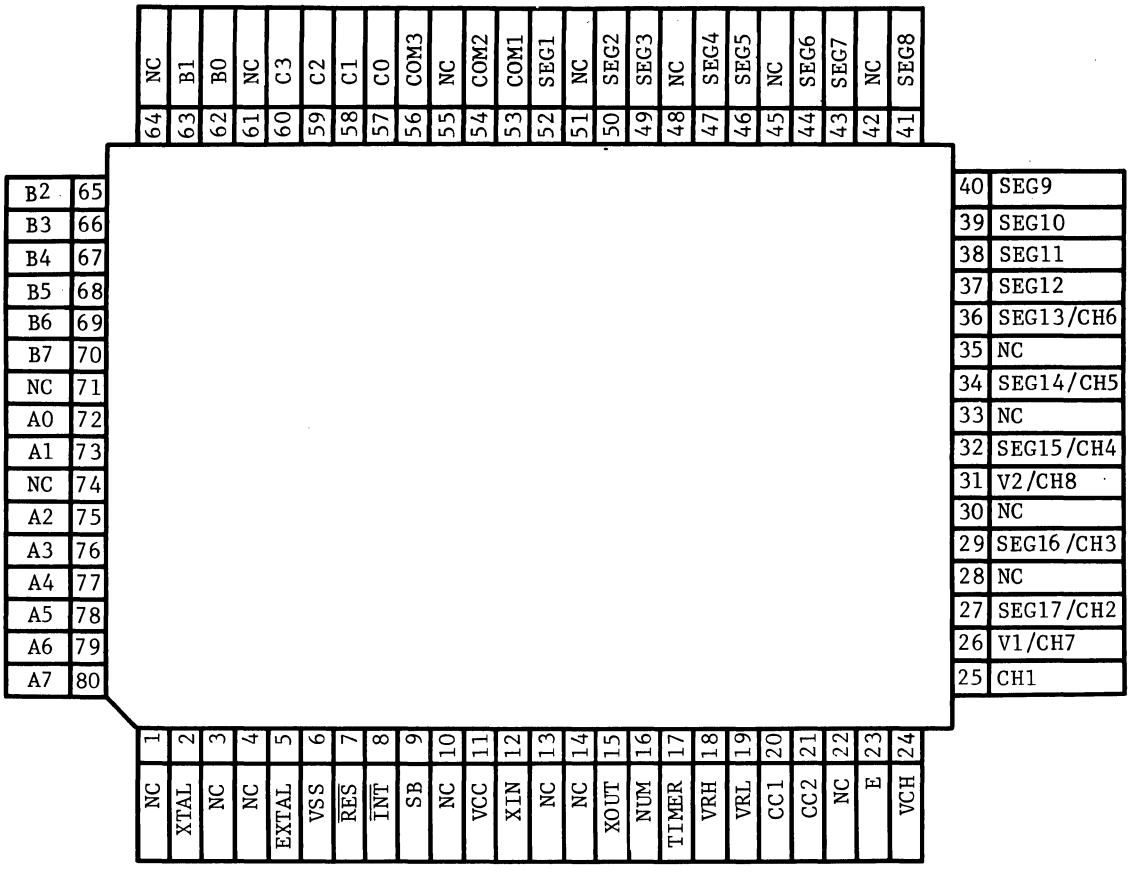
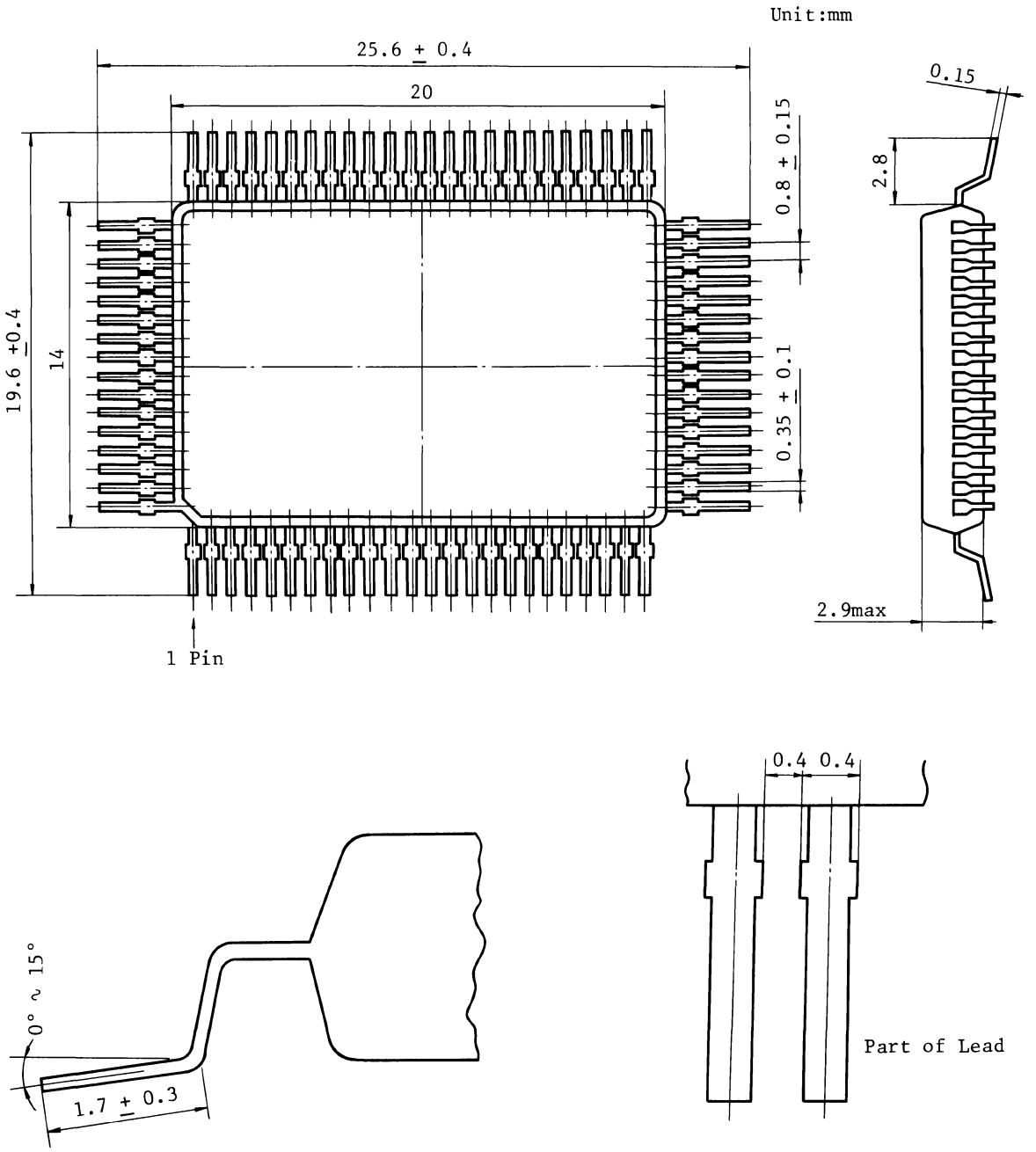


Figure 4-1 Pin Assignment (Top View)



Note
Lead of corner part is below the bottom of package

Figure 4-2 Dimensional Outline (Unit : mm)

5. Electrical Characteristics

■ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3~ +5.5	V
Input Voltage	V_{in}	-0.3 ~ V_{CC}+0.3	V
Output Voltage	V_{out}	-0.3 ~ V_{CC}+0.3	V
Operating Temparature	T_{opr}	-20~+75	°C
Storage Temparature	T_{stg}	-55~+125	°C

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded.
Normal operation should be under recommended operating conditions.
If these conditions are exceeded,it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS (V_{CC}=3.0 0.8V, V_{SS}=0V, Ta=−20∿+75°C, typ means typical value at V_{CC}=3.0V unless otherwise noted.)

● DC CHARACTERISTICS

Item			Symbol	Test Condition	min	typ	max	Unit
Input "High" Level Voltage	XTAL, XIN		V_{IH}	Connect C_L=0.5 F to V_{CH}	V_{CC}-0.3	-	V_{CC}	V
	$\overline{RES}$, $\overline{INT}$, SB				0.5V_{CC}+0.9	-	V_{CC}	V
	TIMER				0.8V_{CC}	-	V_{CC}	V
	NUM (Normal Mode)				V_{CC}-0.2	-	V_{CC}	V
Input "Low" Level Voltage	XTAL, XIN		V_{IL}	Connect C_L=0.5 F to V_{CH}	V_{SS}	-	V_{CC}-1.8	V
	$\overline{RES}$, $\overline{INT}$, SB				V_{SS}	-	0.2V_{CC}	V
	TIMER				V_{SS}	-	0.2V_{CC}	V
	NUM (Test Mode)				V_{SS}	-	0.2	V
Self Check Input Voltage	NUM (Self Check Mode)		V_{IM}		0.5V_{CC} -0.2	-	0.5V_{CC} +0.2	V
Input Pull-Up Current	$\overline{RES}$ ($\overline{INT}$:Mask Option) NUM		$-I_{R1}$	V_{CC}=3.0V, Vin=0V	3	15	30	μA
Input Leackage Current	TIMER, SB		$\|I_{IN}\|$	Vin = 0V V_{CC}	-	-	1.0	μA
Current Dissipation	Crystal Oscillation	During System Operation	I_{CC1}	f =400kHz No load. Tested after setting up the internal status by self check.	-	100	200	μA
		At Halt			-	40	60	μA
		At Stand-By			-	1	5	μA
		At A/D Operation			-	160	400	μA
	RC Oscillation	During System Operation	I_{CC2}	R =100kHz No load. Tested after setting up the internal status by self check.	-	120	200	μA
		At Halt			-	30*	60*	μA
		At Stand-By			-	1	5	μA
		At A/D Operation			-	180	420	μA
Output "Low" Level Voltage	E		V_{OL}	I_{OL} =30 A	-	-	0.3	V

* In the case that OSC1 is stopped by Halt.
These values can be changed without notice, because they are provisional.

● AC CHARACTERISTICS

Item		Symbol	Test Condition	min	typ	max	unit
Operating Clock Frequency		f_{cf}		100	400	500	kHz
Cycle Time		f_{cyc}		8	10	40	μs
Oscillation Frequency (Resistor Option)		f_{OSCR}	R =100kΩ ±1%	300	400	500	kHz
External Clock Duty		Duty		45	50	55	%
Oscillation Start Time (Crystal Option)		t_{OSCf}	C_C =10pF±20%, RS =1kΩ max	-	-	150	ms
Oscillation Start Time (Resistor Option)		t_{OSCR}	R=100kΩ±1%, Connect C_L=0.5μF to V_{CH}	-	-	2	ms
Oscillation Start Time (32kHz)		t_{OSCl}	CC=10pF±20%. RS=20kΩ max	-	-	1	s
Internal Capacitance of Oscillator	EXTAL	C_O		-	10	-	pF
	XOUT			-	10	-	pF
Delay Time of Oscillation Delay Time		t_{PLY}	Selected by mask option	0	-	500	ms
Reset Delay Time		t_{PLH}	External Capacitance =2.2μF	200	-	-	ms
RES Pulse Width		tRWL		t_{cyc}+ 1	-	-	μs
INT Pulse Width		t_{PWL}	When OSCl is not stopped by Halt	t_{cyc}+ 1	-	-	μs
			When OSCl is stopped by Halt.	32	-	-	μs
TIMER Pulse Width		t_{TWL}	In the case of counter	t_{cyc}+ 1	-	-	μs

● PORT CHARACTERISTICS

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Level Voltage	Port A,B,C	V_{OH}	CMOS Output, I_{OH}=-100μA	V_{CC}-0.3	-	-	V
	Port A,B,C		Key Load CMOS Output I_{OH}=-10μA	V_{CC}-0.3	-	-	V
Output "Low" Level Voltage	Port A,B,C	V_{OL}	I_{OL}=100μA	-	-	0.3	V
Input "High" Level Voltage	Port A,B,C	V_{IH}		0.8V_{CC}	-	V_{CC}	V
Input "Low" Level Voltage	Port A,B,C	V_{IL}		V_{SS}	-	0.2V_{CC}	V
Input Leackage Current	Port A,B,C	$\|I_{IN}\|$	V_{in}=0V ~ V_{CC}	-	-	1.0	μA
Input Pull-Up Current	Port A,B,C	$-I_{R2}$	V_{CC}=3.0V, V_{in}=0V	4	20	40	μA

● LCD DRIVER OUTPUT CHARACTERISTICS (V_{CC}=3.0V, V_{SS}=0V, Ta=-20∿ +75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Level Voltage	Segment	V_{OH1}	V = 1.00V, V = 2.00V I_{OH}= -1 μA	2.8	-	-	V
		V_{OH2}		1.8	-	-	V
		V_{OH3}		0.8	-	-	V
Output "Low" Level Voltage	Segment	V_{OL1}	V = 1.00V, V = 2.00S I_{OL}= 1 μA	-	-	2.2	V
		V_{OL2}		-	-	1.2	V
		V_{OL3}		-	-	0.2	V
Output "High" Level Voltage	Common	V_{OH1}	v = 1.00V, V = 2.00V I_{OH}= -5 μA	2.8	-	-	V
		V_{OH2}		1.8	-	-	V
		V_{OH3}		0.8	-	-	V
Output "Low" Level Voltage	Common	V_{OL1}	V = 1.00V, V = 2.00V I_{OH}= 5 μA	-	-	2.2	V
		V_{OL2}		-	-	1.2	V
		V_{OL3}		-	-	0.2	V
Dividing Resistor		R_{LCD}	Tested between V and V	45	90	180	k
Output "High" Level Voltage	Segment	V_{OH}	In the case of Output Port, I_{OH}=-30 μA	V_{CC}-0.3	-	-	V
Output "Low" Level Voltage	Segment	V_{OL}	In the case of Output Port, I_{OL}=30 μA	-	-	0.3	V

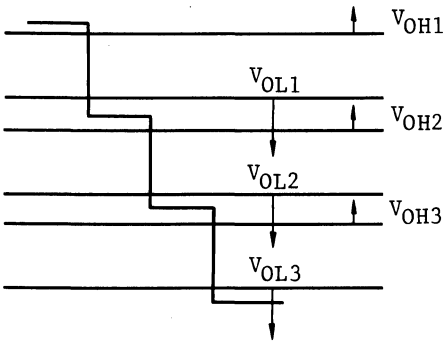


Figure 5-1 Output Level of SEG and COM

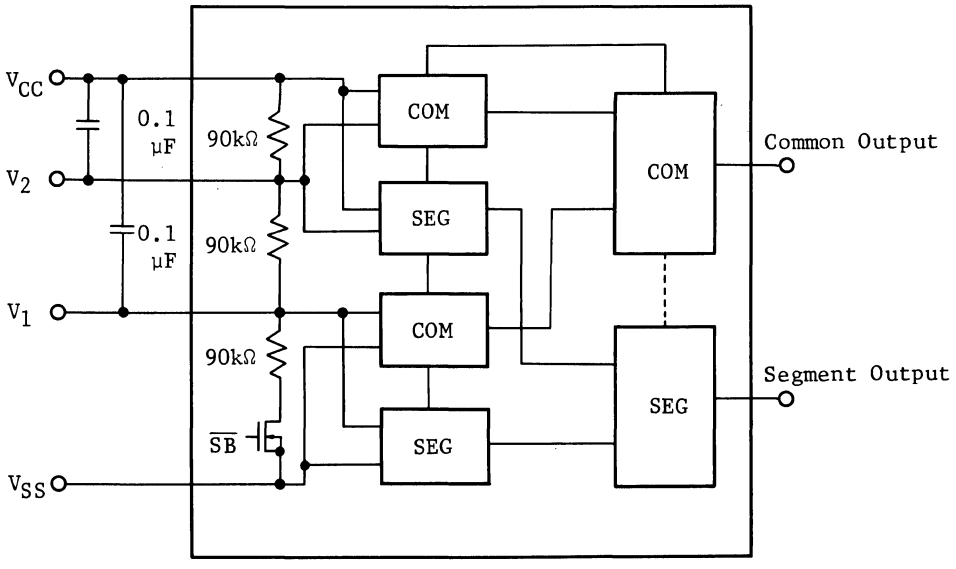


Figure 5-2 Power Supply Circuit for LCD Display

● A/D CONVERTER CHARACTERISTICS *(V_{CC}=3.0V, V_{SS}=0V, Ta=-20°C ∿ +75°C, C=300pF, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Conversion Accuracy	Resolution			-	-	8	bit
	Absolute Accuracy		V_{RL}=0.2V < V_{in} < V_{RH}=2.0V	-2	-	+2	LSB
Reference Voltage	"High" Side	V_{RH}		-	-	V_{CC}	V
	"Low" Side	V_{RL}		V_{SS}	-	-	V
	V_{RH} - V_{RL}	V_{REF}		2.0	-	-	V
	Input Range	V_{IN}		V_{RL}	-	V_{RH}	V
	Input Dynamic Range	V_{DYN}		0.2	-	V_{CC}-1.0	V
Ladder Resistor (V_{RH}- V_{RL})		R_{HL}		40	80	160	kΩ
Conversion Time		t_{CNV}		2	-	4	ms
Programmable Voltage Comparison	Judge Error		V_{RL}=0.2V<V_{in} < V_{RH}=2.0V	-4	-	+4	LSB
	Judge Time	t_{CMP}		-	-	60	µs

* These value can be changed without notice, because they are provisional.

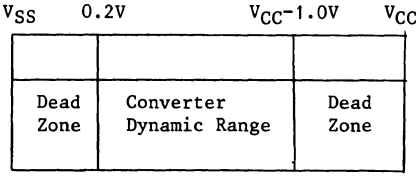


Figure 5-3 Dynamic Range of the Comparator

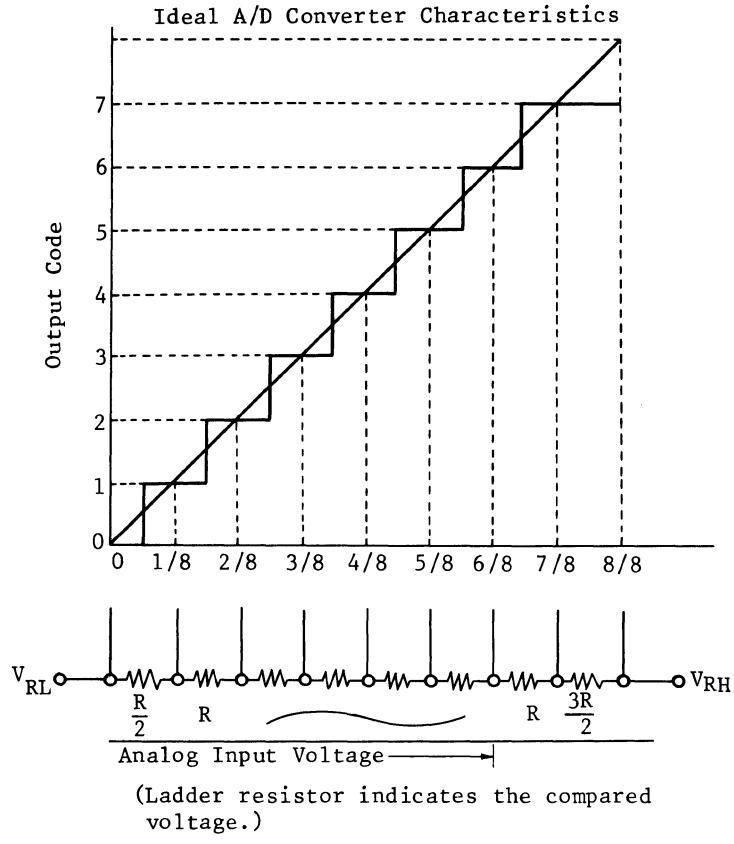


Figure 5-4 Example of 3 bit Resolution

6. Application

6.1 Test Mode

(1) HD63L05 Test Mode

The HD63L05 can take two operation modes based on the state of the NUM terminal. When the NUM terminal is pulled up to Vcc,the MCU operates in the normal mode. However,when the NUM terminal is grounded to the GND, the HD63L05 goes into the test mode. This mode can be used for testing the MCU.

(2) Bus Line

When the HD63L05 is operating in the test mode,the connection with the external circuit is performed through I/O ports.

The port A becomes the o2 synchronous input data bus. In the test mode,the internal data bus which enters the CPU inside the MPU is disconnected,and all the data and instructions are sent to the CPU through the port A.

The port B can be used for data bus. It is possible to refer to inside ROM or RAM through the peripheral data bus.

The port C becomes the control input for A/D converter.

NOTE

The address bus,R/W signal,and LIR signal are not available in the test mode.

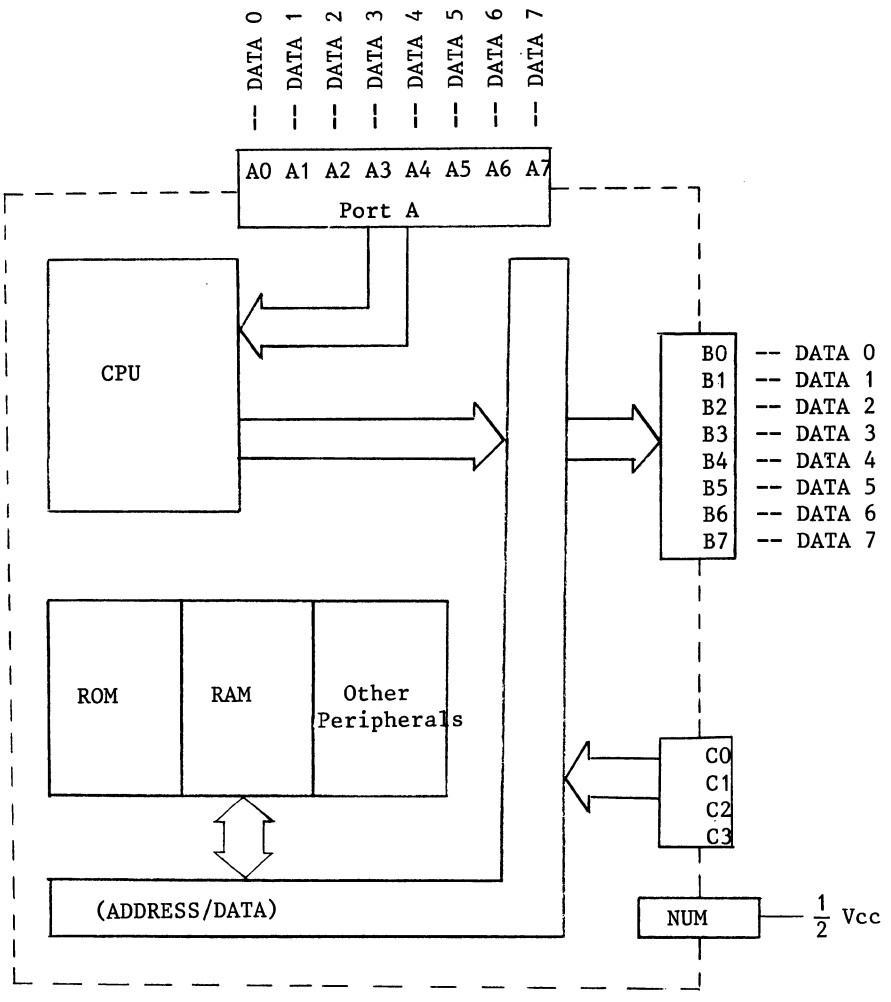


Figure 6-1 Test Mode Block Diagram of the HD63L05

6.2 How to confirm Operation Frequency

When the E terminal of the HD63L05 is pulled up to Vcc through the resistor,o2 clock output from the OSC1 is available as output of the E terminal. The clock output from the OSC2 is available as outputs of the COM1,COM2,or COM3.

6.3 LCD Expansion

SEG0~SEG17,V1,and V2 terminals can be used for output terminals by mask-option. Connecting external LCD driver HD61602 to the Hd63L05, LCD driving capability of the system will be greately improved. An example of the LCD expansion is shown in Fig.6-2.

SEG0~SEG7 are used for data bus to provide display data and address data (2-Byte length) to the LCD driver. A write clock for the LCD driver can be automatically obtained if the LCD1 register is used as a buffer register for the output data. When the data is written into the LCD1, the write clock for the external LCD driver is obtained on the one of the SEGMENT output terminals through the pin-location block in the MCU (same as the display data).

The LCD driver HD61602 has 4 software-controlled driving methods (Static drive,1/2 Bias 1/2 Duty ,1/3 Bias 1/3 Duty, and 1/3 Bias 1/4 Duty) with 51 segment output terminals. Therefore,the LCD driving capability can be expanded up to 204 (51 x 4 =204) segments with one external LCD driver HD61602 in case of 1/3 Bias 1/4 Duty Driving.

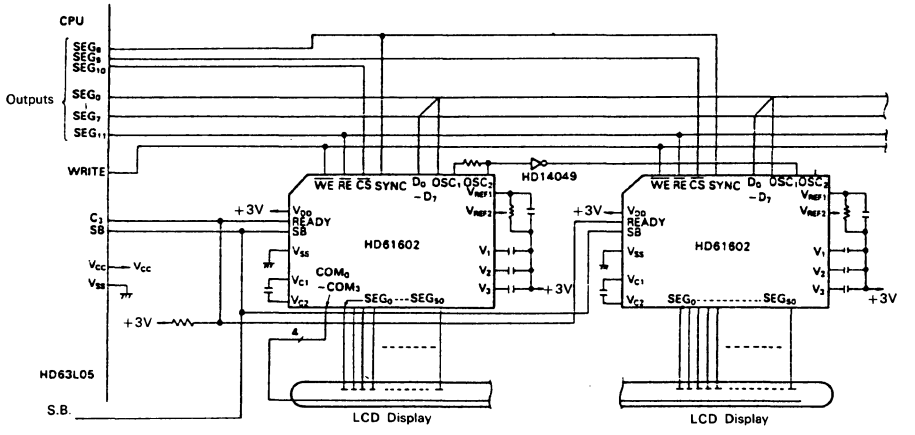


Figure 6-2 Example of LCD Expansion System

6.4 Method of the DAA (Decimal Adjust Accumulator)

(1) Function
This subroutine is a simulation of the DAA instruction performed by the HD63L05. This is used immediately after the addition of two bytes (ADD and ADC) whivh consist of two-digit BCD(Binary Coded DEcimal) respectively. This subroutine converts the result of the BCD addition into two-digit BCD again and produces it to the accumulator.

(2) Linkage
The digit to be converted is input to the accumulator and operation jumps to the routine.

(Example)

```
LDA   ARG1
ADD   ARG2
JSR   DAA---Jumps to the DAA subroutine
STA   ARG3
```

Two-digit BCDs are stored in the ARG1 and ARG2 respectively and the operation jumps to the DAA after addition.The result of the addition is converted into BCD and is stored in the ARG3.

(3) Result
The binary coded decimal digit is output to the accumulator.

(4) Register to be influenced
(i) IX may be guaranteed the contents before jumping to the routine.
(ii) Of the CC(Condition Code Register),the C bit is set when the result of the BCD addition or decimal conversion is rounded up. The previous contents in each bit of H,N and Z cannot be guaranteed.

(5) Program specification
Program specification is shown in Table 6-1.

Table 6-1 Program Specification

Number of words(B)	Work area(B)	Execution time(usec)	Reentrant	Relocation
31	2*	410	Not possible**	Possible
Intermediate interrupt				
Possible				

*: It is necessary to provide this work area within the stored RAM($020 $07F).
**: Depending on the situation,it may be necessary to mask the interrupt during the execution of this subroutine.

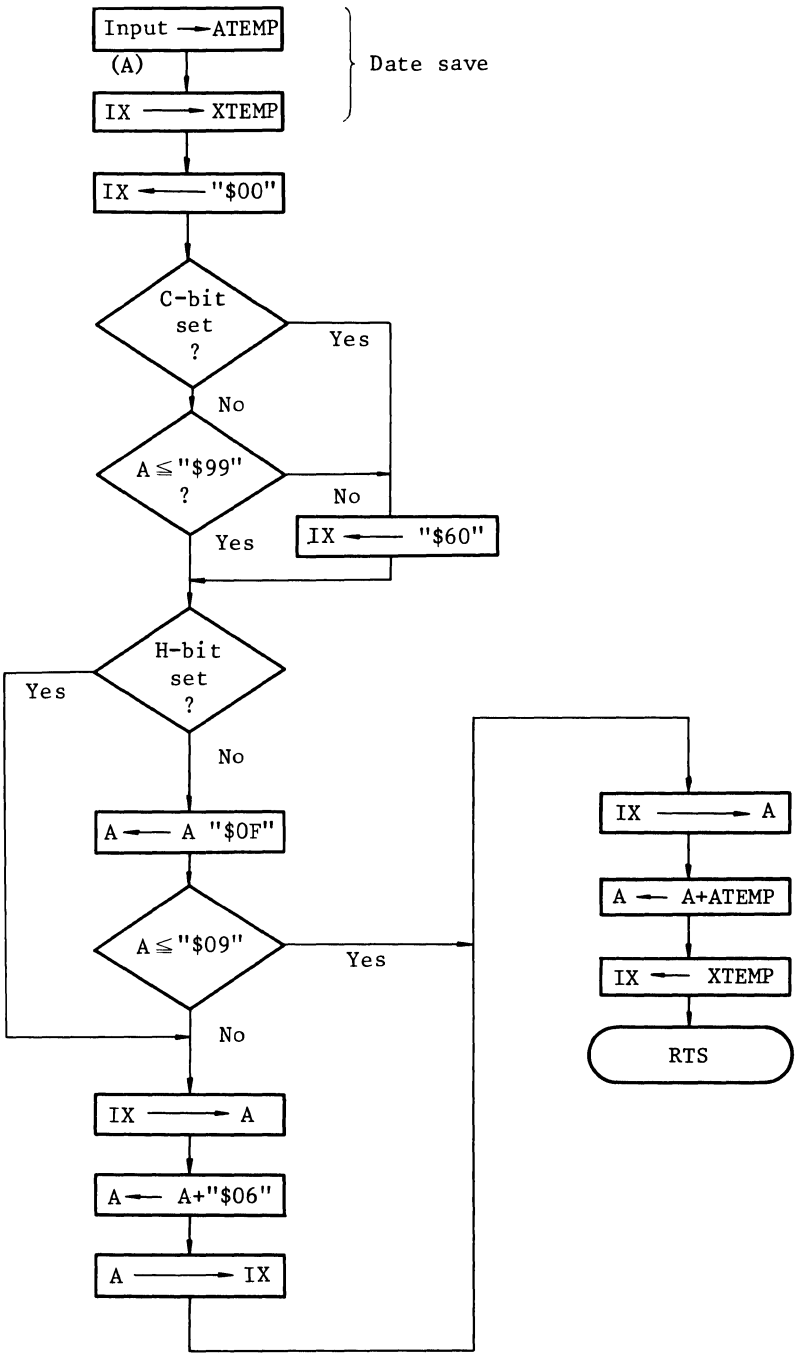


Figure 6-3 DAA Subroutine Flow Chart

Table 6-2 DAA Subroutine Program List

ADDRESS	OP CODE					COMMENT
0044			ATEMP	RMB	1	Work area
0045			XTEMP	RMB	1	
0099	B7	44*	DAA	STA	ATEMP*	Saves the input value
009B	BF	45*		STX	XTEMP*	
009D	5F			CLR	X	Vacates the IX to store the
009E	25	04		BCS	DAAH6	converted value.
00A0	A1	99		CMP	#$99	Branches if the value is equal
00A2	23	02		BLS	DAALOW	to or less than $99.
00A4	AE	60	DAAH6	LDX	#$60	The higher 4 bits are also con-
						verted if not so.
00A6	29	06	DAALOW	BHS	DAAL6	
00A8	A4	0F		AND	#$0F	
00AA	A1	09		CMP	#$09	It is not necessary to convert
00AC	23	04		BLS	DAADNE	if the lower 4 bits are less than
00AE	9F		DAAL6	TXA		$09.Therefore,branching is performed.
00AF	AB	06		ADD	#$06	The converted value of the lower
00B1	97			TAX		4 bits.
00B2	9F		DAADNE	TXA		Stores the converted value in A,
00B3	BB	44*		ADD	ATEMP*	adds it to the original(ATEMP)
00B5	BE	45*		LDX	XTEMP*	and prduces an output.
00B7	81			RTS		

* It is necessary to provide ATEMP and XTEMP within the stored RAM (Address $020~$07F) in the work area.

6.5 Cautions on the Program of the Write Only Register and Control Register

It is not possible to change the contents of the Write Only Register (for example, the DDR Data Direction Register of the I/O port) of the HD63L05 by applying the Read/Modify/Write instructions.

(1) The Write Only Register(for example the DDR of the I/O port) cannot read,the Read/Modify/Write instructions are executed in the the following sequence.

(i) Reading the contents of the specified address
(ii) Changing the read-out data
(iii) Returning the changed data to the original address

It is clear that the Read/Modify/Write instructions cannot be applied to the Write Only Register such as DDR.

(2) For the same reason,do not set the DDR of the I/O port by using the BSET andBCLR instructions of the HD63L05.

(3) It is needed to pay attention to the System Control Register,the Timer Control Register,and the A/D Control Register when BSET,BCLR,or Read/ Modify/Write instructions are applied to them. If own interrupt request occured onto the interrupt request bit (bit7) of the Control Register between read cycle and write cycle of these instructions,the bit7 might be cleared in the write cycle and not acknowledged by CPU.

(4) Store instructions such as STA or STX are used to correctly write in the Write Only Register or to avoid missing an interrupt of the Control Register.

7. Evaluation Chip

The HD63L05E is a CMOS evaluation chip for the HD63L05. Connecting an external EPROM (HN462732) to the chip,it can be operated as a single chip microcomputer HD63L05. This chip is a 100 pins flat package. (See Figure 7-1)

7.1 Block Diagram

Input signals and output signals of the HD63L05E are described below. Basically,same terminal name means same function as the HD63L05.

- Vcc,Vss
  Power is supplied to the LSI by using these terminals. Vcc has a voltage of 3.0V+0.8V and Vss is grounded.

- $\overline{\text{INT}}$
  This terminal is used to enboke an external interruption to the LSI.

- XTAL,EXTAL
  These terminals are control input terminals to the built-in clock circuit. A crystal (400kHz typ.) is connected to these terminals.

- TIMER
  This terminal in an external input terminal to count down the internal timer circuit.

- $\overline{\text{RES}}$
  Used to reset the LSI.

- STANDBY
  An external input terminal used to stop the LSI and hold data.

- A/D Input Terminals (CH1∿CH8)
  Input terminals for analog voltages needed for A/D conversion. These may also be used as level check input under program control.

- VRH,VRL
  Reference voltages for A/D conversion are applied to these two terminals.

- CC1,CC2
  An offset compensating capacitor (300pF typ.) is connected between CC1 and CC2.

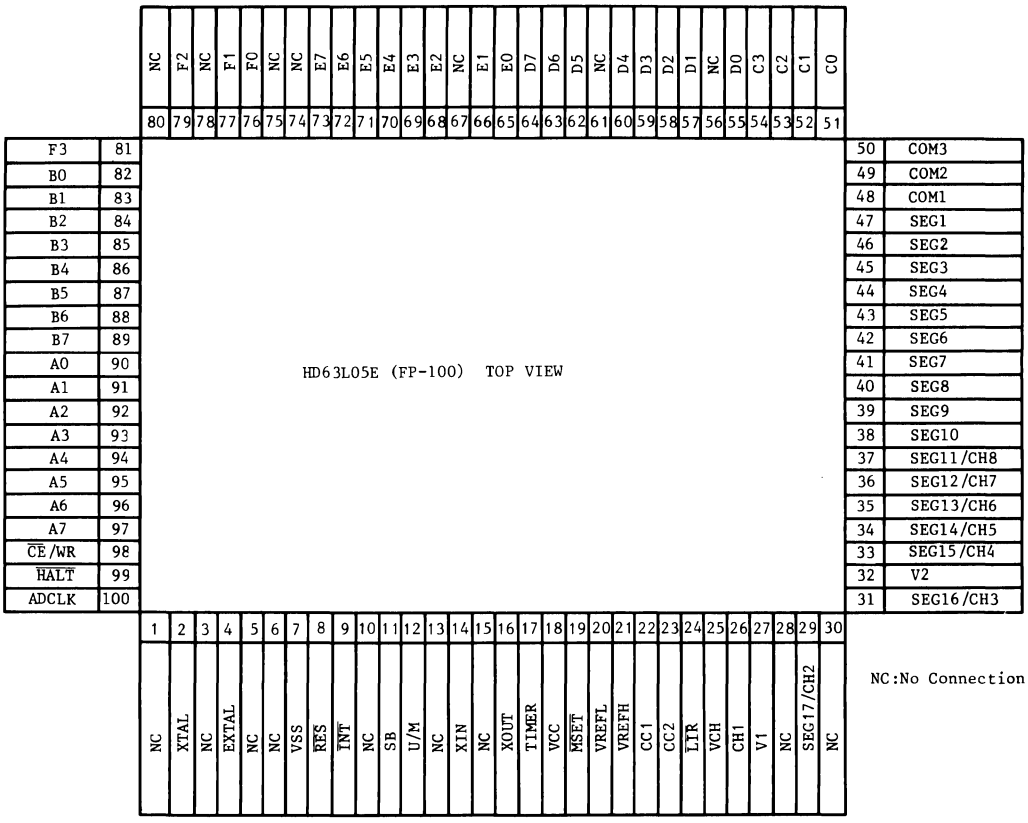


Figure 7-1 Pin Assignment of the HD63L05E

● XIN,XOUT
A crystal(32.768kHz) is connected to these terminals,if necessary.

● VCH
Output terminal from internal voltage regulator. A capacitor (0.5uF) is connected between VCH and Vcc.

● $\overline{\text{MSET}}$
This terminal is not used for user application. Connect it to Vcc.

● ADCLK
1/4 frequency of OSC1 (100kHz typ. synchonized with $\phi 2$) is available from this terminal. NMOS open-drain output.

● U/M
The HD63L05E can take two operation modes based on the state of this terminal. When the terminal is connected to Vcc,the LSI operates as a single chip microcomputer with external EPROM. However,when the terminal is grounded,the HD63L05E operates in external extension mode.

● Input/Output Terminals (A0∿A7,B0∿B7,C0∿C3)
These 20 terminals consist of two 8-bit ports and one 4-bit port. Each terminals may be used as an input or output under program control of the data direction register. These are NMOS open-drain output.

● D0∿D7
These terminals are input terminals for instruction or data from external data bus. For example,output from an external EPROM are applied to these terminals.

● E0∿E7
These terminals are NMOS open-drain outputs. When the U/M is logical "1", the address bus (A0∿A7) from the HD63L05E is available.When the U/M is logical "0",the port E can be used for address bus or data bus. When $\phi 2$ is "Low" the port E becomes address bus. When $\phi 2$ is "High" the port E becomes the peripheral data bus.

● F0∿F3
These terminals are NMOS open-drain outputs. When the U/M is logical "1", the address bus (A8∿A11) is available.When the U/M is logical "0",the port F can be used for address bus while $\phi 2$ is "Low".

● $\overline{\text{CE}}$/WR
This terminal is a NMOS open-drain output. When the U/M is "High", chip enable signal (means address bus is in from $080 to $FFF) is available. When the U/M is "Low", Read/Write clock is available.

● $\overline{\text{LIR}}$
NMOS open-drain output. Fetch signal is available from this terminal.

● $\overline{\text{HALT}}$
NMOS open drain output. When standby signal is acknowledged,the output from this terminal becomes "Low" to control external clock source.

● V1,V2
These are terminals for LCD driver. V1 and V2 are connected to Vcc via capacitors(0.1μF each).

● Liquid Crystal Driver Terminals (COM1~COM3,SEG1~SEG17)
COM1~COM3 are for driving common electrodes,while SEG1~SEG17 are for driving segments. SEG11~SEG17 can be used as analog inputs for A/D converter by the preset data in the EPROM.

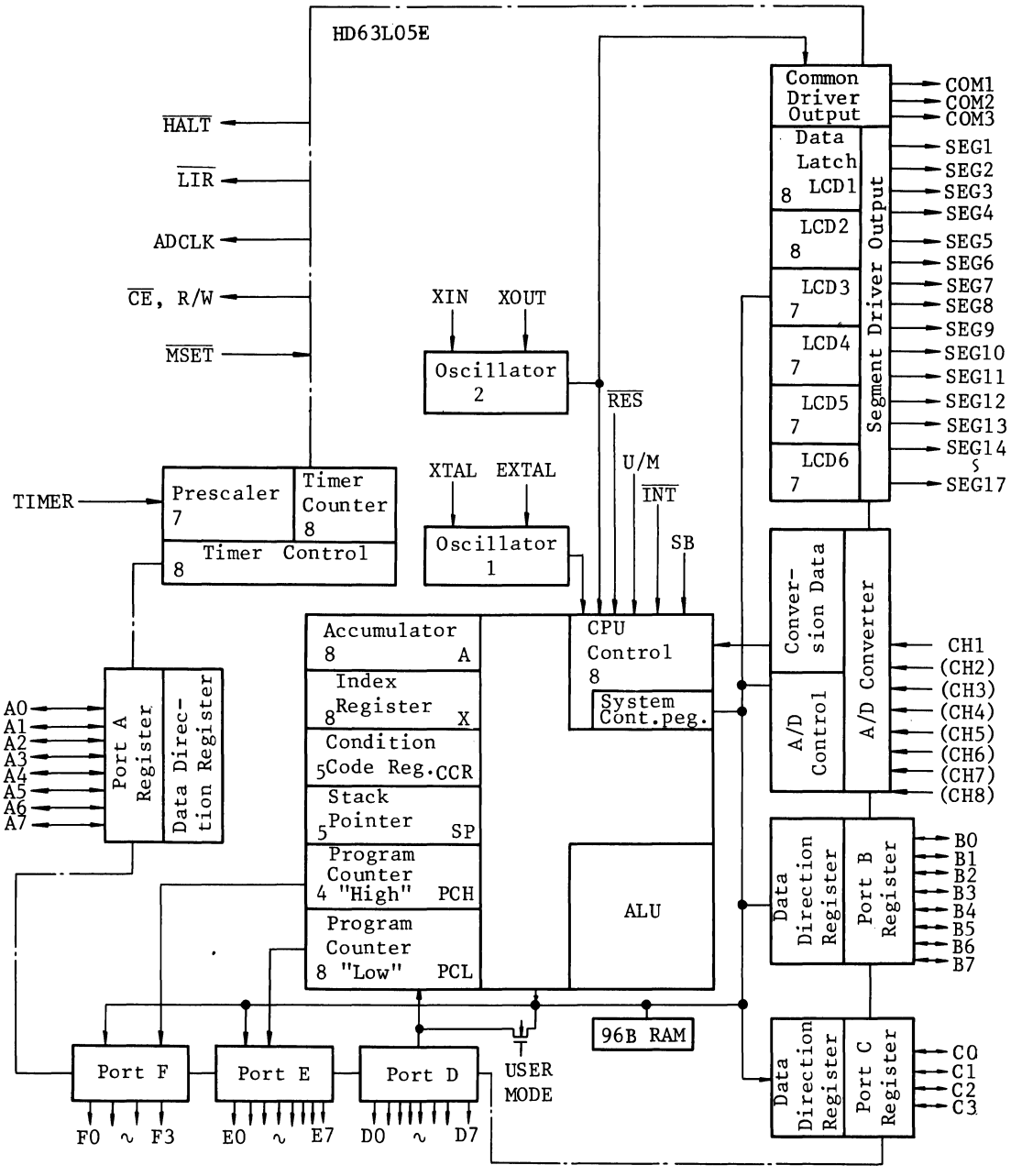


Figure 7-2 HD63L05E Block Diagram

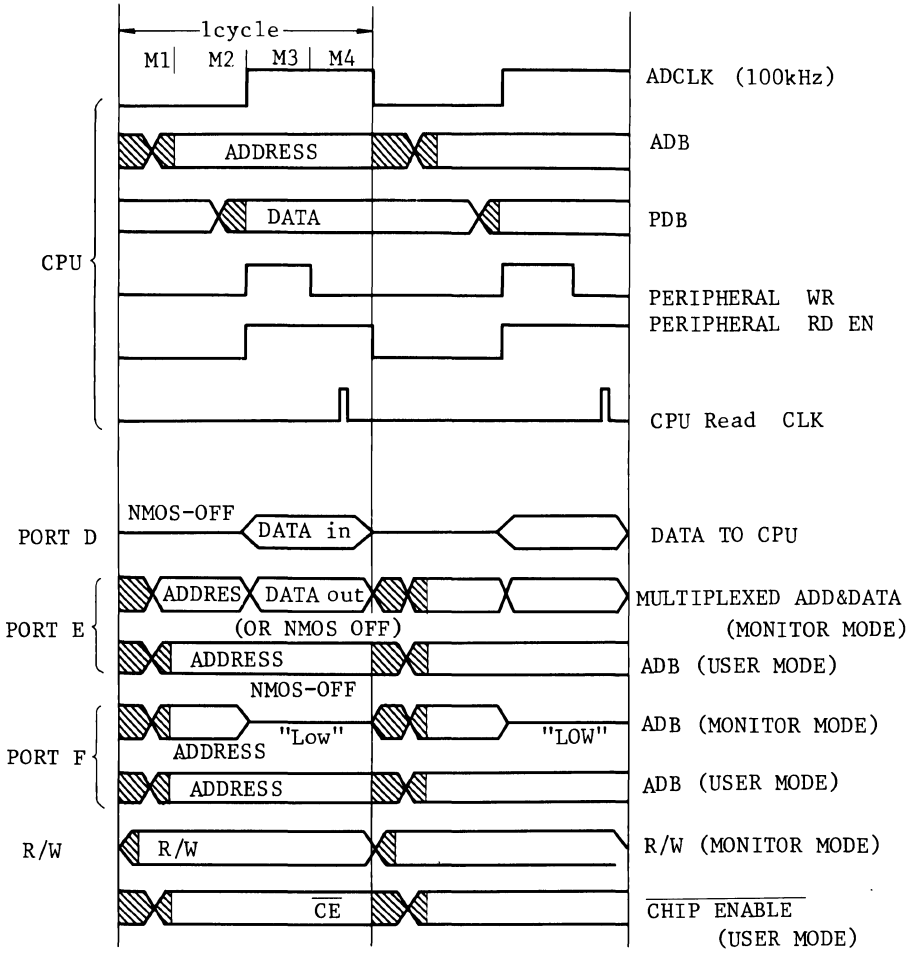


Figure 7-3 HD63L05E Timing Chart

7.2 How to make a single chip microcomputer HD63L05 with the HD63L05 and EPROM

1) Power
3V is supplied to Vcc and Vss is grounded. VCH is connected to Vcc via 0.5μF.(approximately 1V is obtained.) V1 is connected to Vcc via 0.1 μF and 1V is available while V2 is connected to vcc via 0.1μF and 2V is available. (See Figure 7-4)

2) Clock
The XTAL can be used for internal oscillator or input terminal from external clock source. In case of external clock, the "High" level of the clock signal should be Vcc and the "Low" level should be VCH. (See Figure 7-5)

3) Control Signals
The U/M terminal and $\overline{\text{MSET}}$ are connected to Vcc.

4) Interfacing to the user system (See Figure 7-6)

- I/O port (A0~A7,B0~B7,C0~C3)
Each terminals has NMOS open-drain output. Therefore, "High" level of the output is obtained by connecting a resistor to Vcc.
In case of input port, pull up the terminal to Vcc via resistor,if necessary.

- Control terminals ($\overline{\text{RES}}$,$\overline{\text{INT}}$,S.B.,TIMER)
Only RESET is connected to Vcc via internal pull-up PMOS in the LSI.
To avoid the floating input to $\overline{\text{INT}}$,S.B,or TIMER,connect pull-up resistors between these terminals and Vcc,if necessary.

- Others
SEG1~SEG17 are fixed as 1/3 bias 1/3 duty drive and the combination of the LCD register bit and SEG output are also fixed. This chip cannot support the modification of the pin location block in the HD63L05 and output port option. (See Table 7-1)

- A/D inputs are supported from CH1 to CH8

  The selection of SEG13 SEG17(for LCD driver or analog input CH2~CH8) can be specified by the external EPROM data. (See Table 7-2)

5) Interfacing to EPROM (See Figure 7-7)

- Data and instruction input (D0~D7)
The output terminals from the EPROM are connected to these terminals. To reduce the "High" level of the data as small as Vcc of the LSI,connect pull-down resistors to ground.

- Address Outputs(E0~E7,F0~F3)
The address bus (A0~A11) is available from these terminals for EPROM. Connect them to Vcc of the EPROM with pull-up resistors.

- Chip Select ($\overline{\text{CE}}$/WR)
Connect it to Vcc of the EPROM with pull-up resistor.When ROM address is selected (from $080 to $FFF), "Low" level is available.

6) Others ($\overline{\text{HALT}}$,$\overline{\text{LIR}}$,ADCLK)
Normally,these terminals are not used. Open them.

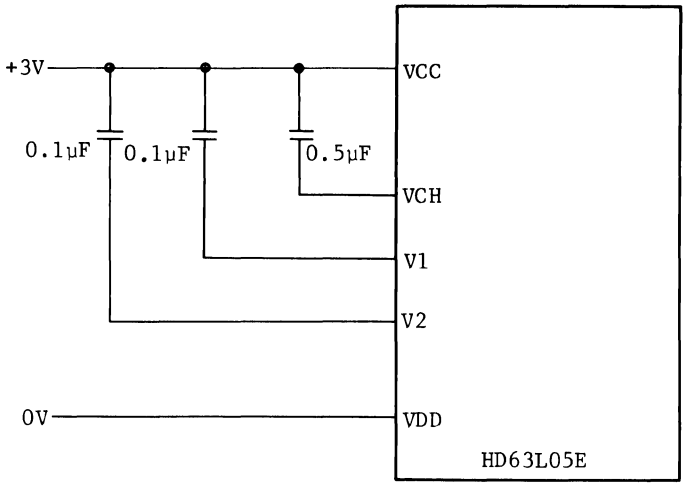
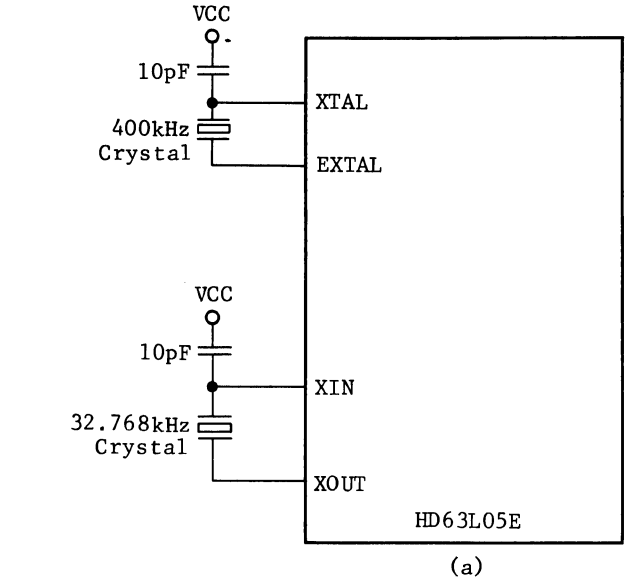


Figure 7-4 Connections for Power Supplying



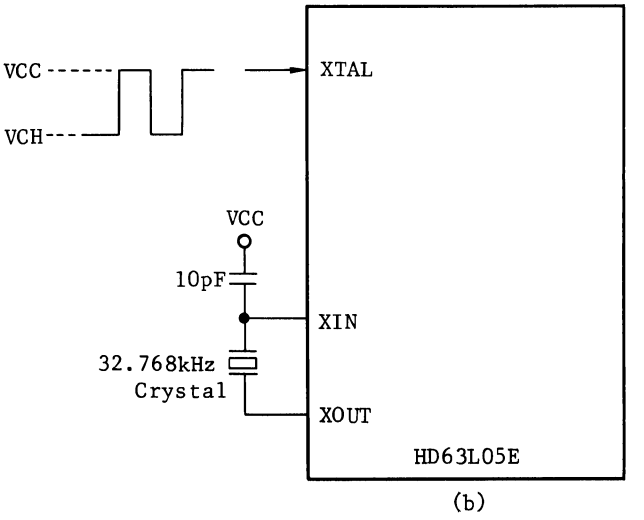


Figure 7-5 Connections for the Oscillators

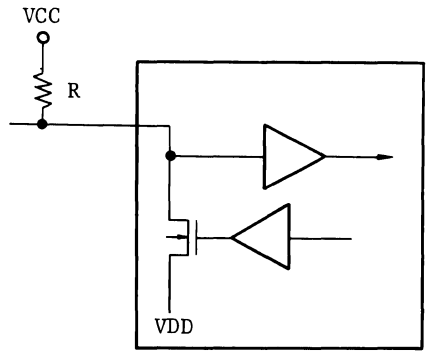


Figure 7-6 Configuration of NMOS open-drain Output

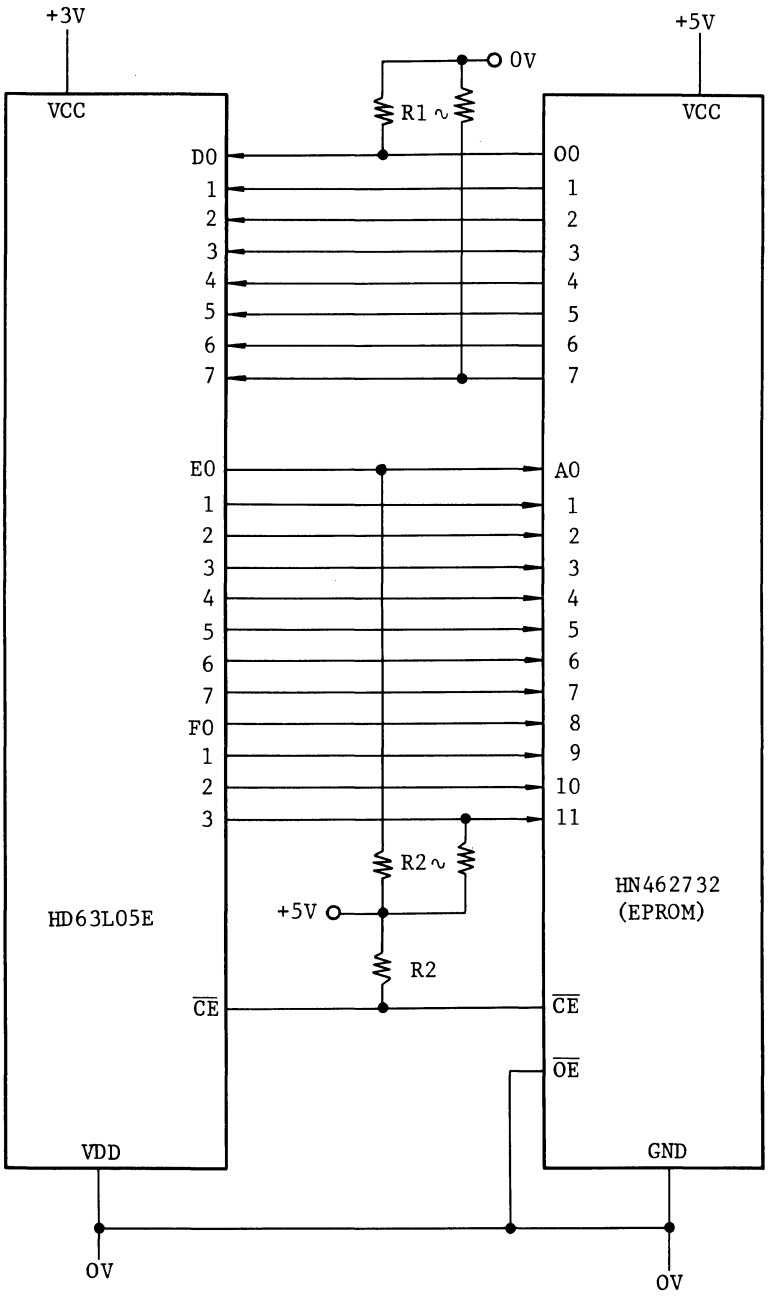


Figure 7-7 Interfacing between HD63L05E and EPROM

Table 7-1 Connections of the Pin location Block

LCD register	Timing	SEGMENT Terminal
LCD1-0	COM2	SEG2
1	COM3	SEG2
2	COM1	SEG3
3	COM1	SEG2
4	COM1	SEG1
5	COM2	SEG1
6	COM3	SEG1
7	Not used	
LCD2-0	COM2	SEG4
1	COM3	SEG5
2	COM2	SEG5
3	COM1	SEG4
4	COM2	SEG3
5	COM3	SEG3
6	COM3	SEG4
7	Not used	
LCD3-0	COM2	SEG6
1	COM2	SEG7
2	COM1	SEG7
3	COM1	SEG6
4	COM1	SEG5
5	COM3	SEG6
6	COM3	SEG7
LCD4-0	COM2	SEG9
1	COM3	SEG9
2	COM1	SEG10
3	COM1	SEG9
4	COM1	SEG8
5	COM2	SEG8
6	COM3	SEG8
LCD5-0	COM2	SEG11
1	COM3	SEG12
2	COM2	SEG12
3	COM1	SEG11
4	COM2	SEG10
5	COM3	SEG10
6	COM3	SEG11
LCD6-0	COM2	SEG13
1	COM2	SEG14
2	COM1	SEG14
3	COM1	SEG13
4	COM1	SEG12
5	COM3	SEG13
6	COM3	SEG14
LCD7-0	COM2	SEG16
1	COM3	SEG16
2	COM1	SEG17
3	COM1	SEG16
4	COM1	SEG15
5	COM2	SEG15
6	COM3	SEG15
LCD8-0	COM2	SEG17
1	COM3	SEG17
2	COM2	Not used
3	COM1	Not used

Table 7-2 Master-slice data in EPROM

Bit / Address	Data	7	6	5	4	3	2	1	0
$FF0	0	*	CH2	CH3	CH4	CH5	CH6	CH7	CH8
	1	*	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11
$FF1	0	*	OSC1 XTAL	*	OSC2 Not used	OSC1 Delay Time 0	1/16	1/2	(sec) 1
	1	* *	OSC1 CR	*	OSC2 Used	0	1/16	1/2	1

* : This bit is not used
Note that only one bit of OSC1 Delay Time select bits can be set logical "1".

7.3 Setting the master-slice data

The HD63L05E has two additional registers to be able to specify the master-slice data for SEG11~SEG17(CH2~CH8)and internal oscillator mask options.

During the RES terminal is "Low",address bus(A0~A11:from Port E, Port F) become alternately $FF0 and $FF1. Therefore,the output data from the external EPROM(Address are $FF1 and $FF1) can be written into these registers via Port D. 3cycles are needed for writing the master-slice data into the registers.

7.4 Electrical Characteristics

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3~+5.5	V
Input Voltage	V_{in}	-0.3~V_{CC}+0.3	V
Output Voltage	V_{out}	-0.3~V_{CC}+0.3	V
Operating Temparature	T_{opr}	-20~+75	°C
Storage Temparature	T_{stg}	-55~+125	°C

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded.
Normal operation should be under recommended operating conditions.
If these conditions are exceeded,it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS (Vcc=3.0+0.8v,Vss=0V,Ta=-20 +75°C,typ means typical value at Vcc=3.0V unless otherwise noted.)

● DC CHARACTERISTICS

Item			Symbol	Test Condition	min	typ	max	Unit
Input "High" Level Voltage	XTAL,XIN			Connect C_L=0.5uF to VCH	Vcc-0.3		Vcc	V
	RES,INT,SB		V_{IH}		0.5Vcc+0.9		Vcc	V
	TIMER				0.8Vcc		Vcc	V
	U/M (User Mode)				Vcc-0.2		Vcc	V
	XTAL,XIN			Connect C_L=0.5uF TO VCH	Vss		Vcc-1.8	V
Input "Low" Level Voltage	RES,INT,SB				Vss		0.2Vcc	V
	TIMER		V_{IL}		Vss		0.2Vcc	V
	U/M (Monitor Mode)				Vss		0.2	V
Input Pull-Up Current	RES U/M		$-I_{R1}$	Vcc=3V,Vin=0V	3	15	30	uA
Input Leackage Current	TIMER,SB		I_{IN}	Vin=0V Vcc			1.0	uA
Current Dissipation	Crystal (400kHz)	During System Operation	I_{CC}	f=400kHz,No load. Tested after setting up the internal status.		100	200	uA
		At Halt				40	60	uA
		At Standby				1	5	uA
		At A/D Operation				160	400	uA

● AC CHRACTERISTICS

Item		Symbol	Test Condition	min	typ	max	Unit
Operating Clock Frequency		fcf		100	400	500	kHz
Cycle Time		tcyc		8	10	40	us
External Clock Duty		Duty		45	50	55	%
Oscillation Start Time		t_{OSCf}	C_G=10pF+20%,RS=1k			150	ms
Oscillation Start Time		t_{OSC1}	C_G=10pF+20%,RS=20k			1	s
Internal Capacitance of the Oscillator	EXTAL	C_O			10		pF
	XOUT				10		pF
Delay Time of Oscillation (Program)		t_{PLY}		0		500	ms
Reset Delay Time		t_{PLH}		200			ms
RES Pulse Width		t_{RWL}		tcyc+1			us
INT Pulse Width		t_{IWL}		tcyc+1			us
TIMER Pulse Width		t_{TWL}		tcyc+1			us

● PORT CHARACTERISTICS

Item		Symbol	Test Condition	min	typ	max	Unit
Output "Low" Level Voltage	Port A,B,C	V_{OL}	I_{OL}=100 uA			0.3	V
Input "High" Level Voltage	Port A,B,C	V_{IH}		0.8Vcc		Vcc	V
Input "Low" Level Voltage	Port A,B,C	V_{IL}		Vss		0.2Vcc	V
Input Leackage Current	Port A,B,C	$\|I_{IN}\|$	Vin=0 Vcc			1	uA
Output "Low" Level Voltage	ADCLK,CE,LIR HALT,Port E,F	V_{OL}	I_{OL}=200 uA			0.3	V
Input "High" Level Voltage	Port D	V_{IH}		0.8Vcc		Vcc	V
Input "Low" Level Voltage	Port D	V_{IL}		Vss		0.2Vcc	V

●LCD DRIVER OUTPUT CHARACTERISTICS(V_{CC}=3.0V,V_{SS}=0V,Ta=-20 +75°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Level Voltage	Segment	V_{OH1}	V = 1.00V, V = 2.00V I_{OH}= -1 μA	2.8	-	-	V
		V_{OH2}		1.8	-	-	V
		V_{OH3}		0.8	-	-	V
Output "Low" Level Voltage	Segment	V_{OL1}	V = 1.00V, V = 2.00S I_{OL}= 1 μA	-	-	2.2	V
		V_{OL2}		-	-	1.2	V
		V_{OL3}		-	-	0.2	V
Output "High" Level Voltage	Common	V_{OH1}	V = 1.00V, V = 2.00V I_{OH}= -5 μA	2.8	-	-	V
		V_{OH2}		1.8	-	-	V
		V_{OH3}		0.8	-	-	V
Output "Low" Level Voltage	Common	V_{OL1}	V = 1.00V, V = 2.00V I_{OH}= 5 μA	-	-	2.2	V
		V_{OL2}		-	-	1.2	V
		V_{OL3}		-	-	0.2	V
Dividing Resistor		R_{LCD}	Tested between V and V	45	90	180	k
Output "High" Level Voltage	Segment	V_{OH}	In the case of Output Port, I_{OH}=-30 μA	V_{CC}-0.3	-	-	V
Output "Low" Level Voltage	Segment	V_{OL}	In the case of Output Port, I_{OL}=30 μA	-	-	0.3	V

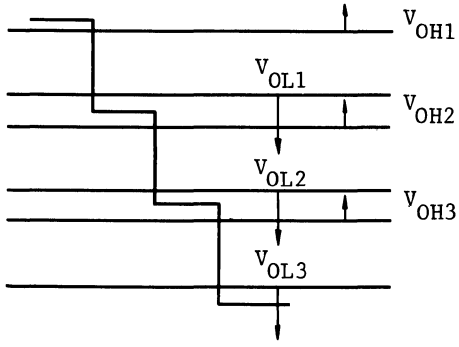


Figure 7-8 Output Level of SEG and COM

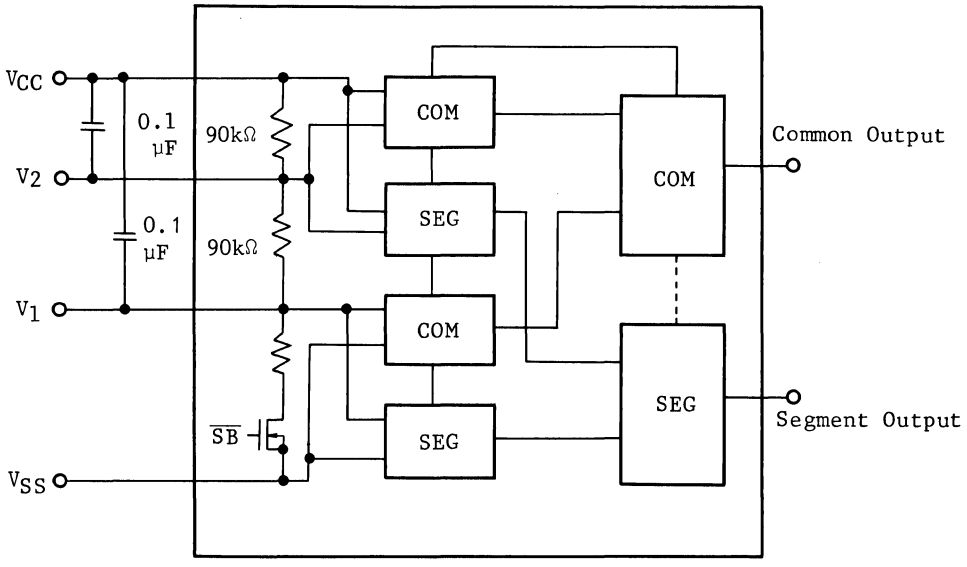


Figure 7-9 Power Supply Circuit for LCD Display

● A/D CONVERTER CHARACTERISTICS *(V_{CC}=3.0V, V_{SS}=0V, Ta=-20°C +75°C, C=300pF, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Conversion Accuracy	Resolution			-	-	8	bit
	Absolute Accuracy		V_{RL}=0.2V < V_{in}<V_{RH}=2.0V	-2	-	+2	LSB
Reference Voltage	"High" Side	V_{RH}		-	-	V_{CC}	V
	"Low" Side	V_{RL}		V_{SS}	-	-	V
	V_{RH} - V_{RL}	V_{REF}		2.0	-	-	V
	Input Range	V_{IN}		V_{RL}	-	V_{RH}	V
	Input Dynamic Range	V_{DYN}		0.2	-	V_{CC}-1.0	V
Ladder Resistor (V_{RH}- V_{RL})		R_{HL}		40	80	160	kΩ
Conversion Time		t_{CNV}		2	-	4	ms
Programmable Voltage Comparison	Judge Error		V_{RL}=0.2V<V_{in} < V_{RH}=2.0V	-4	-	+4	LSB
	Judge Time	t_{CMP}		-	-	60	µs

* These value can be changed without notice, because they are provisional.

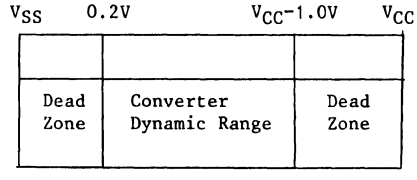


Figure 7-10 Dynamic Range of the Comparator

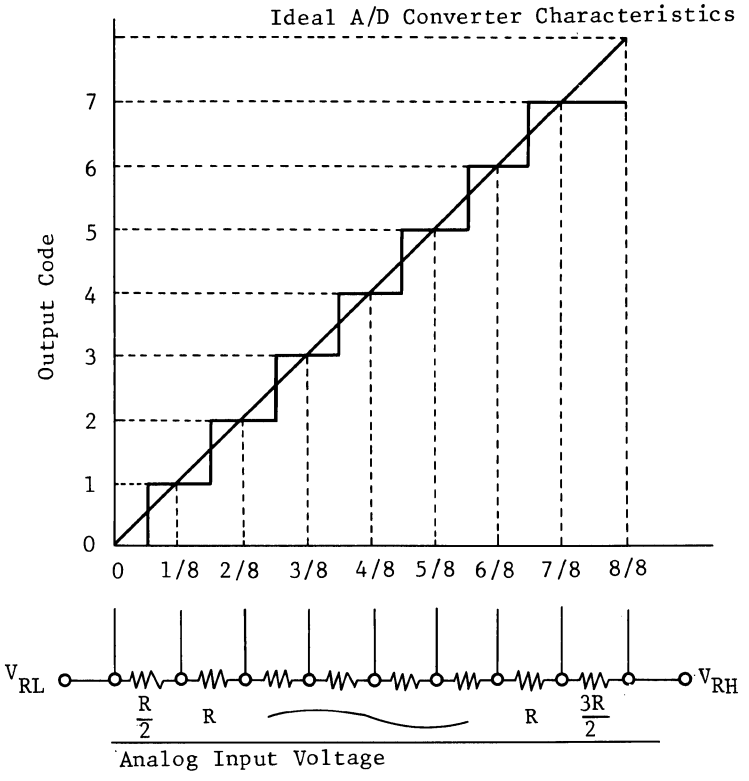


Figure 7-11 Example of 3 bit Resolution

8. ROM Code Order Method

User's programs are mask programmed into ROM by the company and are shipped as LSI. The users are requested to hand in three EPROMs in which the same contents are written,order specifications,mask option list,and list of the ROM contents.

Relationship between the address of the mask ROM and that of the EPROM is shown in Table 8-1. Write $FF for the unused address data of the EPROM.

Table 8-1 Relationship between the Address of Mask ROM and that of EPROM

Type name	Address of Mask ROM	Address of EPROM	Remarks
HD63L05	$080 ~ $F2F	$080 ~ $F2F	User programs are written in this area.
	$F30 ~ $FF3	$F30 ~ $FF3	This area is used for the Self Check program by the company.
	$FF4 ~ $FFF	$FF4 ~ $FFF	Vectors are written into this area.

EPROM is a HN462732 or equivalent product.

Ordering Specifications

(1) Basic Item (Please fill in blank space or mark applicable item with symbol ☑)

LSI Family		Out Line	☐ DP-28 ☐ DP-40 ☐ DP-42 ☐ FP-54 ☐ FP-80
Application		Remarks	
Customer ROM Code ID.			
ROM Code Media	☐Paper tape ☐EPROM		

(2) Environmental Check List

LSI Ambient Temperature	nominal		°C	Target Level of Reliability	☐ 500 fit ☐ 1000 fit ☐( )
	range	°C-	°C		
LSI Ambient Humidity	nominal		%	Acceptable Quality Level	☐ 1.0% ☐ 0.65% ☐ 0.4% ☐ ( )
	range	%-	%		
Power-ON Duration		hour/day typ.		Remarks	
Maximum Applied Voltage to LSI			V		

(3) Electrical Characteristics

☐ Purchasing Specifications _____	☐ Hitachi's Standerd Specifications Refer to Data Sheet _____

Please fill in the space enclosed with ☐

Date of Order _____

Customer _____

Dept. _____

Accepted by

_____

ROM Code Verification

LSI Type No.	
Shipping Date of ROM Code to Customers	
Approved Date of ROM Code from Customers	

Mask Option List

Select one from each item and check ☑ .

Date of Order_____
Customer _____
Dept. _____
Accepted by
_____
ROM Code ID._____
LSI Type No._____

(1) Mask Option

Item	Option		☐ Check	Remarks
Selection of Oscillator1	Crystal		☐ OSCX	
	Resistor		☐ OSCR	
Selection of Oscillator	With 32.768 kHz Crystal		☐ Yes	
			☐ No	1/3 of cycle clock is provided
Delay time of OSC1	No Delay time		☐ No.	
	1/256 seconds		☐ Yes	
	1/16 seconds		☐ Yes	
	1/2 seconds		☐ Yes	
Configuration of SEG1∿SEG17	Segment	Static	☐ Yes	
		1/3 Bias 1/3 Duty	☐ Yes	
	Output Port		☐ Yes	

(2) I/O Option

PIN	I/O	Mask Option						Remarks	Pin	I/O	Mask Option				Remarks
		A	B	C	D	E	F				G	H	J	K	
A0	I/O					*	*		SEG1	O	*			*	
A1	I/O					*	*		SEG2	O	*			*	
A2	I/O					*	*		SEG3	O	*			*	
A3	I/O					*	*		SEG4	O	*			*	
A4	I/O					*	*		SEG5	O	*			*	
A5	I/O					*	*		SEG6	O	*			*	
A6	I/O					*	*		SEG7	O	*			*	
A7	I/O					*	*		SEG8	O	*			*	
B0	I/O					*	*		SEG9	O	*			*	
B1	I/O					*	*		SEG10	O	*			*	
B2	I/O					*	*		SEG11	O	*			*	
B3	I/O					*	*		SEG12	O	*			*	
B4	I/O					*	*		SEG13/CH6	I/O				*	
B5	I/O					*	*		SEG14/CH5	I/O				*	
B6	I/O					*	*		SEG15/CH4	I/O				*	
B7	I/O					*	*		SEG16/CH3	I/O				*	
C0	I/O					*	*		SEG17/CH2	I/O				*	
C1	I/O					*	*		V1/CH7	I/O		*			
C2	I/O					*	*		V2/CH8	I/O		*			
C3	I/O					*	*								
$\overline{INT}$	I	*	*	*	*										

Note

A : CMOS Output without Input pull-up PMOS
B : CMOS Output with Input pull-up PMOS
C : CMOS Output for Key scanning
D : NMOS Open-drain Output
E : Input without pull-up PMOS
F : Input with pull-up PMOS
G : A/D Input
H : Segment Output
J : Port Output (SEG1∿SEG17, V1, and V2)
K : Terminals for LCD Display

(3) LCD Pin Location

		Timing			SEGMENT OUTPUT TERMINAL																
Bit		COM 1	COM 2	COM 3	SEG 1	SEG 2	SEG 3	SEG 4	SEG 5	SEG 6	SEG 7	SEG 8	SEG 9	SEG 10	SEG 11	SEG 12	SEG 13	SEG 14	SEG 15	SEG 16	SEG 17
LCD1	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
	7																				
LCD2	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
	7																				
LCD3	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD4	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD5	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD6	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD7	0																				
	1																				
	2																				
	3																				
	4																				
	5																				
	6																				
LCD8	0																				
	1																				
	2																				
	3																				
ϕWRITE		o																			

Note
Select Display Timing and SEGMENT Terminal for each bit of LCD1~LCD8.

In case of Static Driving or Output Port, the Timing is fixed at COM1.

ϕWRITE Clock is generated when data is written into the LCD1.

Hitachi, Ltd.
6-2, Otemachi 2-Chome, Chiyoda-ku
Tokyo 100
Telephone : Tokyo(270)2111
Cable Address : "HITACHY" TOKYO
Telex : J22395,22432,24491,26375

For inquiry write to :
SAN JOSE
Hitachi America, Ltd.
San Jose Office
1800 Bering Drive,
San Jose, California 95112
Telephone : (408)292-6404
-6405

LONDON
Hitachi Electronic Components
(U.K.) Ltd.
221-225 Station Road
Harrow,
Middlesex, HAL 2XL
England
Telephone : 01-861-1414
Telex : 936293(HITEC-G)

MÜNCHEN
Hitachi Electronic Components
Europe GmbH
Hans-Pinsel-Str.
Haar München
West Germany
Telephone : 089-46140
Telex : 05-22-593

HONG KONG
Hitachi Semiconductor
(Hong Kong) Ltd.
Room 706 - 707, 7/F.,
Wing On Plaza,
Salisbury Road, Tsimshatsu,
Kowloon, Hong Kong.
Telephone : 3 - 7219218 - 9
Telex : 40815 HISAL HX

H3L5MCU-EM