MOTOROLA
intelligence everywhere™

digital dna™

Building an USB-DDC ICP Interface Board Hardware

Designer Reference Manual

M68HC08
Microcontrollers

DRM041/D
Rev. 0, 6/2003

MOTOROLA.COM/SEMICONDUCTORS

Building an USB-DDC ICP Interface Board Hardware Reference Design

Designer Reference Manual — Rev 0

by: Ernest Chan
Motorola Ltd
Hong Kong

Table of Contents

Section 1. General Description

1.1 Introduction 7

1.2 Overview 7

Section 2. ICP Hardware and Firmware Consideration

2.1 Overall Setup 9

2.2 ICP board hardware 10

2.3 Target application hardware configuration 11

2.4 Firmware Consideration on Target MCU 12

2.5 Communication between ICP board and target MCU 16

Section 3. Operation for the DDC ICP

3.1 System Requirement for the Host Computer 17

3.2 Control Software Installation Procedure 17

3.3 Hardware Connection for ICP 18

3.4 Run the Control software 19

3.5 DDC ICP demonstration using two ICP boards 22

Section 4. Schematic

4.1 DDC ICP Board Schematic 25

Section 5. Other Information

5.1 Related Documents 27

Table of Contents

5.2 Programming Time 27
5.3 Test Description 27

Section 6. Glossary

Section 1. General Description

1.1 Introduction

The MC68HC908LD64 is a member of the low cost high performance M68HC Family MCU specially designed for digital monitor applications. The MC68HC908LD64 is capable of In-Circuit Programming (ICP) through its DDC12AB(DDC) module.

For full MC68HC908LD64 specification, please refer to the MC68HC908LD64 data sheet.

1.2 Overview

This document provides the reference design on building an USB-DDC ICP interface board hardware and the target MCU firmware for implementing the DDC ICP.

The ICP function is desirable for development and production. It allows the MC68HC908LD64 to be programmed with final code at the latest production stage. As DDC signal lines are inside the VGA port, it makes DDC ICP possible even when the whole case of the monitor set has been assembled.

Features:

- Simple connection to target application
- Monitor ROM routines handle major ICP process that reduces target firmware overhead
- PC Control Software and ICP board firmware available

General Description

Section 2. ICP Hardware and Firmware Consideration

2.1 Overall Setup

There is a host computer where the control software will be installed. The computer also holds the s-record file that is to be programmed to the flash of the target MCU.

Between the computer and the targeted application (monitor in the usual case), there is an USB-DDC ICP interface board (hereafter referred as ICP board).

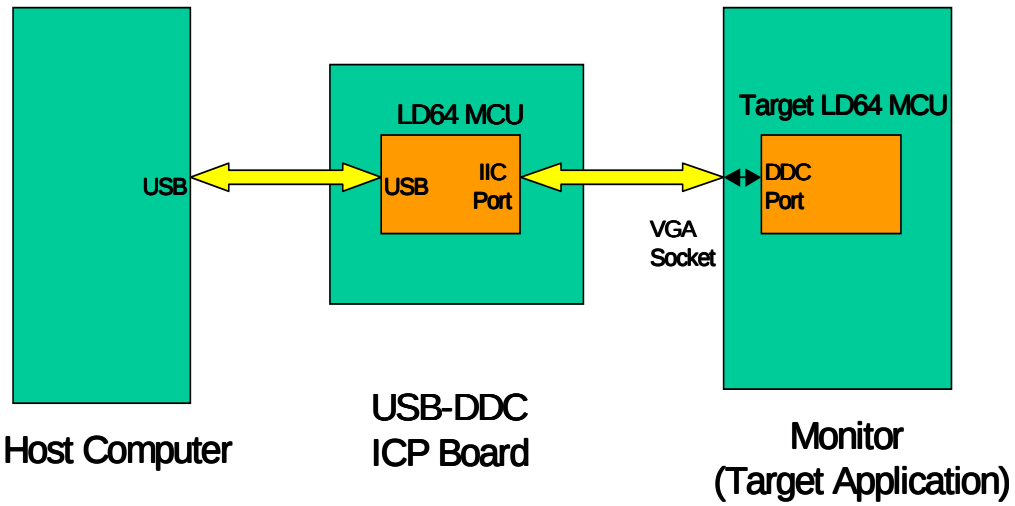


Figure 2-1. Block diagram showing the implementation of DDC ICP

Both the host control software and the ICP board are universal for different target applications.

After building the ICP board hardware and properly configuring the target applications, users can implement the ICP using the available PC control software and ICP board firmware. Please refer to the system

requirement for the host computer to run the control software in Section 3.

2.2 ICP board hardware

The ICP board includes an MC68HC908LD64 MCU. The USB module and the IIC module are the major operation units. This MCU should operate at 6MHz internal bus.

The USB upstream port is to be connected to the host computer USB port and the DPLUS0 and DMINUS0 pins external connection have to be properly configured as shown in the reference ICP board. With the driver and control software installed, the computer can recognize the board as a dedicate ICP device.

The ICP board MCU uses the IIC interface to communicate with the target MCU's DDC12AB interface. The DDC12AB interface is fully compatible with IIC protocol. The IIC port is at +5V input rating even though LD64 is a +3.3V device. It is for the convenience of interfacing to external +5V system in monitor application. For examples, the HSYNC,VSYNC and DDC12AB are also +5V input rating. The IIC SDA and SCL pull high resistors are connected to the positive terminal of supply connector CON7. There are small filtering capacitors at the IIC SCL and IIC SDC lines. The values of these capacitors could be fine tune according to the actual application.

The LEDs in ICP board are for indicating different status of the ICP progress.

The ICP board can also function as a target MCU board for demonstration of the DDC ICP. The jumper setting of J1-J4 can be adjusted for such purpose. Please refer to section DDC ICP demonstration using two ICP boards for more details.

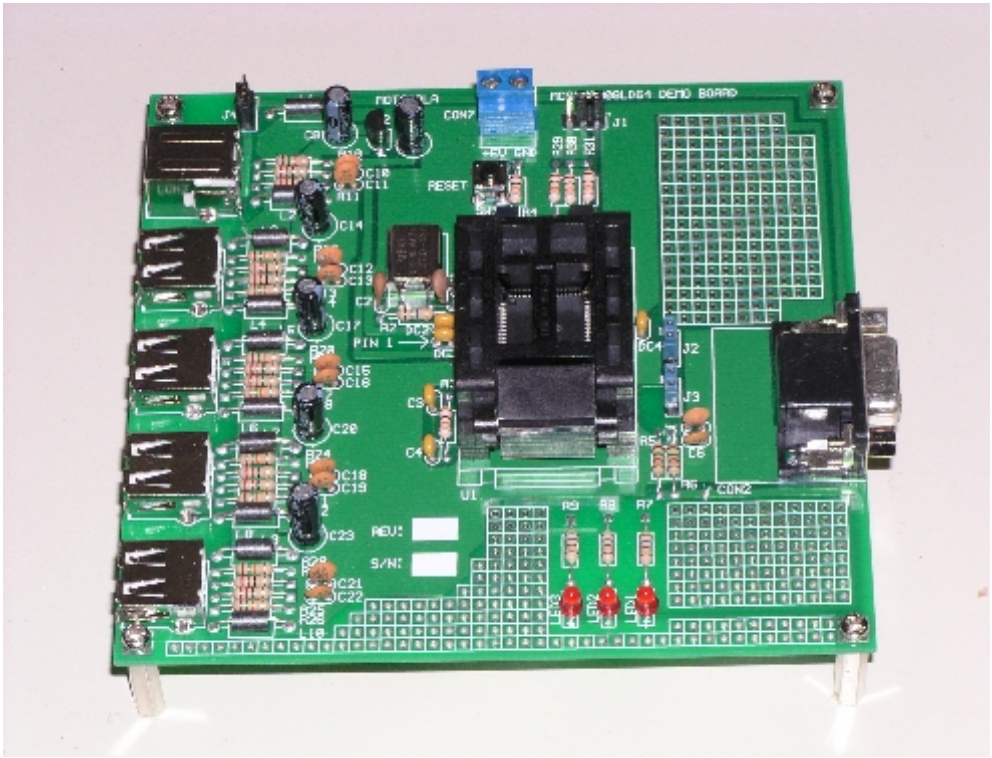


Figure 2-2. USB-DDC ICP board

2.3 Target application hardware configuration

The BUS frequency for the MC68HC908LD64 should be operating at 6MHz. It is also the typical BUS frequency for MC68HC908LD64 in monitor application.

The DDC ICP can be done when the target MCU is operating at normal user mode or monitor mode. Users can refer to the MC68HC908LD64 device datasheet on the monitor mode entry requirement. But in case for DDC ICP, the internal BUS speed should be 6MHz even in monitor mode. In case for normal user mode DDC ICP, some firmware requirement on target MCU is necessary. Please refer to Section **DDC ICP at user mode application.**

2.4 Firmware Consideration on Target MCU

The DDC ICP make use of the some built-in routines inside the target MC68HC908LD64. These routines are resident inside the monitor ROM area. Once the DDC ICP link has been built up between the ICP board and the target MC68HC908Ld64 MCU, the host computer software will take the control of the ICP process.

2.4.1 DDC ICP at user mode application

The original firmware on the target MC68HC908LD64 need to initiate the ICP communication.

Usually, this can be done by adding a DDC ICP detection routine at the main entry of the user firmware. This routine is responsible to check if there is any ICP board connection with its DDC channel. If an ICP board has been detected during power up, the program will jump to DDC ICP entry point within the on-chip resident ROM. The address of DDC ICP entry point is $FADC.

The corresponding ROM routines and the firmware from ICP board MCU will communicate to each other. All the later ICP process is then handled by the host computer control software, ICP board MCU firmware and those resident ROM routines in the target MCU.

After the ICP process has been completed, the target can be power down. The target MCU is ready for application according to the new firmware in the next power on cycle.

In ordinary application, there is no ICP board connected. After power up, the DDC ICP detection routine determines no DDC ICP connection. It will pass the control to the application firmware to start the application.

In order that DDC ICP can be taken place in later firmware upgrade, the same detection routine has to be installed in all application firmware.

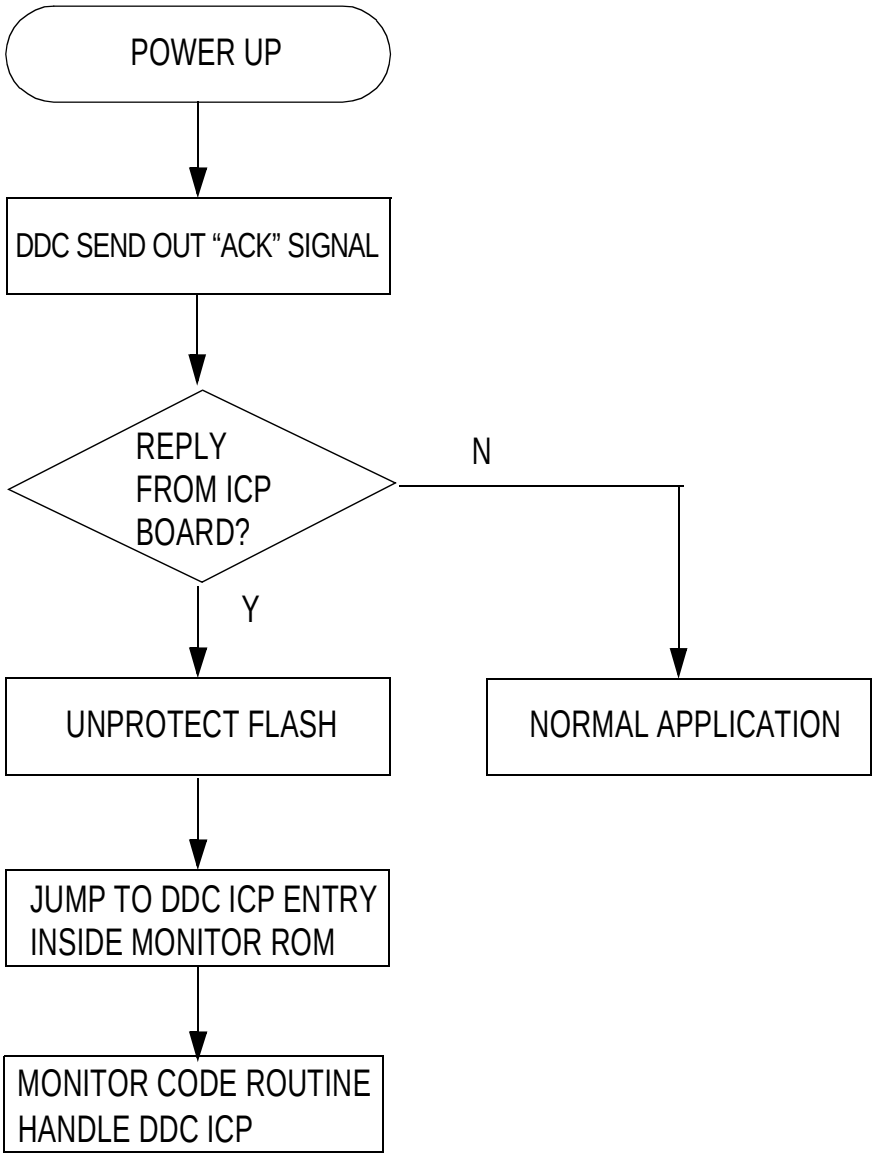


Figure 2-3. The flow for the DDC ICP detection in target MCU

```
*****************************************************************************
* MC68HC908LD64 DDC ICP detection and execution
*****************************************************************************
ACK_SIGNAL          equ         $AF
ICP_ADDRESS         equ         $36
MON_ICP_ON          equ         $FADC
DDCDSR              equ         $19
RXBFB               equ         0
TXBEB               equ         1
TXIFB               equ         6
RXIFB               equ         7
MASTB               equ         4
BBB                 equ         5
NAKIFB              equ         6
ALIFB               equ         7
DDCENB              equ         7
DMCR                equ         $16
DADR                equ         $17
DDCDCR              equ         $18
DDTR                equ         $1A
D2ADR               equ         $1C
CONFIG              equ         $1F
FLBPR               equ         $FE08
FLBPR1              equ         $FE0B
PDCR                equ         $69


;----------- Reset entry point ----------------------------------
RESET_INIT:
            SEI                             ; disable all interrupts
            LDHX    #$480                   ; set SP to bottom of RAM
            TXS                             ; and reserve stack for ICP

ICP_FUNC:
            BCLR    RXIFB,DDCDSR            ; init. DDC
            BCLR    TXIFB,DDCDSR            ;
            CLR     DMCR                    ;
            CLR     DADR                    ;
            CLR     D2ADR                   ;
            CLR     DDCDCR                  ;
            LDA     #$30                    ; enable DDC SDA, SCL pins
            STA     PDCR                    ;
            LDA     #ACK_SIGNAL             ;
            BSR     ACK_ICPMCU              ; Send ACK to ICP source
            BCS     ICP_EXIT                ; if link fail  --> go normal application
ICP_GO:
            BCLR    TXIFB,DDCDSR            ; if link ok  --> go ICP
            BCLR    RXIFB,DDCDSR            ;
            MOV     #$03,CONFIG             ; Disable COP, Enable STOP cmd
            LDA     #$00                    ; 0000
            STA     FLBPR                   ; unprotect entire flash
            LDA     #$00                    ; 0000
            STA     FLBPR1                  ; unprotect entire flash
            JMP     MON_ICP_ON              ; jsr to ICP routine (monitor ROM)
```

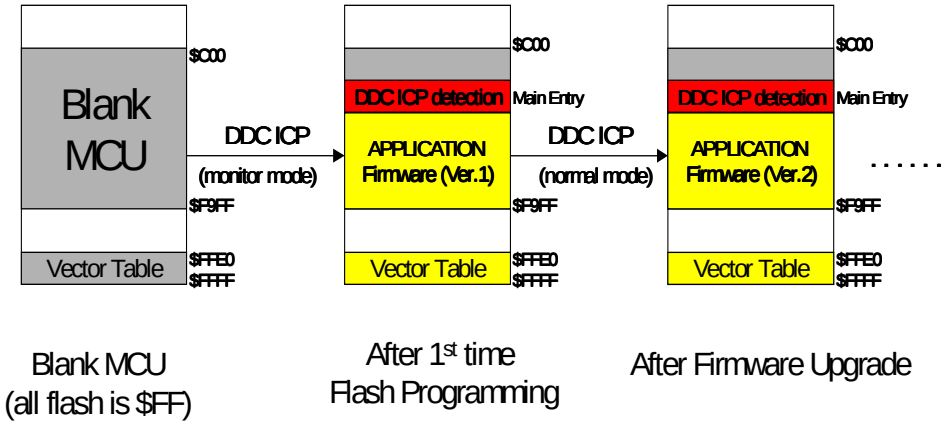
```
;========== Master send ACK through DDC ============================
ACK_ICPMCU:
            BSET    DDCENB,DDCDCR           ; DDC enable
            MOV     #ICP_ADDRESS,DADR       ;
            BSET    MASTB,DMCR              ; master mode
            STA     DDTR                    ; send ACC
WAIT_000:
            BRSET   ALIFB,DMCR,ARBI_LOST    ;
            BRSET   NAKIFB,DMCR,ARBI_LOST   ; C=1 --> error
            BRCLR   TXBEB,DDCDSR,WAIT_000   ;
            BRCLR   BBB,DMCR,ARBI_LOST      ;
            BCLR    MASTB,DMCR              ; slave mode
            MOV     #$FF,DDTR               ;
            CLC                             ; C=0 --> ok
            RTS
ARBI_LOST:
            SEC                             ;
            RTS                             ;


ICP_EXIT:
            LDA     #$40                    ; 4000-ffff
            STA     FLBPR                   ; protect entire flash
            LDA     #$0C                    ; 0C00-3fff
            STA     FLBPR1                  ; protect entire flash

;==========START OF APPLICATION=====================================
;Application firmware may start here
;
```

Figure 2-4. USB DDC ICP detection routine

2.4.2 DDC ICP at monitor mode (BLANK MCU)

If the target board MC68HC908LD64 is power up with monitor mode entered, the ICP link between target MC68HC908LD64 and ICP will be established through monitor code routine automatically. DDC ICP can be taken place even if the DDC ICP detection routine has been not been installed in the target MCU.

Usually, the ICP through monitor mode can be used as the first time flash programming for the blank MC68HC908LD64. It is because the blank MC68HC908LD64 contains no DDC ICP detection routine inside its flash. However, as the reset vector is blank in this case, the MC68HC908LD64 will automatically enter monitor mode once it has been powered up for the ICP process.



2.5 Communication between ICP board and target MCU

There are several basic flash programming routines in the MC68HC908LD64 monitor ROM. The routines are used for flash program, block erase, mass erase and code verification. A command value is assigned to each of the basic routine. The ICP board MCU can call those basic routines for operation through valid command strings.

The table below shows some commands and address of the monitor code ICP routines:

Constant name	Hex Value
Block Program	55
Block Erase	AA
Mass Erase	A5
Block Verify	5A
ACK	AF
NAK	5F
Target MCU address	34
DDC ICP board MCU address	36
DDC ICP entry address in monitor ROM	FACD

Section 3. Operation for the DDC ICP

3.1 System Requirement for the Host Computer

- PC with one USB port
- Windows 98 / ME / 2000
- Microsoft Visual Studio 6.0 installed
- Intel UHCI USB chip set compatible

3.2 Control Software Installation Procedure

1. Plug the “USB-ICP board” to the host computer.
2. The system will detect a new device, then follow the message, click “Next”.
3. Select “Search for the best driver for your device”, then click “Next”.
4. Specify the location of USB-ICP driver, then click “Next”.
5. Windows will detect “Motorola USB-->DDC ICP Device”, click “Next”.
6. Driver installation is completed, click “Finish”.
7. Create an folder in the PC and copy “ICPexe.exe” and “ISPdll.dll” to that folder.
8. Create an folder in the PC and copy all the personality file(.imp files) to that folder.
9. The ICP program, “ICPexe.exe”, is ready for execution if the ICP hardware connection has been setup.

3.3 Hardware Connection for ICP

1. Set the “USB-ICP board” to the correct jumper setting:

Jumper	Position	Remarks
J1	1-2 short	ICP board MCU operate in normal user mode
J2	2-3 short	IICSDA connected to CON2
J3	2-3 short	IICSCL connected to CON2
J4	1-2-3 short	ICP board run on USB BUS power. External supply at CON7 is NOT required. IIC pull high resistors connected to USB BUS power.

2. Power up the ICP board by connecting a USB cable from CON1 to the PC USB port. The host will recognize the ICP board and the board will run on USB bus power.
3. The LED1 should be flashing.
4. Connect DDC(VGA) cable from the target application to “USB-ISP board” CON2.
5. Power on the target board. LED3 should turn on indicating ICP board MCU receive the “ACK” signal from target MCU.

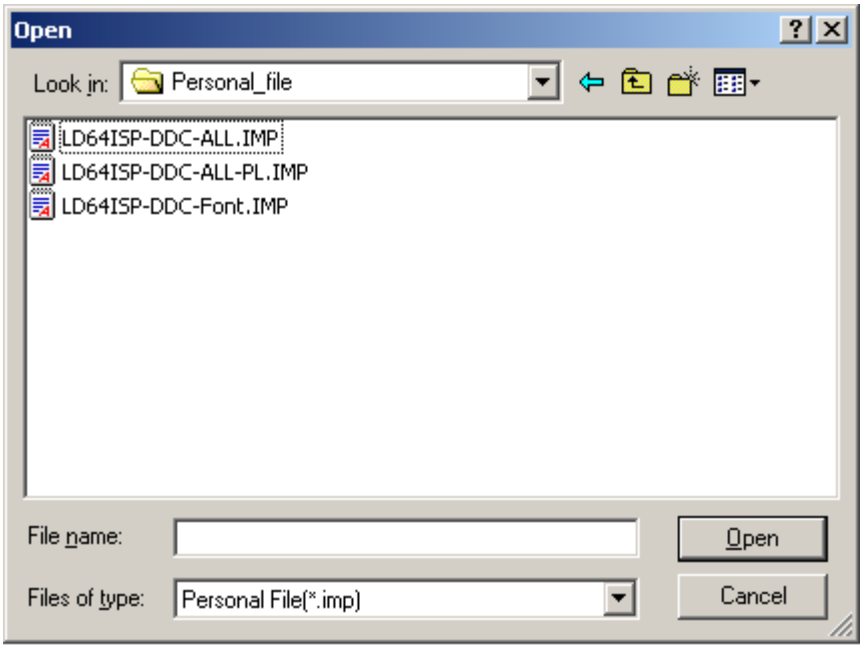
NOTE: 1.The LEDs’ light up sequence should be correct before running the control software.

2. If LED3 turns on at step 4 before the target board supply has been applied, one possible reason is the target MCU has momentary got the power through the DDC link (target MCU’s DDC pins may have pull up resistors connected to its VDD). Try setting J4 jumper to 2-3 position only and repeat the setup.

3. User need to restart the steps from 2 to 6 again when there is incorrect step and the ICP cannot start up properly.

3.4 Run the Control software

- Double click the “ICPexe.exe” to start the program
- A pop-up menu will appear asking for Personality file



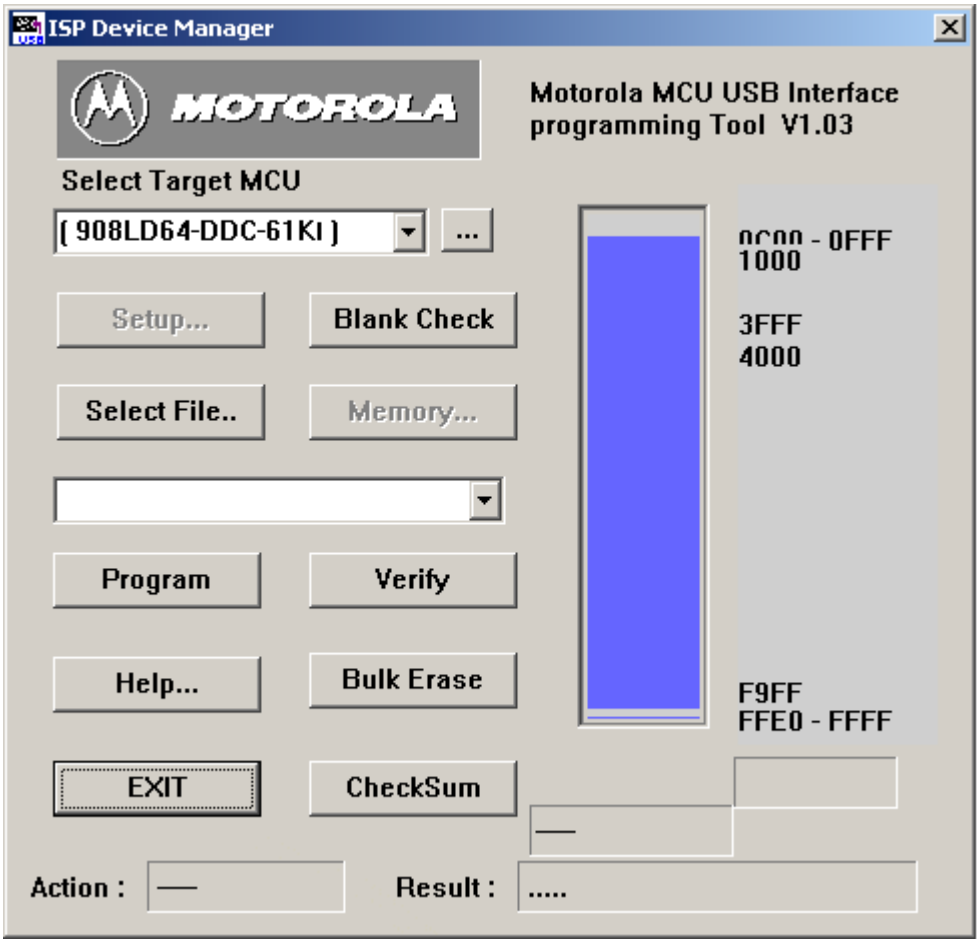
- Select the personality file according to the desired operation:

Personality File	Usage
LD64ISP-DDC-ALL.IMP	Program any part of the Flash area
LD64ISP-DDC-ALL-PL.IMP	Program any part of the flash area and verify the contents
LD64ISP-DDC-Font.IMP	Only program s-record for the OSD font flash area

- After select the personality file, the main screen of the ICP program will appear. Select the s-record file to be programmed by clicking the “Select File” button

Operation for the DDC ICP

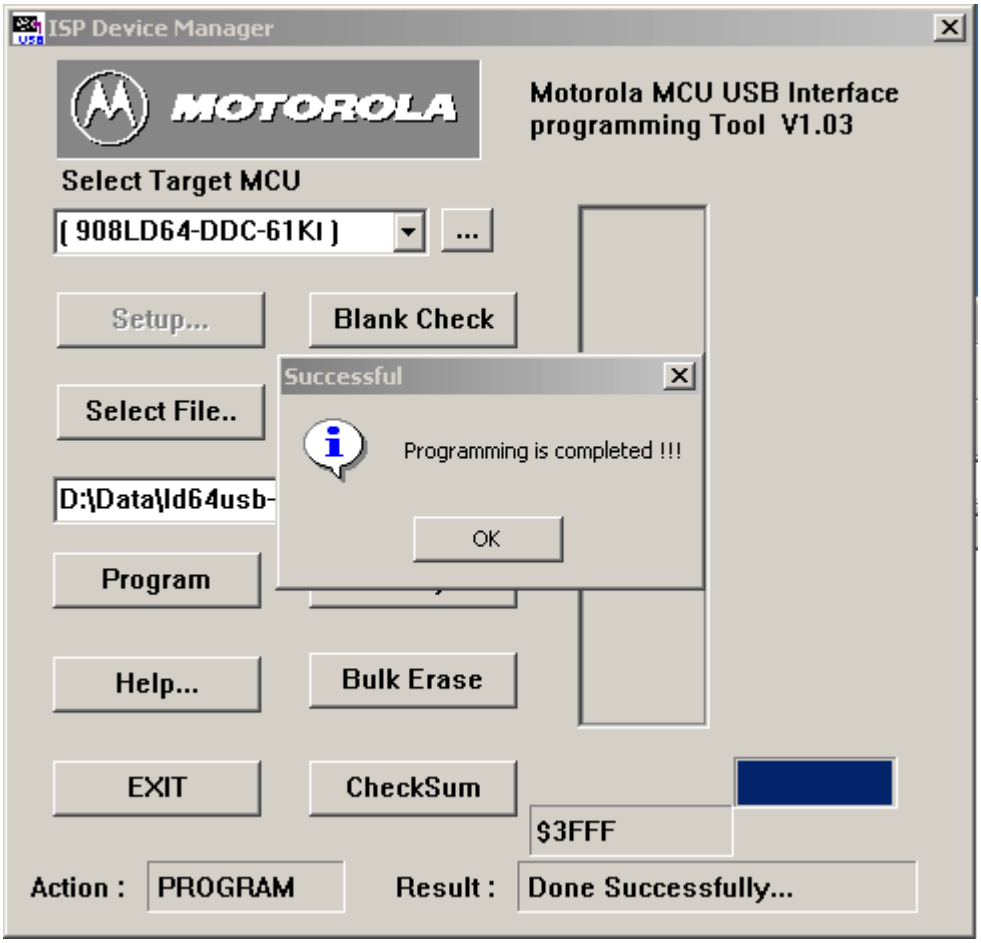
-



- If personality file of “LD64ISP-DDC-ALL.IMP” or “LD64ISP-DDC-ALL-PL.IMP” is selected, users have to click the “Bulk Erase” button to mass erase the MCU before programming the s-record

NOTE: It is possible to program the OSD font flash area($1000-$3FFF) s-record without first mass erase the target MCU if this area is originally blank. Do not place the DDC ICP detection routine inside OSD font flash area in this case. It is to ensure the DDC ICP can be conducted again if there is any programming error such as power failure.

- Press “Program” button will start the flash programming
- A pop-up menu will appear after the flash programming has been finished

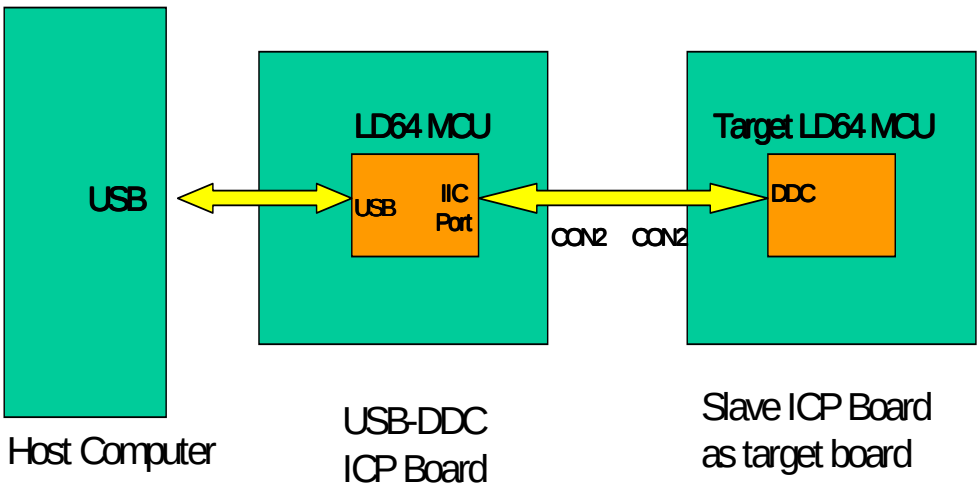


- The target can then be power down and disconnected from the USB-DDC ICP for normal operation

Operation for the DDC ICP

3.5 DDC ICP demonstration using two ICP boards

The ICP board can also function as a target board for demonstration of the DDC ICP. The setup is illustrated below:



The jumper setting in the master ICP board is similar in normal DDC ICP, except the J4 setting, please refer to the below table:

Table 3-1. Master ICP board jumper setting

Jumper	Position	Remarks
J1	1-2 short	ICP board MCU operate in normal user mode
J2	2-3 short	IICSDA connected to CON2
J3	2-3 short	IICSCL connected to CON2
J4	2-3 short	ICP board run on USB BUS power. External supply at CON7 is NOT required. IIC pull high resistor pull up by slave board.

The jumper setting in the target board will be different from the master ICP board. If the target board MC68HC908LD64 MCU firmware has the DDC ICP detection routine installed, the jumper setting is:.

Table 3-2. Target board Jumper Setting:MCU with DDC ICP detection routine

Jumper	Position	Remarks
J1	1-2	Target MCU operates in normal user mode IRQ=+3.3V
J2	1-2	DDCSDA connected to CON2
J3	1-2	DDCSCL connected to CON2
J4	1-2	Target board Power come from CON7

The setup also supports programming the target MCU without the DDC ICP routine installed. It can be done by configuring the target ICP board MCU to operate in monitor mode. The corresponding jumper setting is:**.**

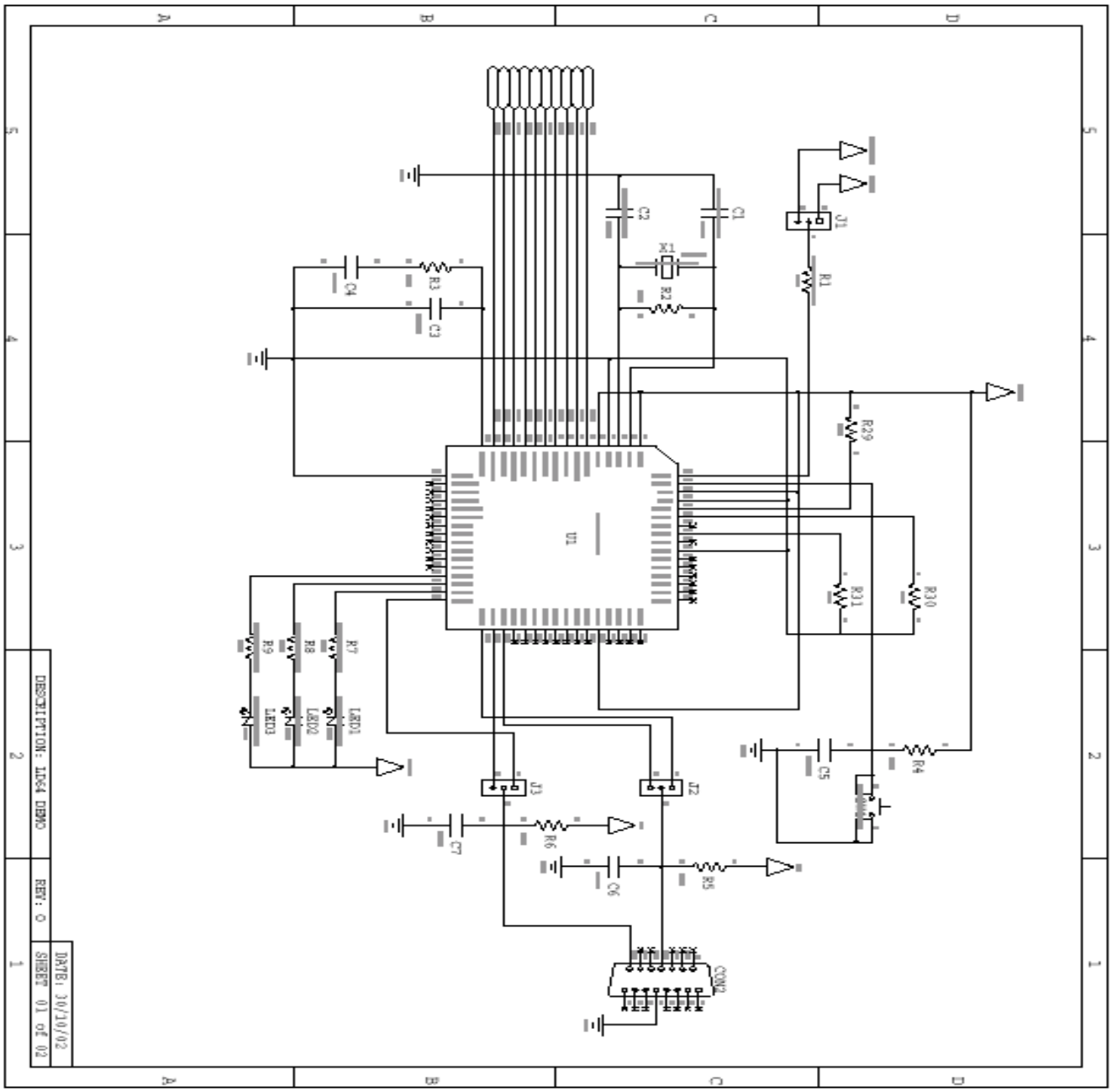
Table 3-3. Target board Jumper Setting:MCU without DDC ICP detection

Jumper	Position	Remarks
J1	2-3	Target MCU operates in monitor mode IRQ=+5V
J2	1-2	DDCSDA connected to CON2
J3	1-2	DDCSCL connected to CON2
J4	1-2	Target board Power come from CON7

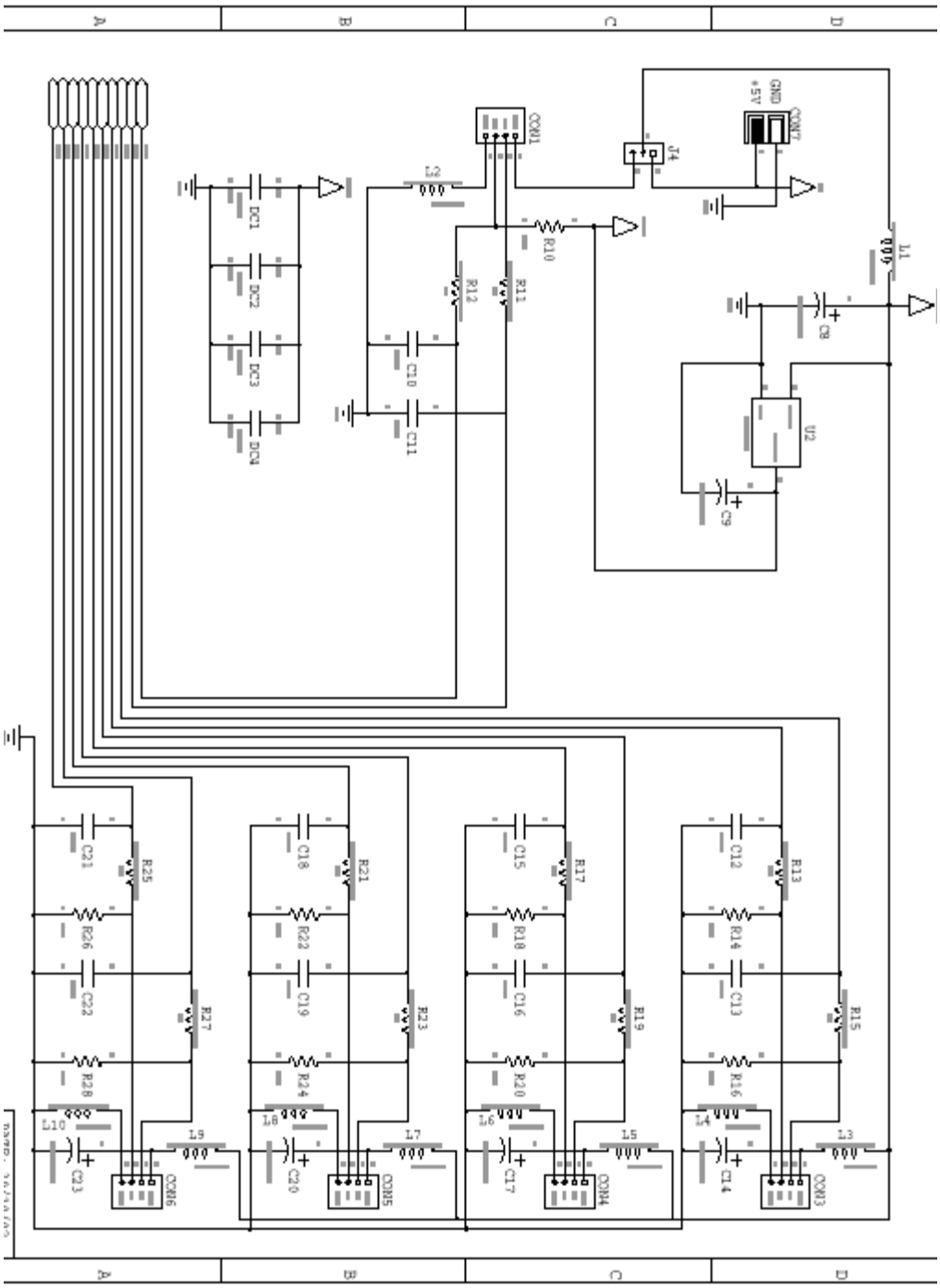
Operation for the DDC ICP

Section 4. Schematic

4.1 DDC ICP Board Schematic



Schematic



Section 5. Other Information

5.1 Related Documents

MC68HC908LD64 Technical Data

5.2 Programming Time

Programming speed may depend on the host computer configuration. Programming time for the whole flash is about 70 seconds measured a at PC with 1.8GHz Pentitum IV CPU and 384M RAM memory.

5.3 Test Description

The flash contents of MC68HC908LD64 MCU programmed by DDC ICP have been counter verified by MCUscribe Programmer software.

Other Information

Section 6. Glossary

A — See “accumulator (A).”

accumulator (A) — An 8-bit general-purpose register in the CPU08. The CPU08 uses the accumulator to hold operands and results of arithmetic and logic operations.

acquisition mode — A mode of PLL operation during startup before the PLL locks on a frequency. Also see “tracking mode”.

address bus — The set of wires that the CPU or DMA uses to read and write memory locations.

addressing mode — The way that the CPU determines the operand address for an instruction. The M68HC08 CPU has 16 addressing modes.

ALU — See “arithmetic logic unit (ALU).”

arithmetic logic unit (ALU) — The portion of the CPU that contains the logic circuitry to perform arithmetic, logic, and manipulation operations on operands.

asynchronous — Refers to logic circuits and operations that are not synchronized by a common reference signal.

baud rate — The total number of bits transmitted per unit of time.

BCD — See “binary-coded decimal (BCD).”

binary — Relating to the base 2 number system.

binary number system — The base 2 number system, having two digits, 0 and 1. Binary arithmetic is convenient in digital circuit design because digital circuits have two permissible voltage levels, low and high. The binary digits 0 and 1 can be interpreted to correspond to the two digital voltage levels.

binary-coded decimal (BCD) — A notation that uses 4-bit binary numbers to represent the 10 decimal digits and that retains the same positional structure of a decimal number. For example,

234 (decimal) = 0010 0011 0100 (BCD)

bit — A binary digit. A bit has a value of either logic 0 or logic 1.

Glossary

branch instruction — An instruction that causes the CPU to continue processing at a memory location other than the next sequential address.

break module — A module in the M68HC08 Family. The break module allows software to halt program execution at a programmable point in order to enter a background routine.

breakpoint — A number written into the break address registers of the break module. When a number appears on the internal address bus that is the same as the number in the break address registers, the CPU executes the software interrupt instruction (SWI).

break interrupt — A software interrupt caused by the appearance on the internal address bus of the same value that is written in the break address registers.

bus — A set of wires that transfers logic signals.

bus clock — The bus clock is derived from the CGMOUT output from the CGM. The bus clock frequency, f_{op}, is equal to the frequency of the oscillator output, CGMXCLK, divided by four.

byte — A set of eight bits.

C — The carry/borrow bit in the condition code register. The CPU08 sets the carry/borrow bit when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some logical operations and data manipulation instructions also clear or set the carry/borrow bit (as in bit test and branch instructions and shifts and rotates).

CCR — See “condition code register.”

central processor unit (CPU) — The primary functioning unit of any computer system. The CPU controls the execution of instructions.

CGM — See “clock generator module (CGM).”

clear — To change a bit from logic 1 to logic 0; the opposite of set.

clock — A square wave signal used to synchronize events in a computer.

clock generator module (CGM) — A module in the M68HC08 Family. The CGM generates a base clock signal from which the system clocks are derived. The CGM may include a crystal oscillator circuit and or phase-locked loop (PLL) circuit.

comparator — A device that compares the magnitude of two inputs. A digital comparator defines the equality or relative differences between two binary numbers.

computer operating properly module (COP) — A counter module in the M68HC08 Family that resets the MCU if allowed to overflow.

condition code register (CCR) — An 8-bit register in the CPU08 that contains the interrupt mask bit and five bits that indicate the results of the instruction just executed.

control bit — One bit of a register manipulated by software to control the operation of the module.

control unit — One of two major units of the CPU. The control unit contains logic functions that synchronize the machine and direct various operations. The control unit decodes instructions and generates the internal control signals that perform the requested operations. The outputs of the control unit drive the execution unit, which contains the arithmetic logic unit (ALU), CPU registers, and bus interface.

COP — See “computer operating properly module (COP).”

counter clock — The input clock to the TIM counter. This clock is the output of the TIM prescaler.

CPU — See “central processor unit (CPU).”

CPU08 — The central processor unit of the M68HC08 Family.

CPU clock — The CPU clock is derived from the CGMOUT output from the CGM. The CPU clock frequency is equal to the frequency of the oscillator output, CGMXCLK, divided by four.

CPU cycles — A CPU cycle is one period of the internal bus clock, normally derived by dividing a crystal oscillator source by two or more so the high and low times will be equal. The length of time required to execute an instruction is measured in CPU clock cycles.

CPU registers — Memory locations that are wired directly into the CPU logic instead of being part of the addressable memory map. The CPU always has direct access to the information in these registers. The CPU registers in an M68HC08 are:

- A (8-bit accumulator)
- H:X (16-bit index register)
- SP (16-bit stack pointer)
- PC (16-bit program counter)
- CCR (condition code register containing the V, H, I, N, Z, and C bits)

CSIC — customer-specified integrated circuit

cycle time — The period of the operating frequency: $t_{CYC} = 1/f_{OP}$.

decimal number system — Base 10 numbering system that uses the digits zero through nine.

Glossary

direct memory access module (DMA) — A M68HC08 Family module that can perform data transfers between any two CPU-addressable locations without CPU intervention. For transmitting or receiving blocks of data to or from peripherals, DMA transfers are faster and more code-efficient than CPU interrupts.

DMA — See “direct memory access module (DMA).”

DMA service request — A signal from a peripheral to the DMA module that enables the DMA module to transfer data.

duty cycle — A ratio of the amount of time the signal is on versus the time it is off. Duty cycle is usually represented by a percentage.

EEPROM — Electrically erasable, programmable, read-only memory. A nonvolatile type of memory that can be electrically reprogrammed.

EPROM — Erasable, programmable, read-only memory. A nonvolatile type of memory that can be erased by exposure to an ultraviolet light source and then reprogrammed.

exception — An event such as an interrupt or a reset that stops the sequential execution of the instructions in the main program.

external interrupt module (IRQ) — A module in the M68HC08 Family with both dedicated external interrupt pins and port pins that can be enabled as interrupt pins.

fetch — To copy data from a memory location into the accumulator.

firmware — Instructions and data programmed into nonvolatile memory.

free-running counter — A device that counts from zero to a predetermined number, then rolls over to zero and begins counting again.

full-duplex transmission — Communication on a channel in which data can be sent and received simultaneously.

H — The upper byte of the 16-bit index register (H:X) in the CPU08.

H — The half-carry bit in the condition code register of the CPU08. This bit indicates a carry from the low-order four bits of the accumulator value to the high-order four bits. The half-carry bit is required for binary-coded decimal arithmetic operations. The decimal adjust accumulator (DAA) instruction uses the state of the H and C bits to determine the appropriate correction factor.

hexadecimal — Base 16 numbering system that uses the digits 0 through 9 and the letters A through F.

high byte — The most significant eight bits of a word.

illegal address — An address not within the memory map

illegal opcode — A nonexistent opcode.

I — The interrupt mask bit in the condition code register of the CPU08. When I is set, all interrupts are disabled.

index register (H:X) — A 16-bit register in the CPU08. The upper byte of H:X is called H. The lower byte is called X. In the indexed addressing modes, the CPU uses the contents of H:X to determine the effective address of the operand. H:X can also serve as a temporary data storage location.

input/output (I/O) — Input/output interfaces between a computer system and the external world. A CPU reads an input to sense the level of an external signal and writes to an output to change the level on an external signal.

instructions — Operations that a CPU can perform. Instructions are expressed by programmers as assembly language mnemonics. A CPU interprets an opcode and its associated operand(s) and instruction.

interrupt — A temporary break in the sequential execution of a program to respond to signals from peripheral devices by executing a subroutine.

interrupt request — A signal from a peripheral to the CPU intended to cause the CPU to execute a subroutine.

I/O — See “input/output (I/0).”

IRQ — See “external interrupt module (IRQ).”

jitter — Short-term signal instability.

latch — A circuit that retains the voltage level (logic 1 or logic 0) written to it for as long as power is applied to the circuit.

latency — The time lag between instruction completion and data movement.

least significant bit (LSB) — The rightmost digit of a binary number.

logic 1 — A voltage level approximately equal to the input power voltage (V_{DD}).

logic 0 — A voltage level approximately equal to the ground voltage (V_{SS}).

low byte — The least significant eight bits of a word.

low voltage inhibit module (LVI) — A module in the M68HC08 Family that monitors power supply voltage.

Glossary

LVI — See “low voltage inhibit module (LVI).”

M68HC08 — A Motorola family of 8-bit MCUs.

mark/space — The logic 1/logic 0 convention used in formatting data in serial communication.

mask — 1. A logic circuit that forces a bit or group of bits to a desired state. 2. A photomask used in integrated circuit fabrication to transfer an image onto silicon.

mask option — A optional microcontroller feature that the customer chooses to enable or disable.

mask option register (MOR) — An EPROM location containing bits that enable or disable certain MCU features.

MCU — Microcontroller unit. See “microcontroller.”

memory location — Each M68HC08 memory location holds one byte of data and has a unique address. To store information in a memory location, the CPU places the address of the location on the address bus, the data information on the data bus, and asserts the write signal. To read information from a memory location, the CPU places the address of the location on the address bus and asserts the read signal. In response to the read signal, the selected memory location places its data onto the data bus.

memory map — A pictorial representation of all memory locations in a computer system.

microcontroller — Microcontroller unit (MCU). A complete computer system, including a CPU, memory, a clock oscillator, and input/output (I/O) on a single integrated circuit.

modulo counter — A counter that can be programmed to count to any number from zero to its maximum possible modulus.

monitor ROM — A section of ROM that can execute commands from a host computer for testing purposes.

MOR — See “mask option register (MOR).”

most significant bit (MSB) — The leftmost digit of a binary number.

multiplexer — A device that can select one of a number of inputs and pass the logic level of that input on to the output.

N — The negative bit in the condition code register of the CPU08. The CPU sets the negative bit when an arithmetic operation, logical operation, or data manipulation produces a negative result.

nibble — A set of four bits (half of a byte).

object code — The output from an assembler or compiler that is itself executable machine code, or is suitable for processing to produce executable machine code.

opcode — A binary code that instructs the CPU to perform an operation.

open-drain — An output that has no pullup transistor. An external pullup device can be connected to the power supply to provide the logic 1 output voltage.

operand — Data on which an operation is performed. Usually a statement consists of an operator and an operand. For example, the operator may be an add instruction, and the operand may be the quantity to be added.

oscillator — A circuit that produces a constant frequency square wave that is used by the computer as a timing and sequencing reference.

OTPROM — One-time programmable read-only memory. A nonvolatile type of memory that cannot be reprogrammed.

overflow — A quantity that is too large to be contained in one byte or one word.

page zero — The first 256 bytes of memory (addresses $0000–$00FF).

parity — An error-checking scheme that counts the number of logic 1s in each byte transmitted. In a system that uses odd parity, every byte is expected to have an odd number of logic 1s. In an even parity system, every byte should have an even number of logic 1s. In the transmitter, a parity generator appends an extra bit to each byte to make the number of logic 1s odd for odd parity or even for even parity. A parity checker in the receiver counts the number of logic 1s in each byte. The parity checker generates an error signal if it finds a byte with an incorrect number of logic 1s.

PC — See “program counter (PC).”

peripheral — A circuit not under direct CPU control.

phase-locked loop (PLL) — A oscillator circuit in which the frequency of the oscillator is synchronized to a reference signal.

PLL — See “phase-locked loop (PLL).”

pointer — Pointer register. An index register is sometimes called a pointer register because its contents are used in the calculation of the address of an operand, and therefore points to the operand.

polarity — The two opposite logic levels, logic 1 and logic 0, which correspond to two different voltage levels, V_{DD} and V_{SS}.

polling — Periodically reading a status bit to monitor the condition of a peripheral device.

Glossary

port — A set of wires for communicating with off-chip devices.

prescaler — A circuit that generates an output signal related to the input signal by a fractional scale factor such as 1/2, 1/8, 1/10 etc.

program — A set of computer instructions that cause a computer to perform a desired operation or operations.

program counter (PC) — A 16-bit register in the CPU08. The PC register holds the address of the next instruction or operand that the CPU will use.

pull — An instruction that copies into the accumulator the contents of a stack RAM location. The stack RAM address is in the stack pointer.

pullup — A transistor in the output of a logic gate that connects the output to the logic 1 voltage of the power supply.

pulse-width — The amount of time a signal is on as opposed to being in its off state.

pulse-width modulation (PWM) — Controlled variation (modulation) of the pulse width of a signal with a constant frequency.

push — An instruction that copies the contents of the accumulator to the stack RAM. The stack RAM address is in the stack pointer.

PWM period — The time required for one complete cycle of a PWM waveform.

RAM — Random access memory. All RAM locations can be read or written by the CPU. The contents of a RAM memory location remain valid until the CPU writes a different value or until power is turned off.

RC circuit — A circuit consisting of capacitors and resistors having a defined time constant.

read — To copy the contents of a memory location to the accumulator.

register — A circuit that stores a group of bits.

reserved memory location — A memory location that is used only in special factory test modes. Writing to a reserved location has no effect. Reading a reserved location returns an unpredictable value.

reset — To force a device to a known condition.

ROM — Read-only memory. A type of memory that can be read but cannot be changed (written). The contents of ROM must be specified before manufacturing the MCU.

SCI — See “serial communication interface module (SCI).”

serial — Pertaining to sequential transmission over a single line.

serial communications interface module (SCI) — A module in the M68HC08 Family that supports asynchronous communication.

serial peripheral interface module (SPI) — A module in the M68HC08 Family that supports synchronous communication.

set — To change a bit from logic 0 to logic 1; opposite of clear.

shift register — A chain of circuits that can retain the logic levels (logic 1 or logic 0) written to them and that can shift the logic levels to the right or left through adjacent circuits in the chain.

signed — A binary number notation that accommodates both positive and negative numbers. The most significant bit is used to indicate whether the number is positive or negative, normally logic 0 for positive and logic 1 for negative. The other seven bits indicate the magnitude of the number.

software — Instructions and data that control the operation of a microcontroller.

software interrupt (SWI) — An instruction that causes an interrupt and its associated vector fetch.

SPI — See “serial peripheral interface module (SPI).”

stack — A portion of RAM reserved for storage of CPU register contents and subroutine return addresses.

stack pointer (SP) — A 16-bit register in the CPU08 containing the address of the next available storage location on the stack.

start bit — A bit that signals the beginning of an asynchronous serial transmission.

status bit — A register bit that indicates the condition of a device.

stop bit — A bit that signals the end of an asynchronous serial transmission.

subroutine — A sequence of instructions to be used more than once in the course of a program. The last instruction in a subroutine is a return from subroutine (RTS) instruction. At each place in the main program where the subroutine instructions are needed, a jump or branch to subroutine (JSR or BSR) instruction is used to call the subroutine. The CPU leaves the flow of the main program to execute the instructions in the subroutine. When the RTS instruction is executed, the CPU returns to the main program where it left off.

synchronous — Refers to logic circuits and operations that are synchronized by a common reference signal.

TIM — See “timer interface module (TIM).”

Glossary

timer interface module (TIM) — A module used to relate events in a system to a point in time.

timer — A module used to relate events in a system to a point in time.

toggle — To change the state of an output from a logic 0 to a logic 1 or from a logic 1 to a logic 0.

tracking mode — Mode of low-jitter PLL operation during which the PLL is locked on a frequency. Also see “acquisition mode.”

two’s complement — A means of performing binary subtraction using addition techniques. The most significant bit of a two’s complement number indicates the sign of the number (1 indicates negative). The two’s complement negative of a number is obtained by inverting each bit in the number and then adding 1 to the result.

unbuffered — Utilizes only one register for data; new data overwrites current data.

unimplemented memory location — A memory location that is not used. Writing to an unimplemented location has no effect. Reading an unimplemented location returns an unpredictable value. Executing an opcode at an unimplemented location causes an illegal address reset.

V —The overflow bit in the condition code register of the CPU08. The CPU08 sets the V bit when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow bit.

variable — A value that changes during the course of program execution.

VCO — See “voltage-controlled oscillator.”

vector — A memory location that contains the address of the beginning of a subroutine written to service an interrupt or reset.

voltage-controlled oscillator (VCO) — A circuit that produces an oscillating output signal of a frequency that is controlled by a dc voltage applied to a control input.

waveform — A graphical representation in which the amplitude of a wave is plotted against time.

wired-OR — Connection of circuit outputs so that if any output is high, the connection point is high.

word — A set of two bytes (16 bits).

write — The transfer of a byte of data from the CPU to a memory location.

X — The lower byte of the index register (H:X) in the CPU08.

Z — The zero bit in the condition code register of the CPU08. The CPU08 sets the zero bit when an arithmetic operation, logical operation, or data manipulation produces a result of $00.

HOW TO REACH US:

DRM041/D

USA/EUROPE/LOCATIONS NOT LISTED:
Motorola Literature Distribution
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

JAPAN:
Motorola Japan Ltd.
SPS, Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

ASIA/PACIFIC:
Motorola Semiconductors H.K. Ltd.
Silicon Harbour Centre
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

HOME PAGE:
http://motorola.com/semiconductors

MC68HC908LD64
Rev. 0
6/2003