

Chapter 2
Architecture of 89C52
Microcontroller

CHAPTER 2**ARCHITECTURE OF 89C52 MICROCONTROLLER**

2.1 Introduction:

Single chip microcomputers, also known as Microcontrollers, are used primarily to perform dedicated functions. They are used as independent controllers in machines or as slaves in distributed processing. Generally these include all the necessary hardware of a computer on a single chip as ALU, RAM, ROM, Timers, Interrupts and I/O lines (serial and parallel). A variety of single-chip Microcontrollers are available in the market to meet diversified industrial needs. The details of few products with features are presented in the appendix

2.2 Processor Architectures

There are two types of processor architectures⁸

- I. Harvard – Princeton architecture,
- II. CISC – RISC architecture.

Harvard architecture: This had a design that used separate memory banks for program storage, processor stack and RAM. This architecture was ignored until the late 70's till the manufacturers realized the potential advantages that the design had. The Harvard architecture executes instructions in fewer instruction cycles than the

Von Neumann architecture⁸. This is because of instruction parallelism is possible in the Harvard architecture. The block diagram of the architecture is shown in figure 2.1

Parallelism means that fetches for the next instruction can take place during the execution of the current instruction, without having to either wait for either a “dead” cycle of the instruction’s execution or stop the processor’s operation with the next instruction is being fetched. Most of the microcontroller based circuits are extensively used for process control applications only⁹.

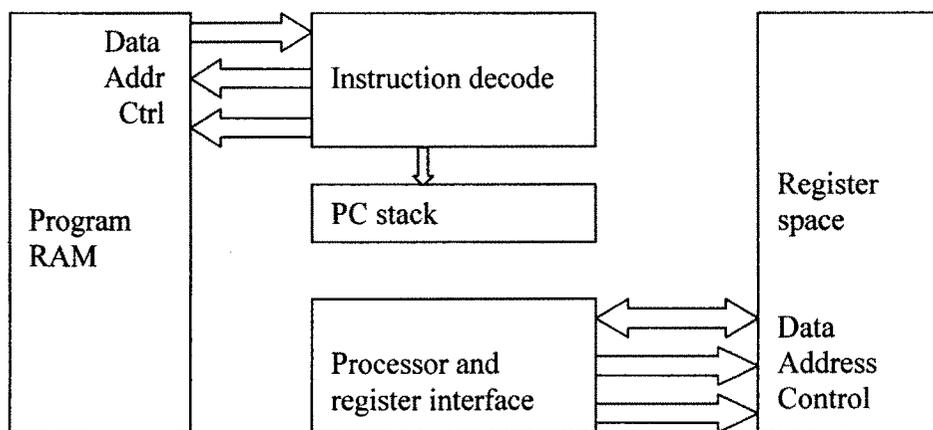


Fig 2.1 Harvard Architecture Block Diagram

Princeton Architecture: This had a common memory for storing (the control) program as well as variables and other data structures, also known as Von-Neumann Architecture. In many Princeton architected processors, when the execution of instructions are going on the processor time can also be used to fetch the next instruction. This is known as “pre-fetching”. Its advantage is also that it simplifies the Microcontroller chip design because only one memory is accessed always. The block diagram of the architecture is shown in figure 2.2. In the present study the

microcontroller 89c52 used for the development of capacitance meter and the details of the microcontroller is given below.

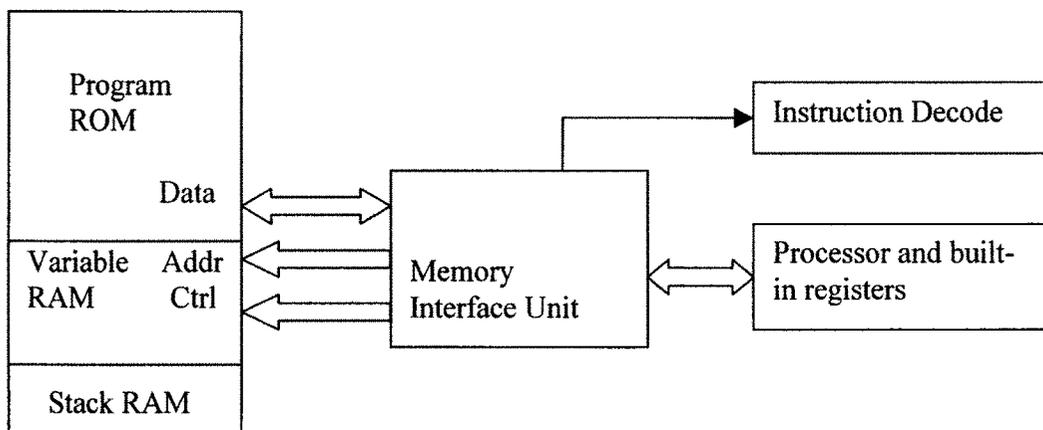


Fig 2.2 Princeton Architecture Block Diagram

2.3 8052 Architecture:

The internal logic design of a device is called its architecture¹⁰. The generic 8052 architecture supports a Harvard architecture, which contains two separate buses for both program and data. So, it has two distinctive memory spaces of 64K * 8 size for program and 64K * 8 for data. The on-chip flash allows the program memory to be reprogrammed in-system or by a conventional non-volatile memory programmer¹¹. The ALU is an 8 bit accumulator and another 8 bit B register as main processing blocks. Other portions of the architecture include few 8 bit and 16 bit registers and 8 bit memory locations. The operations that do not use the internal 128 bytes from 00H to 7FH are done by group of specific internal registers¹²

The base architecture supports on chip peripheral functions. The microcontroller has four ports, labeled Port 0,1,2 & 3¹³, timers/counters, interrupts and a serial communication port.

Features of AT89C52:

- 8K Bytes of In-System Reprogrammable

Flash Memory

-Endurance: 1,000 Write/Erase Cycles

- Fully Static Operation: 0 Hz to 24 MHz
- Three-level Program Memory Lock
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low-power Idle and Power-down Modes

Instruction Set

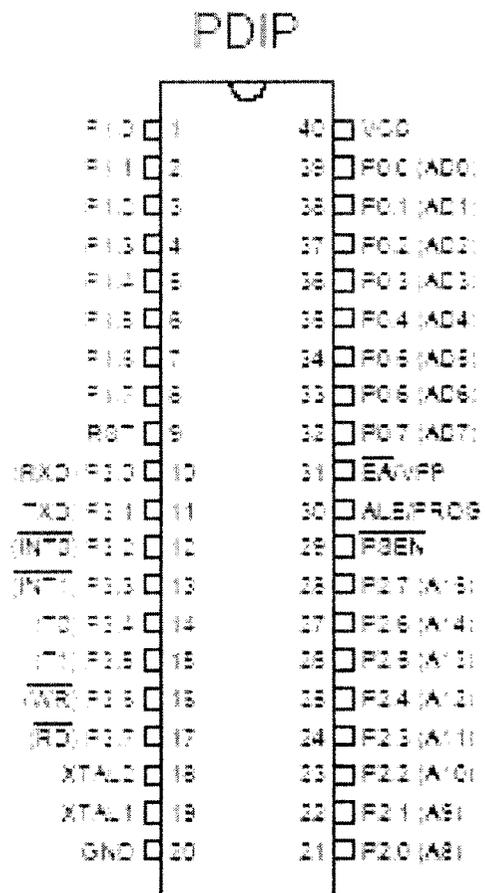


Figure 2.3 Pin diagram of 89c52

The 89c52 Microcontroller supports a powerful and versatile instruction set that enables the user to develop a compact program¹⁴. The instruction set provides extensive support for direct bit manipulation in control and logical applications¹⁵. The instruction set consists of the following type of instructions:

i. Data Transfer Instructions

Transferring the data byte from source to destination is of different types. The instruction, which do not disturb the Accumulator, which do the data transfer

MOV <DES> <SOU> ex: MOV R0, #10

The stack resides in on chip ROM, which increments upwards. The PUSH instruction increments the stack pointer and the POP instruction decrements the stack pointer.

PUSH <SOU> POP <DEST>

The data transfer instructions include a 16 bit MOV that can be used to initialize the data pointer for lookup tables in program memory or for 16 bit external data memory accesses.

MOV DPTR, #DATA

The following instruction has the capability to exchange a complete byte (8 bits) from source to accumulator

XCH A, <DATA>

ii. Data Transfer in External Ram

Data from external source like RAM is accessed with this type of addressing mode. Only indirect addressing can be used¹⁶. This depends on weather the address is of 8 bits long or 16 bits long. In accessing external Data Memory¹⁷ by 8-bit addresses port P0 serves as multiplexed address/data bus to the RAM and 3 lines of port 2 are

used to page the RAM. In this type of mode the accumulator is either source or destination of data.

MOV A, @Ri MOVX A, @DPTR

iii. Lookup Tables

The lookup tables are in program memory; hence these are only read but not updated. The MOVC instruction of the table can read a byte from 256 entries, numbered 0 through 255. The number of the desired entry loaded into the accumulator and data pointer is set up to point to beginning of the table.

MOV A, @A+DPTR

Another instruction works with program counter. Hence pc acts as the table base and the accumulator should carry the table entry value.

MOVC A, @A+PC

iv. Arithmetic Instructions

The arithmetic operation in basic math's are all accommodated like addition, subtraction, multiplication and division; apart from these increment, decrement, decimal adjust operations are also possible

ADD A, <DATA> SUBB A, <DATA>

MUL AB DIV AB

INC A DEC A DA A

v. Logical Instructions

Complete logical operations like AND, OR, NOT, EX-OR and few additional instructions to rotate left or right 1 bit in the accumulator and even to swap nibble in the accumulator are possible with the instructions, which allow logical manipulation. All these operations are on a bit by bit basis. Boolean operations can be performed on any byte in the lower 128 internal data memory space or the SFR space using direct addressing without having to use the accumulator.

ANL A,<DATA.> ORL A,<DATA>

CPL A XRL A,<DATA>

vi. Program Control Instructions – Call, Return

LJUMP encodes a 16-bit address in the instruction bytes. The destination may be anywhere in the 64K byte program memory address space.

AJUMP instruction encodes its destination using a 11-bit address which is embedded in the instruction itself. Address bits 10 through 8 form a 3-bit field in the opcode and address bits 7 through 0 form a second byte. Address bits 15 through 11 remain unchanged from the incremented contents of the PC, so AJUMP can only be used when the destination is known to be within the same 2K-memory block.

vii. Jump Instructions

There are few instructions, which do unconditional jumping in the program. Instruction, which jumps depending upon certain case, is also available. The destination address is calculated in runtime as the sum of the 16 bit DPTR register and

be treated as a separate single bit port. This range of bit operations are not easily obtained in other architectures with any amount of byte oriented software.

Boolean instructions even support good number of jumping instructions. The destination address for these jumps is specified in the second byte of the instruction. This is a signed (two's complement) offset byte that is added to the PC if the jump is executed. The range of jump is therefore -128 to $+127$ program memory bytes relative to the first byte following the jump instruction.

ANL C, <BIT>

CLR C

SETB C

JC <REL>

2.4 Timing of Microcontroller

Almost all Microcontrollers have an on-chip oscillator, which can be used as the clock source for the CPU. To use the oscillator, connect a crystal or ceramic resonator between the XTAL1 and XTAL2 pins of the Microcontroller, and connect the capacitors to ground.

Time for ON state and one OFF state is the oscillator periods (clock cycle) of the crystal which can be seen by using a CRO. The internal clock generator defines the sequence of states that make up the Microcontroller machine cycle. A machine cycle consists of a sequence of 6 states. Each state time lasts for two oscillator periods. Thus, a machine cycle lasts 12 oscillator periods. If you use a 12 MHz crystal then one machine cycle is $1 \mu\text{s}$.

2.5 Timers and Counters

89S52 family Microcontroller is equipped with up to two inputs, which may be used as general-purpose interrupts. A typical device provides a total of 5 interrupt sources. Timer

0 and Timer 1 generate vectored interrupts, as does the Serial Port. Applications that require more than two externally signaled vectored interrupts, and do not use one or more of the counters or the serial port, can be configured to use these facilities for additional external interrupt inputs. Minimum response time is a goal for this configuration. Another popular method to implement extra interrupt inputs is to poll under software control a port pin configured as an input. This method is necessary when the on-chip peripherals are in use. Applications where this approach is recommended are ones in which the processor spends more than half of the time executing a “wait loop,” or a short code sequence which jumps or branches back on itself without performing any functions. In this case, the instructions that will check the state of input used as an interrupt source are inserted into this sequence. Consequently, this input is ignored when other routines are being executed. This input may have to be latched externally, or the processor may miss the signal while executing other routines. Dedicated interrupt inputs that vector the processor to individual service do not have the drawbacks of the method described above.

The standard 8052 core has two timer/counters, each of which is full 16 bits. Each timer/counter can be function as a free running timer or to count falling edges on the signal applied to their respective I/O pin. When used as a counter, as incoming

signal is sampled at every instruction the input signal must have a frequency equal to or less than the instruction cycle frequency divided by 2.

The TCON SFR has dramatically increased the functionality of the timer/counter. The functions include start or stop the timers, hold the overflow flags, the software can freeze the operation of either timer as well as restart the timers simply by changing a bit in TCON register, the timers can indicate the overflow of the register content by setting respective flags, this bit can be used to interrupt the controller. Therefore the interrupt can be prevented or can be forced by the software.

Special purpose register is used to configure the timers by altering the value in the TMOD (timer mode), by this we can control the mode of two timers and the source they use to count, source can be hardware or software. This register can be thought as two nibbles, the upper nibble is for timer 1 and the lower nibble is for timer 0.

Timer 1				Timer 0			
Gate	C/T	M1	M0	Gate	C/T	M1	M0

GATE: Software control can be gained by setting this bit to 0, now the timer will count any time its TRx bit in the TCON register is high. In the hardware control mode, both the TRx bit and the INTx pin on the chip must be high for the timer to count. When a low is detected at the INTx pin, the timer will stop. This is useful for measuring pulse widths of signals on the INTx pin.

C/T: Altering this bit can set the source for the timer. 1 for counting the pulses on the I/O pin associated to it, 0 for counting the internal clock cycles.

M1 & M0: Both bits are used to select the mode of timer and counter. The 8052 support 4 modes, mode 0 thru mode 3. The mode of operation are explain below:

Mode 0 and Mode 1: in mode 0 the timer/counter will behave like a 13 bit counter. When the counter overflows, the TF0 or TF1 bit in the TCON SFR is set. This will cause the appropriate timer interrupt, if it is enabled. In mode 1 every thing works the same as timer 0 with the exception all the 16 bits are used instead of 13 bits. This mode is used for counting explicitly and interrupting when triggered.

Mode 2: The timer is set up as an eight bit counter, in 16 bits lower 8 bits are used for counting and the remaining higher 8 bits are used to store the count value for automatic reloading when overflow is detected. This mode is especially used in baud rate generation.

Mode 3: the timer 0 becomes two eight-bit counters, which are implemented in TH0 and TL0. The counter implemented in TL0 maintains control of all the timer 0 flags, but the counter in TH0 takes over the control flags in TCON from timer 1. This means that the timer 1 can no longer force interrupts, however, it can be used for any purpose which will not require the overflow interrupt such as a baud rate generator or as timer/counter which is polled by the software. This mode is very much useful when an application must use a UART mode and also requires two timer/counters.

2.6 Serial Port

The serial port can be placed in mode 2, which is a 9-bit UART with the baud rate derived from the oscillator. The external interrupt is signaled through this port on the RxD receive data pin. Reception is initiated by a detected 1-to-0 transition at RxD.

The signal must stay at 0 for at least five-eighths of a bit period for this level to be recognized.

Refer to the description of baud rates to determine the length of a bit period at the oscillator frequency selected for the application. The input signal should remain low for at least one bit period and for not more than 9 bit periods. To prepare the serial port for use as an external interrupt, the following bits must be set up:

In SCON:

SM0 = 1, SM1 = 0, SM2 = 0 and REN = 1

2.7 Derivatives -- Device Selection

The released Microcontrollers after the success of 89c52 have special features apart from supporting core 89c52. This is mandatory to gain access to specific area of the Microcontroller usage. Different companies started their own Microcontrollers, which have special features on each of them. Here are some special features of 89c52 core derivatives.

i ANALOG DEVICES - AduC812

The features include analog I/O 8-Channel¹⁸, High Accuracy 12-Bit ADC On-Chip, 40 ppm/_C Voltage Reference High Speed 200 kSPS, DMA Controller for High Speed ADC-to-RAM Capture, Two 12-Bit Voltage Output DACs On-Chip Temperature Sensor Function. 8K Bytes On-Chip Flash/EE Program Memory, 640 Bytes On-Chip Flash/EE Data Memory, On-Chip Charge Pump (No Ext. VPP Requirements), 256 Bytes On-Chip Data RAM, 16M Bytes External Data Address Space, 64K Bytes External Program Address Space.

89c52-core 12 MHz Nominal Operation (16 MHz Max), Three 16-Bit Timer/Counters, 32 Programmable I/O lines High Current Drive Capability—Port 3, Nine Interrupt Sources, Two Priority Levels POWER Specified for 3 V and 5 V Operation, Normal, Idle and Power-Down Modes Peripherals include UART Serial I/O 2-Wire (I2C®-Compatible) and SPI® Serial I/O Watchdog Timer.

ii ATMEL – AT89C52

Most popular and widely used in many applications Microcontroller features include Compatible with MCS-51™ Products, 4K Bytes of In-System Reprogrammable Flash Memory (Endurance: 1,000 Write/Erase Cycles), Fully Static Operation: 0 Hz to 24 MHz

Three-Level Program Memory Lock, 128 x 8-Bit Internal RAM, 32 Programmable I/O Lines, Two 16-Bit Timer/Counters, Six Interrupt Sources, Programmable Serial Channel Low Power Idle and Power Down Modes.

iii WINBOND – W77E58

The features of winbond w77e58 which is a 89c52 core derivate are 8 – bit CMOS, high speed architecture of 4 clocks/machine cycle runs up to 40 MHz, pin compatible with standard 80c52, instruction-set compatible with MCS-51 core, four 8-bit I/O ports, three 16-bit timers, 12 interrupt sources with two levels of priority, on-chip oscillator and clock circuitry, two enhanced full duplex serial ports, 32 KB flash EPROM, 256 bytes scratch-pad RAM, 1KB on-chip SRAM for MOVX instruction, programmable watchdog timer, dual 16-bit data pointers, software programmable access cycle to external RAM/peripherals.