# 8051 Core Specification

*Authors: Jaka Simsic*
*Simon Teran*
*jakas@opencores.org*
*simont@opencores.org*

**Rev. 0.1**
**August 14, 2001**

**First Draft**

# Revision History

| Rev. | Date | Authors | Description |
|---|---|---|---|
| 0.1 | 14/08/01 | Jaka Simsic, Simon Teran | First Draft |
| | | | |
| | | | |

# Table Of Contents


**INTRODUCTION ........ 7**
FEATURES ........ 7

**ARCHITECTURE ........ 9**
MEMORY ORGANIZATION ........ 9
Logical separation of program and data memory ........ 9
Program Memory ........ 9
Data Memory ........ 9
Using both external memories together ........ 10
CPU CLOCK ........ 10
INTERRUPT STRUCTURE ........ 10
PORT STRUCTURES ........ 11
Writing to a Port ........ 11
Read-Modify-Write Feature ........ 12
TIMER/COUNTERS ........ 12
RESET ........ 12

**OPERATION ........ 13**
INSTRUCTION SET ........ 13
Addressing Modes ........ 13
Instructions ........ 14
TIMER/COUNTERS ........ 18
Timer/Counter modes ........ 18
INTERRUPTS ........ 19
External interrupts ........ 20
Timer 0 and Timer 1 interrupts ........ 20
Handling of Interrupts ........ 20
RESET ........ 21

**REGISTERS ........ 23**
LIST OF REGISTERS ........ 23
PSW: Program Status Word ........ 24
INTERRUPTS HANDLING REGISTERS ........ 24
IE: Interrupt Enable Register ........ 24
IP: Interupt Priority Register ........ 25
TIMER/COUNTERS CONTROL REGISTERS ........ 25
TCON: Timer/Counter Control Register ........ 25

TMOD: Timer/Counter Mode Control Register……………………………………26

# Table Of Figures

Figure 1 Features of 8051 core ........ 8
Figure 2 Internal Data Memory ........ 10
Figure 3 Interrupt locations in 8051 Program Memory ........ 21

# Table Of Tables

Table 1 Alternate Functions of Port 3 Pins ........ 11
Table 2 Read-modify-write Instructions ........ 12
Table 3 List of Arithmetic Instructions ........ 15
Table 4 List of Logical Instructions ........ 15
Table 5 List of Data Transfer Instructions that Access Internal Data Memory........ 16
Table 6 List of DataTransfer Instructions that Access External Data Memory Space ... 16
Table 7 List of Lookup Table Read Instructions........ 16
Table 8 List of Boolean Instructions ........ 17
Table 9 List of Unconditional Jumps ........ 17
Table 10 List of Conditional Jumps ........ 17
Table 11 Interrupt Priority Within Level........ 20
Table 12 List of Registers ........ 23
Table 13 Register Bank selection........ 24
Table 14 Timer/Counter Operating Mode selection........ 26

# 1

# Introduction

The 8051 microcontroller is member of MCS-51 family, originally designed in the 1980's by Intel. The 8051 has gained great popularity since its introduction and is estimated it is used in a large percentage of all embedded system products.
The basic form of 8051 core includes several on-chip peripherals, like timers and counters, additionally there are 128 bytes of on-chip data memory and up to 4K bytes of on-chip program memory.

## Features

The features of the 8051 core are:

- 8-bit CPU optimized for control applications
- Exstensive Boolean processing (single-bit logic) capabilities
- 64K Program Memory address space
- 64K Data Memory address space
- up to 4K bytes of on-chip Program Memory
- 128 bytes of on-chip Data RAM
- 32 bidirectional and individually adressable I/O lines
- Two 16-bit timer/counters
- 6-source/5-vector interrupt structure with two priority levels

Figure 1 shows the features of 8051 core.

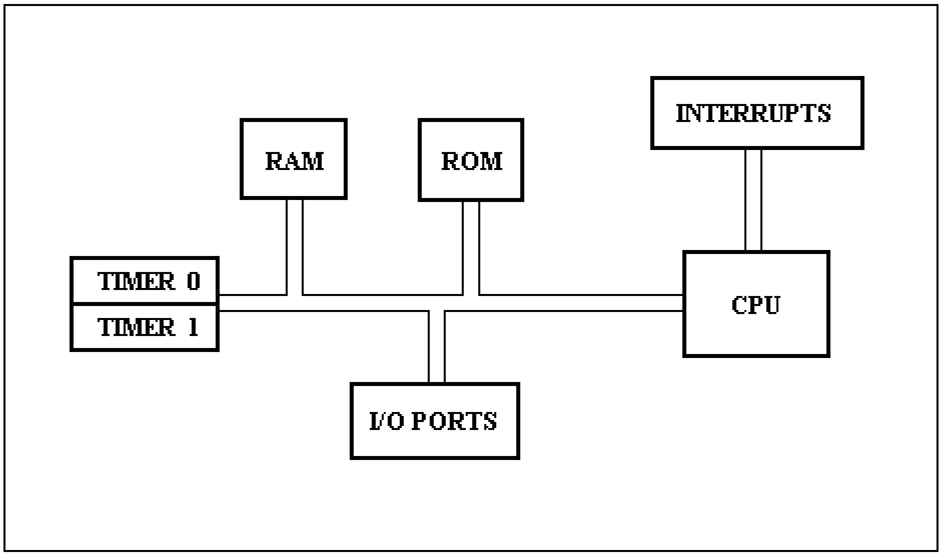


**Figure 1 Features of 8051 core**

# 2

# Architecture

## MEMORY ORGANIZATION

### Logical separation of program and data memory

All 8051 devices have specific memory organization, they have separate address spaces for Program (ROM) and Data (RAM) Memory. This logical separation of Memory is useful because it allows the Data Memory to be accessed by 8-bit addresses, which can obviously be more quickly stored and manipulated by an 8-bit CPU. Of course, the l6-bit Data Memory addresses can still be generated with the DPTR register.

### Program Memory

Program Memory can only be read, not written to. The address space for 8051 core is 16-bit, so there is maximum of 64K bytes of Program Memory. Up to 4 Kbytes of Program Memory can be on chip, internal Program Memory of the 8051 core. For access to external Program Memory is used signal PSEN (Program Store Enable).

### Data Memory

Data Memory is on a separate address space than Program Memory. For external Data Memory accesses the CPU generates read and write signals RD and WR, as needed.
The memory architecture of 8051 core includes 128 bytes of on-chip Data Memory which are more easily accessible directly by its instructions and there is also a number of Special Function Registers (SFRs). Internal Data Memory contains four banks of eight registers and a special 32-byte long segment which is bit addressable by a special subset of the 8051 instructions, the bit-instructions. External memory of maximum 64K bytes is accessable by "movx" instructions.
Figure 2 shows 128 bytes of on-chip Data Memory, with its 4 banks of registers and 32-byte bit addressable segment.

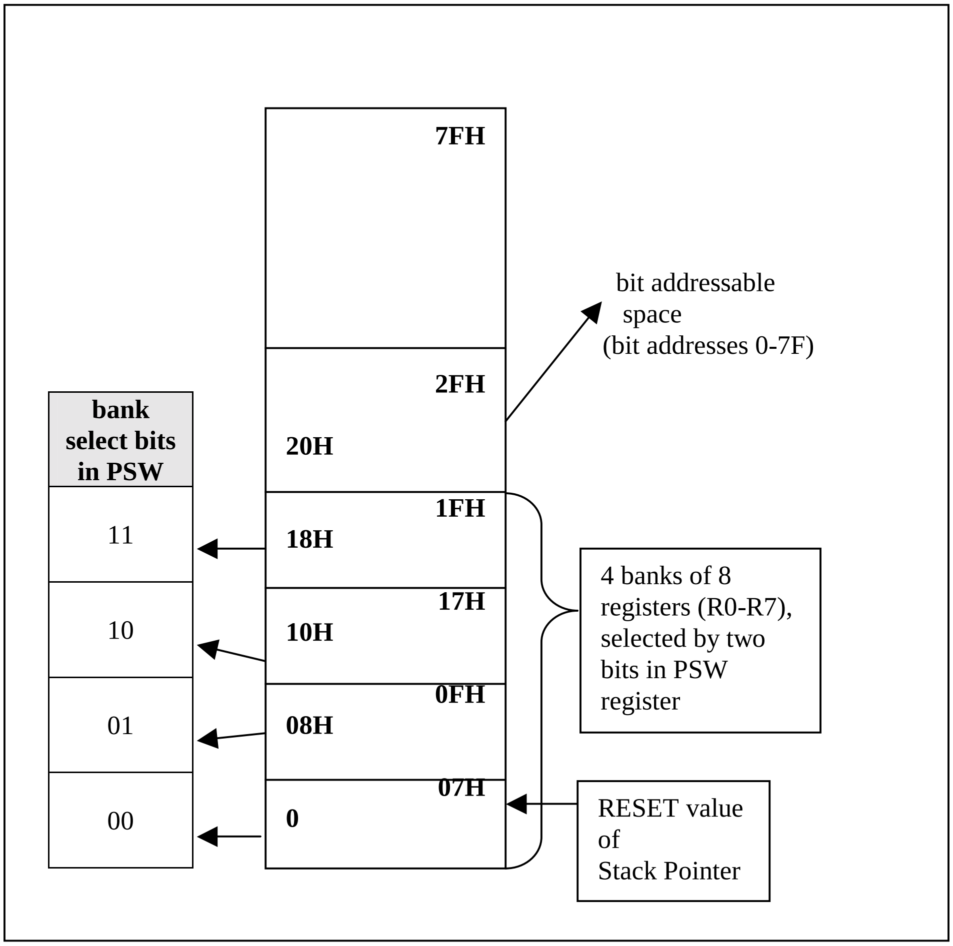


**Figure 2 Internal Data Memory, 128 bytes**

## Using both external memories together

Both memories (external Program Memory and external Data Memory) can be combined if desired.The read strobe to the external Program/Data memory is composed of the RD and PSEN signals which must be applied to the inputs of an AND gate and then the output of the gate can be used as the read strobe.

## CPU CLOCK

8051 microcontroller has a clock input pin.

## INTERRUPT STRUCTURE

The 8051 core provides 4 interrupt sources: 2 external interrupts, 2 timer interrupts.

Additional description of interrupt structure follows in Operations chapter.

# PORT STRUCTURES

The 8051 core contains four I/O ports. All four ports in the 8051 core are bidirectional. Each port has SFR (Special Function Registers P0 through P3) which works like a latch, an output driver and an input buffer.

Both the output driver and the input buffer of Port 0, and the output driver of Ports 2 are used for accessing the external memory. It works like this: Port 0 outputs the low byte of the external memory address (which is time-multiplexed with the byte being written or read) and Port 2 outputs the highbyte of the external memory address (this is only needed when the address is 16 bits wide). If the address in question is 8 bits wide the Port 2 pins are not needed in this application.

The Port 3 pins are multifunctional. Their alternate functions are listed in Table 1.

| Port Pin | Alternate Function |
|---|---|
| P3.2 | ~INT0 (external interrupt) |
| P3.3 | ~INT1 (external interrupt) |
| P3.4 | T0 (Timer/Counter 0 external input) |
| P3.5 | T1 (Timer/Counter 1 external input) |
| P3.6 | ~WR (external Data Memory writestrobe) |
| P3.7 | ~RD (external DataMemory readstrobe) |

**Table 1 Alternate Functions of Port 3 Pins**

This alternate functions are activated with the 1 written in the corresponding bit latch in the port SFR.

## Writing to a Port

The new value arrives at the latch during the last phase (Phase 2), of the final cycle of the instruction that changes the value in a port latch. Because the port latches are sampled by their output buffers only during Phase 1 of any clock period (during Phase 2 the output buffer holds the value it saw during the previous Phase 1), the new value in the port latch won’t actually appear at the output pin until the next Phase 1, which will be at the beginning of the following machine cycle.

# Read-Modify-Write Feature

When reading a port some instructions read the latch and others read the pin. The instructions that read the latch rather than the pin are the ones that read a value (possibly change it), and then rewrite it to the latch. Instructions with these attributes are called “read-modify-write” instructions. The instructions listed in a Table 2 below are read-modify-write instructions.

| | |
|---|---|
| ANL | (logical AND, e.g., ANL PI, A) |
| ORL | (logical OR, e.g., ORL P2, A) |
| XRL | (logical EX-OR, e.g., XRL P3, A) |
| JBC | (jump if bit = 1 and clear bit, e.g., JBC P1.1, LABEL) |
| CPL | (complement bit, e.g., CPL P3.0) |
| INC | (increment, e.g., INC P2) |
| DEC | (decrement, e.g., DEC P2) |
| DJNZ | (decrernent and jump if not zero, e.g., DJNZ P3, LABEL) |
| MOV,PX.Y, C | (move carry bit to bit Y of Port X) |
| CLR PX.Y | (clear bit Y of Port X) |
| SETB PX.Y | (set bit Y of Port X) |

**Table 2 Read-modify-write Instructions**

# TIMER/COUNTERS

The 8051 has two 16-bit Timer/Counter registers: Timer 0 and Timer 1. Both can work either as timers or event counters.
Both have four different operating modes from which to select. All modes are described in Operations chapter.

# RESET

The reset input is the RST pin.

# 3

# Operation

## INSTRUCTION SET

The instruction set of 8051 core is like already said optimized for 8-bit control applications. This optimization shows in a variety of fast addressing modes for accessing the internal RAM in order to facilitate byte operations on small data structures. The instruction set is also good for systems that require a lot of Boolean processing because it has an extensive support for one-bit variables as a separate data type (that makes direct bit manipulation a lot easier).

## Addressing Modes

In the next section follows a brief description of the 8051 instruction set addressing modes.

### DIRECT ADDRESSING

In direct addressing the operand is specified by an 8-bit address field in the instruction. This address mode is possible only for addressing internal Data RAM and SFRs.

### INDIRECT ADDRESSING

In indirect addressing the instruction specifies a register which contains the address of the operand. The address register for 8-bit addresses can be R0 or R1 of the selected register bank, or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit "data pointer" register, DPTR.
Both internal and external RAM can be indirectly addressed.

### REGISTER INSTRUCTIONS

Special instructions are used for accessing four register banks (containing registers R0 to R7). This instructions have a 3-bit register specification within the opcode of the instruction.
This way of accessing the registers is much more efficient, since there is no need for the address byte. When such instruction is executed one of registers in selected register bank is accessed.
Register bank is selected by two bank select bits in PSW.

### REGISTER-SPECIFIC INSTRUCTIONS

These are instructions which are specific to a certain register and they don't need an address byte (they always operate with the same register).

### IMMEDIATE CONSTANTS

In this address mode the value of a constant follows the opcode.
For example:

MOV A, #10

loads the Accumulator with the decimal number 10.

### INDEXED ADDRESSING

In this mode only Program Memory can be accessed and it can only be a read.
This addressing mode is used for reading look-up tables in Program Memory and another type of indexed addressing is used in the “case jump” instruction.

## Instructions

In Tables 3 to 10 in following section are described the instructions of 8051 core. Each Table contains a categorie of instructions, like Arithmetic instructions, Logical instructions, etc.

Values in column Execution time in all Tables are meant to be in μs but this depends from the speed of processor.

The abreviations in column Addressing modes stand for:
Dir……..Direct Addressing

Ind……. Indirect Addressing
Reg…….Register Instructions
Imm……Immediate Constants
All Addressing modes are described above.
If the instruction is register-specific the column Addressing modes notes which register is used.

## ARITHMETIC INSTRUCTIONS

| Mnemonic | Operation | Addressing modes | Execution time |
|---|---|---|---|
| ADD A, <byte> | A = A + <byte> | Dir, Ind, Reg, Imm | 1 |
| ADDC A, <byte> | A = A + <byte> + C | Dir, Ind, Reg, Imm | 1 |
| SUBB A, <byte> | A = A - <byte> - C | Dir, Ind, Reg, Imm | 1 |
| INC A | A = A + 1 | Accumulator only | 1 |
| INC <byte> | <byte> = <byte> + l | DPTR only | 1 |
| INC DPTR | DPTR = DPTR + 1 | Dir, Ind, Reg | 2 |
| DEC A | A = A – 1 | Accumulator only | 1 |
| DEC <byte> | <byte> = <byte> - l | Dir, Ind, Reg | 1 |
| MUL AB | B:A=B x A | ACC and B only | 4 |
| DIV AB | A = Int [A/B]<br>B = Mod [A/Bl | ACC and B only | 4 |
| DA A | Decimal Adjust | Accumulator only | 1 |

**Table 3 List of Arithmetic Instructions**

## LOGICAL INSTRUCTIONS

| Mnemonic | Operation | Addressing modes | Execution time |
|---|---|---|---|
| ANL A, < byte> | A = A .AND. <byte> | Dir, Ind, Reg, Imm | 1 |
| ANL A, < byte> | <byte> = <byte> .AND. A | Dir | 1 |
| ANL <byte>, #data | <byte> = <byte> .AND. #data | Dir | 2 |
| ORL A, < byte> | A = A .OR. <byte> | Dir, Ind, Reg, Imm | 1 |
| ORL A, < byte> | <byte> = <byte> .OR. A | Dir | 1 |
| ORL <byte>, #data | <byte> = <byte> .OR. #data | Dir | 2 |
| XRL A, < byte> | A = A .XOR. <byte> | Dir, Ind, Reg, Imm | 1 |
| XRL A, < byte> | <byte> = <byte> .XOR. A | Dir | 1 |
| XRL <byte>, #data | <byte> = <byte> .XOR. #data | Dir | 2 |
| CRL A | A = 00H | Accumulator only | 1 |
| CPL A | A = .NOT. A | Accumulator only | 1 |
| RL A | Rotate ACC Left 1 bit | Accumulator only | 1 |
| RLC A | Rotate Left through Carry | Accumulator only | 1 |
| RR A | Rotate ACC Right 1 bit | Accumulator only | 1 |
| RRC A | Rotate Right through Carry | Accumulator only | 1 |
| SWAP A | Swap Nibbles in A | Accumulator only | 1 |

**Table 4 List of Logical Instructions**

## DATA TRANSFERS

| Mnemonic | Operation | Addressing modes | Execution time |
|---|---|---|---|
| MOV A, <src> | A = <src> | Dir, Ind, Reg, Imm | 1 |
| MOV <dest>, A | <dest> = A | Dir, Ind, Reg | 1 |
| MOV <dest>, <src> | <dest> = <src> | Dir, Ind, Reg, Imm | 2 |
| MOV DPTR, #data 16 | DPTR = 16-bit immediate constant | Imm | 2 |
| PUSH <src> | INCSP: MOV “@’SP’, <src> | Dir | 2 |
| POP <dest> | MOV <dest>, “@SP”: DECSP | Dir | 2 |
| XCH A, <byte> | ACC and <byte> exchange data | Dir, Ind, Reg | 1 |
| XCHD A, @Ri | ACC and @Ri exchange low nibbles | Ind | 1 |

**Table 5 List of Data Transfer Instructions that Access Internal Data Memory**

| Address width | Mnemonic | Operation | Execution time |
|---|---|---|---|
| 8 bits | MOVX A, @Ri | Read external RAM @Ri | 2 |
| 8 bits | MOVX @Ri, A | Write external RAM @Ri | 2 |
| 16 bits | MOVX A, @DPTR | Read external RAM @DPTR | 2 |
| 16 bits | MOVX @DPTR, A | Write external RAM @DPTR | 2 |

**Table 6 List of DataTransfer Instructions that Access External Data Memory Space**

## LOOKUP TABLES

| Mnemonic | Operation | Execution time |
|---|---|---|
| MOVC A, @A+DPTR | Read Program Memory at (A + DPTR) | 2 |
| MOVC A, @A+PC | Read Program Memory at (A + PC) | 2 |

**Table 7 List of Lookup Table Read Instructions**

## BOOLEAN INSTRUCTIONS

| Mnemonic | Operation | Execution time |
|---|---|---|
| ANL C, bit | C = C .AND. bit | 2 |
| ANL C, /bit | C = C .AND. .NOT. bit | 2 |
| ORL C, bit | C = C .OR. bit | 2 |
| ORL C, /bit | C = C .OR. .NOT. bit | 2 |
| MOV C, bit | C = bit | 1 |
| MOV bit, C | bit = C | 2 |
| CRL C | C = 1 | 1 |
| CRL bit | bit = 0 | 1 |
| SETB C | C = 1 | 1 |
| SETB bit | bit = 1 | 1 |
| CPL C | C = .NOT. C | 1 |
| CPL bit | bit = .NOT. bit | 1 |
| JC rel | Jump if C = 1 | 2 |
| JNC rel | Jump if C = 0 | 2 |
| JB bit, rel | Jump if bit = 1 | 2 |
| JNB bit, rel | Jump if bit = 0 | 2 |
| JBC bit, rel | Jump if bit = 1; CLR bit | 2 |

**Table 8 List of Boolean Instructions**

## JUMP INSTRUCTIONS

| Mnemonic | Operation | Execution Time |
|---|---|---|
| JMP addr | Jump to addr | 2 |
| JMP @A + DPTR | Jump to A + DPTR | 2 |
| CALL addr | Call subroutine at addr | 2 |
| RET | Return from subroutine | 2 |
| RETI | Return from interrupt | 2 |
| NOP | No operation | 1 |

**Table 9 List of Unconditional Jumps**

| Mnemonic | Operation | Addressing Modes | Execution Time |
|---|---|---|---|
| JZ rel | Jump if A = 0 | Accumulator only | 2 |
| JNZ rel | Jumpif A ≠ 0 | Accumulator only | 2 |
| DJNZ <byte>, rel | Decrement and jump if not zero | Dir, Reg | 2 |
| CJNE A, <byte>, rel | Jump if A ≠ <byte> | Dir, Imm | 2 |
| CJNE <byte>, #data, rel | Jump if <byte> ≠ #data | Ind, Reg | 2 |

**Table 10 List of Conditional Jumps**

# TIMER/COUNTERS

Like already said the 8051 core has two 16-bit Timer/Counter registers. These are Timer/Counter 0 and Timer/Counter 1. These registers can be used as timers or as event counters. When a register is in the “Timer” state, it is incremented every machine cycle, in the “Counter” function, the register is incremented when there is a l-to-0 transition at its external input pin, pin T0 for Timer/Counter 0 or pin T1 for Timer/Counter 1.
Both registers have additional operating modes. These four operating modes are all described in next section.

## Timer/Counter modes

The selection for "Timer" or "Counter" function is done by control bits C/T in the Special Function Register TMOD (see Registers chapter). Both Timer/Counters have four operating modes which, Modes 0, 1, and 2 are the same for both Timer/Counters, Mode 3 is different. Modes are selected by bit pairs (M1, M0) in TMOD SFR. Another SFR, which is used for work with the Timer/Counters is TCON, which, among other, contains flag (TFx) and control (TRx) bits.

### MODE 0

Both Timer 1 and Timer 0 in Mode 0 operate as an 8-bit Counters (with a divide-by-32 prescaler).
In this mode, the Timer register is configured as a 13-Bit register. The 13-Bit register consists of all 8 bits of THl and the lower 5 bits of TL1. The upper 3 bits of TLl are indeterminate and should be ignored. Setting the run flag (TR1) does not clear the registers.
The Timer interrupt flag TF1 is set when the count rolls over from all 1s to all 0s.

Mode 0 operation is the same for Timer 0 as for Timer 1. Just substitute Timer 0 for the corresponding Timer 1 signals.

### MODE 1

Mode 1 is the same as Mode 0 (for both Timers), except that the Timer register is configured as 16-bit register.

**MODE 2**

In Mode 2 both Timer registers are configured as an 8-bit Counters (TLl and TL0) with automatic reload. Overflow from TL1 (TL0) sets TFl (TF0), and also reloads TL1 (TL0) with the contents of THl (TH0), which is preset by software. The reload leaves TH1 (TH0) unchanged.

**MODE 3**

Mode 3 is different for Timer 1 and Timer0.
In Mode 3 Timer 1 just holds its count. It operates the same as when the TRl is set to 0.

For Timer 0 is different, in Mode 3 TL0 and TH0 of Timer 0 are established as two separate counters. TL0 uses the Timer 0 control bits for its work: C/T, GATE, TR0, ~INT0, and TF0. TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TFl from Timer 1. TH0 is now actually in control of the “Timer 1“ interrupt.

Mode 3 is provided for applications that require an extra 8-bit timer or counter. With Timer 0 in Mode 3, 8051 core looks like it has three Timer/Countes. When Timer 0 is in Mode 3, Timer 1 can be turned on and off by switching it out of and into its own Mode 3, or can still be used in any application not requiring an interrupt.

# INTERRUPTS

As previously noted, the 8051 core provides 4 interrupt sources (2 external interrupts, 2 timer interrupts).

They are all controlled via two SFRs, IE and IP (both registers are described in Registers chapter). Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the SFR named IE (Interrupt Enable). In the register also exists a global disable bit, which can be cleared to disable all interrupts at once.
Likewise each interrupt source can also be individually set to one of two priority levels by setting or clearing a bit in the SFR named IP (Interrupt Priority).
A low-priority interrupt can be interrupted by high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can’t be interrupted by any other interrupt source.
If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced, so within each priority level there is a second priority structure. This internal priority structure is determined by the polling sequence, the priority levels are shown in Table 11.

| Source | Priority Within Level |
|---|---|
| IE0 | highest |
| TF0 | |
| IE1 | |
| TF1 | lowest |

**Table 11 Interrupt Priority Within Level**

# External interrupts

The external interrupts ~INT0 and ~INT1 have two ways of activation, they can be level-activated or transition-activated. This depends on bits IT0 and ITl in Special Function Register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in SFR TCON. On-chip hardware cleares the flag that generated an external interrupt when the service routine is vectored to, but only if the interrupt was transition-activated. When the interrupt is level-activated, then the external requesting source is controling the request flag, not the on-chip hardware.

# Timer 0 and Timer 1 interrupts

Beside the external interrupt there are also the Timer 0 and Timer 1 interrupts. The Timers interrupts are generated by TF0 and TFl flags in their respective Timer/Counter registers. Similary like in the case of transition-activated external interrupts , the flag that generated an interrupt is cleared by the on-chip hardware when the service routine is vectored to.

# Handling of Interrupts

When interrupt occurs (or correctly, when the flag for an enabled interrupt is found to be set (l)), the interrupt system generates an LCALL to the appropriate location in Program Memory, unless some other condition blocks the interrupt.
Several conditions can block an interrupt:
- an interrupt of equal or higher priority level is already in progress
- the current (polling) cycle is not the final cycle in the execution of the instruction in progress
- the instruction in progress is RETI or any write to the IE or IP registers.

If an interrupt flag is active but not being responded to for one of the above conditions, must be still active when the blocking condition is removed, or the denied interrupt will not be serviced.

Next step is saving the registers on stack. The hardware-generated LCALL causes only the contents of the Program Counter to be pushed onto the stack, and reloads the PC with the beginning address of the service routine. In some cases it also clears the flag that

generated the interrupt, and in other cases it doesn't. It clears an external interrupt flag (IE0 or IEl) only if it was transition-activated.
Like already said only the Program Counter is automatically pushed onto the stack, not the PSW or any other register. Having only the PC be automatically saved gives programmer more freedom to decide how much time to spend saving other registers. Programmer must also be more careful with proper selection, which registers to save.

The service routine for each interrupt begins at a fixed location. The interrupt locations are spaced at 8-byte interval, begining at 0003H for External Interrupt 0, 000BH for Timer 0, 0013H for External Interrupt 1 and 001BH for Timer 1, like shown in Figure 3.

Execution of service routine continues from that location until the end, that is until it encounters RETI instruction. The RETI instruction does two things. It informs the processor that this interrupt routine is finished and secondly, reloads the PC from the two top bytes from the stack.
Similar results colud be accomplished with RET instruction, with the distinction that the interrupt control system would be thinking an interrupt was still in progress.

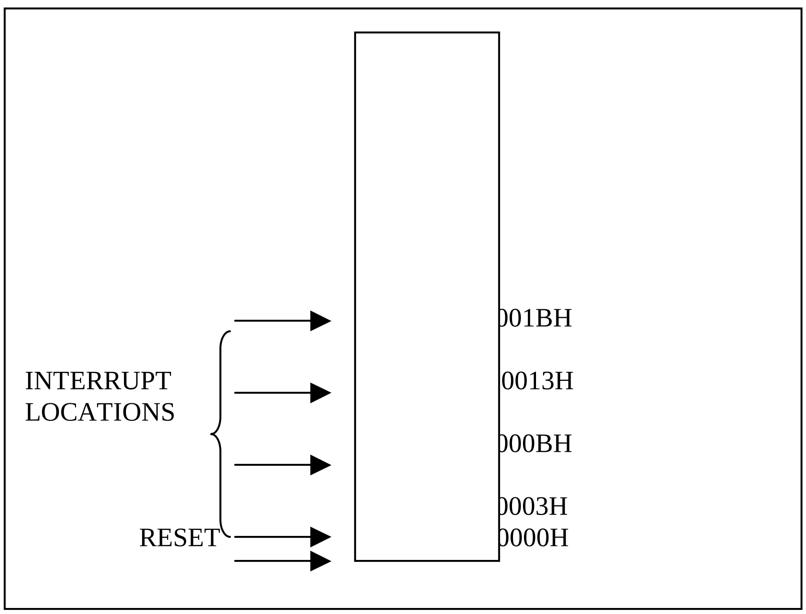


**Figure 3 Interrupt locations in 8051 Program Memory**

# RESET

The reset input is the RST pin. To accomplish a reset the RST pin must be held high for at least two machine cycles.
In the response on the RST signal the CPU generates an internal reset. The external reset signal is asynchronous to the internal clock.

In the internal reset algorithm, 0s are written to all the SFRs except the port latches and the Stack Pointer. The port latches are initialized to FFH and the Stack Pointer to 07H. Important is also to know that driving the ALE and PSEN pins to 0 while reset is active could cause the device to go into an indeterminate state.

The internal RAM is not affected by reset. On power up the RAM content is indeterminate.

# 4

# Registers

This section describes registers inside the 8051 core. The 8051 has a number of Special Function Registers (SFRs). Following Table 12 contains a list of all the SFRs end their addresses:

## LIST OF REGISTERS

| | | |
|---|---|---|
| ACC | Accumulator | 0E0H |
| B | B Register | 0F0H |
| PSW | Program Status Word | 0D0H |
| SP | Stack Pointer | 81H |
| DPTR | Data Pointer (2 Bytes) | |
| DPL | Low Byte | 82H |
| DPH | High Byte | 83H |
| P0 | Port 0 | 80H |
| P1 | Port 1 | 90H |
| P2 | Port 2 | 0A0H |
| P3 | Port 3 | 0B0H |
| IP | Interrupt Priority Control | 0B8H |
| IE | Interrupt Enable Control | 0A8H |
| TMOD | Timer/Counter Mode Control | 89H |
| TCON | Timer/Counter Control | 88H |
| TH0 | Timer/Counter 0 High Byte | 8CH |
| TL0 | Timer/Counter 0 Low Byte | 8AH |
| TH1 | Timer/Counter 1 High Byte | 8DH |
| TL1 | Timer/Counter 1 Low Byte | 8BH |

**Table 12 List of Registers**

Some SFRs can have their bits assigned for various functions. These registers are PSW, PCON, IE, IP, TCON and TMOD. A brief description of this SFRs is provided in next section.

## PSW: Program Status Word (bit addressable)

The PSW register contains several status bits that reflect the current state of the CPU.

| CY | AC | F0 | RS1 | RS0 | OV | — | P |
|---|---|---|---|---|---|---|---|

| | | |
|---|---|---|
| CY | PSW.7 | Carry Flag |
| AC | PSW.6 | Auxiliary Carry Flag |
| FO | PSW.5 | Flag 0 available to the user for general purpose |
| RS1 | PSW.4 | Register Bank selector bit 1 |
| RS0 | PSW.3 | Register Bank selector bit 0 |
| OV | PSW.2 | Overflow Flag |
| — | PSW.1 | User definable flag |
| P | PSW.0 | Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even umber of '1' bits in the accumulator. |

With RS1 and RS0 bits we can select the corresponding register bank.

| RS1 | RS0 | Register Bank | Address |
|---|---|---|---|
| 0 | 0 | 0 | 00H-07H |
| 0 | 1 | 1 | 08H-0FH |
| 1 | 0 | 2 | 10H-17H |
| 1 | 1 | 3 | 18H-1FH |

**Table 13 Register Bank selection**

## INTERRUPTS HANDLING REGISTERS

Both IE and IP registers are used for managing and controlling interrupts. The IE register is used for enableing (or disableing) interrupts and IP is used for setting the priority of interrupts.

## IE: Interrupt Enable Register (bit addressable)

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

| EA | — | — | — | ET1 | EX1 | ET0 | EX0 |
|---|---|---|---|---|---|---|---|

EA IE.7 EA Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA =1, each interrupt source is individually enabled or disabled by setting or

| | | |
|---|---|---|
| | | clearing its enable bit. |
| — | IE.6 | Not implemented |
| — | IE.5 | Not implemented |
| — | IE.4 | Not implemented |
| ET1 | IE.3 | Enable or disable the Timer 1 overflow interrupt |
| EX1 | IE.2 | Enable or disable External Interrupt 1 |
| ET0 | IE.1 | Enable or disable the Timer 0 overflow interrupt |
| EX0 | IE.0 | Enable or disable External Interrupt 0 |

## IP: Interuppt Priority Register (bit addressable)

If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| — | — | — | — | PT1 | PX1 | PT0 | PX0 |

| | | |
|---|---|---|
| — | IP.7 | Not implemented |
| — | IP.6 | Not implemented |
| — | IP.5 | Not implemented |
| — | IP.4 | Not implemented |
| PT1 | IP.3 | Defines the Timer 1 interrupt priority level |
| PXl | IP.2 | Defines External Interrupt 1 priority level |
| PT0 | IP.1 | Defines the Timer 0 interrupt priority level |
| PX0 | IP.0 | Defines the External Interrupt 0 priority level |

# TIMER/COUNTERS CONTROL REGISTERS

Next two registers (TCON and TMOD) are used for work with Timer/Counters. Registers are used for controlling Timer/Counters and for switching between different operating modes.

## TCON: Timer/Counter Control Register (bit addressable)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

| | | |
|---|---|---|
| TFl | TCON.7 | Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine. |
| TR1 | TCON.6 | Timer 1 run control bit. Set/clared by software to turn |

| | | |
|---|---|---|
| | | Timer/Counter 1 ON/OFF. |
| TF0 | TCON.5 | Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine. |
| TR0 | TCON.4 | Timer 0 run control bit. Set/cleared by software to turn Timer/Counter ON/OFF. |
| IE1 | TCON.3 | External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed. |
| IT1 | TCON.2 | Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt. |
| IE0 | TCON.1 | External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed. |
| IT0 | TCON.0 | Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt. |

## TMOD: Timer/Counter Mode Control Register **(not bit addressable)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| GATE | C/T | M1 | M0 | GATE | C/T | M1 | M0 |

First half (from left) of TMOD register is for controling and managing TIMER 1, second half (from left) is for TIMER 0.

| | |
|---|---|
| GATE | When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control). |
| C/T | Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin). |
| Ml | Mode selector bit (see Table 14) |
| M0 | Mode selector bit (see Table 14) |

| M1 | M0 | Operating Mode |
|---|---|---|
| 0 | 0 | 3 13-bit Timer (MCS-48 compatible) |
| 0 | 1 | 3 16-bit Timer/Counter |
| 1 | 0 | 3 8-bit Auto-Reload Timer/Counter |
| 1 | 1 | 3 (Timer 0). TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits, TH0 is an 8-bit Timer and is controlled by Timer 1 control bits |
| 1 | 1 | 3 (Timer 1) Timer/Counter 1 stopped |

**Table 14 Timer/Counter Operating Mode selection**