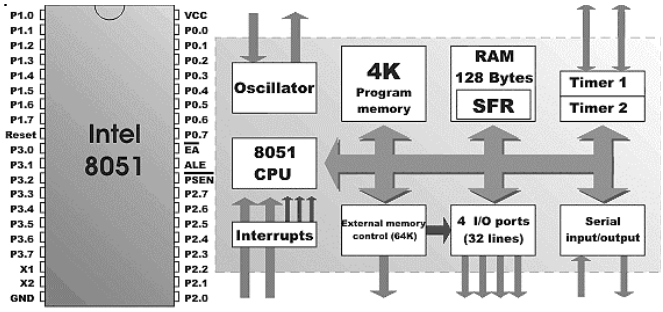
LESSON 12:
INTRODUCTION TO 8051 MICROCONTROLLER INTERNAL ARCHITECTURE

A struggle has been going on between MCU manufacturers for quite a long time, each of them trying to best respond to the ever-increasing demands of the market. Every couple of days there is a brand new chip available, working at higher frequency, with more memory or with better A/D converters. And yet, a closer look to their interior reveals the same or at least very similar structural design referred to as "8051 compatibility". What is it all about? The story began in the 80's when Intel introduced their microcontroller family MCS 8051 to the market. Although this family had quite limited capabilities by today's notions, it quickly captivated the world and became the standard for what is today understood as 'microcontroller'. The most significant cause for such a success can be found in the cleverly chosen configuration which can satisfy a diversity of needs, yet allowing for continuous upgrades (in form of new controllers). In a brief period of time, a decent amount of software has been developed for 8051, making further changes of the hardware core simply uneconomical. Consequently, there is a variety of MCUs available today, basically just the upgraded 8051 models. What exactly makes this microcontroller so special and universal that it is still manufactured by all the major companies, just under a different label?



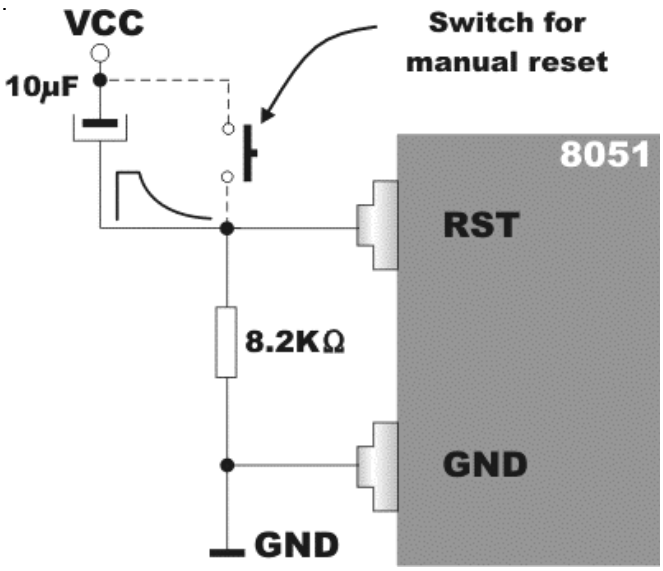
As can be seen on the image above, there is nothing particularly remarkable about MCU 8051:

- 4 kilobytes of ROM is neither too little nor too much.
- 128 bytes of RAM (SFR registers included) can satisfy the basic needs, but is not really astounding.
- 4 ports totaling 32 I/O lines, are usually sufficient for connecting to the environs and are by no means luxury.

Obviously, 8051 configuration is intended to satisfy the needs of programmers developing the controlling devices and instruments. This is one part of its key to success: there is nothing missing, yet there is no lavishness; it is meant for the average user. The other clue can be found in the organization of RAM, Central Processor Unit (CPU), and ports - all of which maximally utilize the available resources and allow further upgrades.

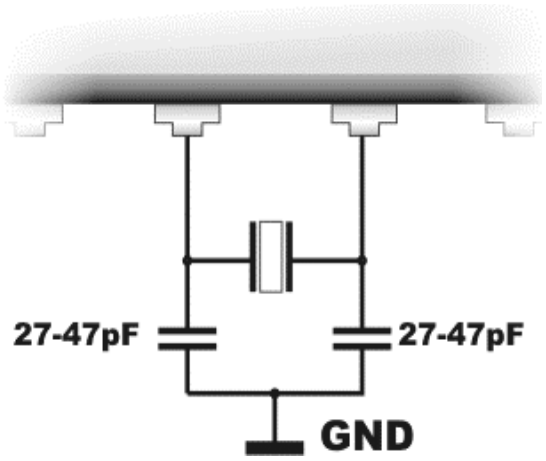
Pins On The Case

- **1–8: Port 1**; Each of these pins can be used as either input or output according to your needs. Also, pins 1 and 2 (P1.0 and P1.1) have special functions associated with Timer 2.
- **9: Reset Signal**; High logical state on this input halts the MCU and clears all the registers. Bringing this pin back to logical state zero starts the program anew as if the power had just been turned on. In another words, positive voltage impulse on this pin resets the MCU. Depending on the device's purpose and environs, this pin is usually connected to the push-button, reset-upon-start circuit or a brown out reset circuit (covered in the previous chapter). The image shows one simple circuit for safe reset upon starting the controller. It is utilized in situations when power fails to reach its optimal voltage.



- **10-17: Port 3**; As with Port 1, each of these pins can be used as universal input or output. However, each pin of Port 3 has an alternative function:
- Pin 10: RXD - serial input for asynchronous communication.
- Pin 11: TXD - serial output for asynchronous communication or clock output for synchronous communication
- Pin 12: INT0 - input for interrupt 0
- Pin 13: INT1 - input for interrupt 1
- Pin 14: T0 - clock input of counter 0
- Pin 15: T1 - clock input of counter 1

- Pin 16: WR - signal for writing to external (add-on) RAM memory synchronous communication.
- Pin 11: TXD - serial output for asynchronous communication or clock output for synchronous communication
- Pin 12: INT0 - input for interrupt 0
- Pin 13: INT1 - input for interrupt 1
- Pin 14: T0 - clock input of counter 0
- Pin 15: T1 - clock input of counter 1
- Pin 16: WR - signal for writing to external (add-on) RAM memory
- Pin 17: RD - signal for reading from external RAM memory
- **18-19: X2 and X1**; Input and output of internal oscillator. Quartz crystal controlling the frequency commonly connects to these pins. Capacitances within the oscillator mechanism (see the image) are not critical and are normally about 30pF. Instead of a quartz crystal, miniature ceramic resonators can be used for dictating the pace. In that case, manufacturers recommend using somewhat higher capacitances (about 47 pF). New MCUs work at frequencies from 0Hz to 50MHz+.



- **20: GND**; Ground
- **21- 28: Port 2**; If external memory is not present, pins of Port 2 act as universal input/output. If external memory is present, this is the location of the higher address byte, i.e. addresses A8 – A15. It is important to note that in cases when not all the 8 bits are used for addressing the memory (i.e. memory is smaller than 64kB), the rest of the unused bits are not available as input/output.
- **29: PSEN**; MCU activates this bit (brings to low state) upon each reading of byte (instruction) from program memory. If external ROM is used for storing the program, PSEN is directly connected to its control pins.
- **30: ALE**; Before each reading of the external memory, MCU sends the lower byte of the address register (addresses A0 – A7) to port P0 and activates the output ALE. External register (74HCT373 or 74HCT375 circuits are common), memorizes the state of port P0 upon receiving a signal from ALE pin, and uses it as part of the address for memory chip. During the second part of the mechanical MCU cycle, signal on ALE is off, and port P0 is used as *Data Bus*. In this way, by adding only one cheap integrated circuit, data from port can be multiplexed and the port simultaneously used for transferring both addresses and data.
- **31: EA**; Bringing this pin to the logical state zero (mass) designates the ports P2 and P3 for transferring addresses regardless of the presence of the internal memory. This means that even if there is a program loaded in the MCU it will not be executed, but the one from the external ROM will be used instead. Conversely, bringing the pin to the high logical state causes the controller to use both memories, first the internal, and then the external (if present).
- **32-39: Port 0**; Similar to Port 2, pins of Port 0 can be used as universal input/output, if external memory is not used. If external memory is used, P0 behaves as address output (A0 – A7) when ALE pin is at high logical level, or as data output (Data Bus) when ALE pin is at low logical level.
- **40: VCC**; Power +5V

Input – Output (I/O) Ports

Every MCU from 8051 family has 4 I/O ports of 8 bits each. This provides the user with 32 I/O lines for connecting MCU to the environs. Unlike the case with other controllers, there is no specific SFR register for designating pins as input or output. Instead, the port itself is in charge: 0=output, 1=input. If particular pin on the case is needed as output, the appropriate bit of I/O port should be cleared. This will generate 0V on the specified controller pin. Similarly, if particular pin on the case is needed as input, the appropriate bit of I/O port should be set. This will designate the pin as input, generating +5V as a side effect (as with every TTL input

Port 0

Port 0 has two fold role: if external memory is used, it contains the lower address byte (addresses A0-A7), otherwise all bits of the port are either input or output. Another feature of this port comes to play when it has been designated as output. Unlike other ports, Port 0 lacks the “pull up” resistor (resistor with +5V on one end). This seemingly insignificant change has the following consequences:

- When designated as input, pin of Port 0 acts as high impedance offering the infinite input resistance with no “inner” voltage.
- When designated as output, pin acts as “open drain”. Clearing a port bit grounds the appropriate pin on the case (0V). Setting a port bit makes the pin act as high impedance.

Therefore, to get positive logic (5V) at output, external “pull up” resistor needs to be added for connecting the pin to the positive pole.

Therefore, to get one (5V) on the output, external "pull up" resistor needs to be added for connecting the pin to the positive pole.

Port 1

This is "true" I/O port, devoid of dual function characteristic for Port 0. Having the "pull up" resistor, Port 1 is fully compatible with TTL circuits

Port 2

When using external memory, this port contains the higher address byte (addresses A8–A15), similar to Port 0. Otherwise, it can be used as universal I/O port

Port 3

Beside its role as universal I/O port, each pin of Port 3 has an alternate function. In order to use one of these functions, the pin in question has to be designated as input, i.e. the appropriate bit of register P3 needs to be set. From a hardware standpoint, Port 3 is similar to Port 0.

Memory

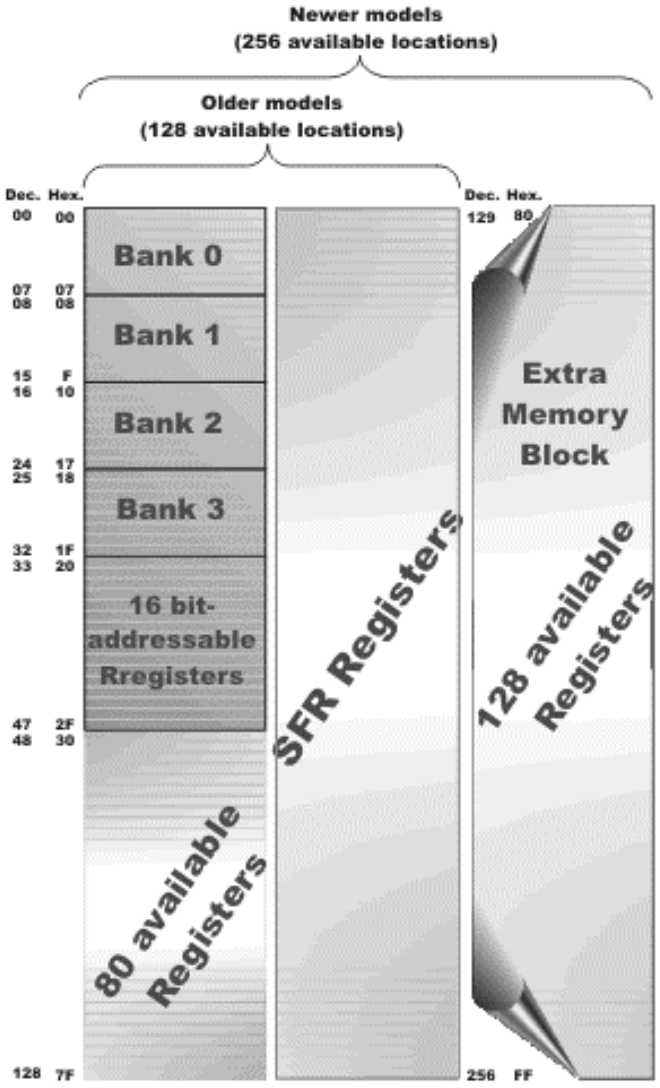
During the runtime, microcontroller uses two different types of memory: one for holding the program being executed (ROM memory), and the other for temporary storage of data and auxiliary variables (RAM memory). Depending on the particular model from 8051 family, this is usually few kilobytes of ROM and 128/256 bytes of RAM. This amount is built-in and is sufficient for common tasks performed "independently" by the MCU. However, 8051 can address up to 64KB of external memory. These can be separate memory blocks, (separate RAM chip and ROM chip) totaling 128KB of memory on MCU which is a real programming goody.

ROM Memory

First models from 8051 family lacked the internal program memory, but it could be added externally in a form of a separate chip. These MCUs can be recognized by their mark which begins with 803 (e.g. 8031 or 8032). New models have built-in ROM, although there are substantial variations. With some models internal memory cannot be programmed directly by the user. Instead, the user needs to proceed the program to the manufacturer, so that the MCU can be programmed (masked) appropriately in the process of fabrication. Obviously, this option is cost-effective only for large series. Fortunately, there are MCU models ideal for experimentation and small specialized series. Many manufacturers deliver controllers that can be programmed directly by the user. These come in a ceramic case with an opening (EPROM version) or in a plastic case without an opening (EEPROM version). This book deals with one of the latter models that can be programmed via simple programmer, even if the chip has already been mounted to the designated device.

RAM Memory

As previously stated, RAM is used for storing temporary data and auxiliary results generated during the runtime. Apart from that, RAM comprises a number of registers: hardware counters and timers, I/O ports, buffer for serial connection, etc. With older versions, RAM spanned 256 locations, while new models feature additional 128 registers. First 256 memory locations form the basis of RAM (addresses 0 – FFh) of every 8051 MCU. Locations that are available to the user span addresses from 0 to 7Fh, i.e. first 128 registers, and this part of RAM is split into several blocks as can be seen in the image below.



- First block comprises 4 "banks" of 8 registers each, marked as R0 - R7. To address these, the parent bank has to be selected.
- Second memory block (range 20h – 2Fh) is bit-addressable, meaning that every belonging bit has its own address (0 to 7Fh). Since the block comprises 16 of these registers, there is a total of 128 addressable bits. (Bit 0 of byte 20h has bit address 0, while bit 7 of byte 2Fh has bit address 7Fh).
- Third is the group of available registers at addresses 2Fh – 7Fh (total of 80 locations) without special features or a preset purpose

Extra Memory Block

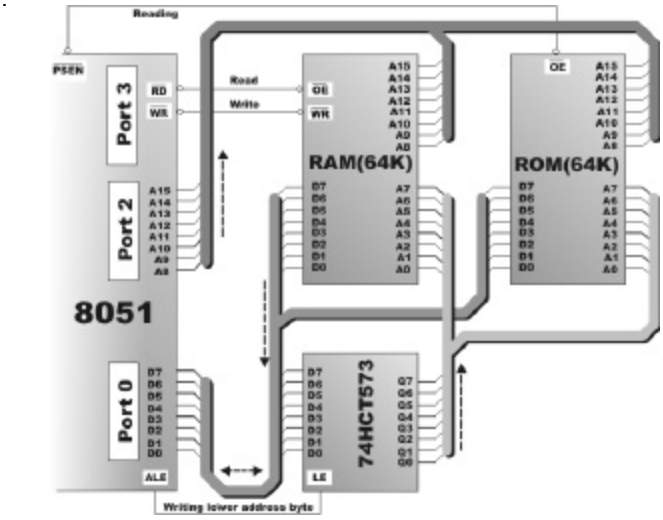
To satisfy the programmers' ever-increasing demands for RAM, latest 8051 models were added an extra memory block of 128 locations. But it is not all that simple... The problem lies in the fact that the electronics which addresses RAM employs 1 byte (8 bits), reaching only the first 256 locations. Therefore, a little trick

had to be applied in order to keep the existing 8-bit architecture for the sake of compatibility with older models. The idea is to make the additional memory block share the addresses with the existent locations intended for SFR registers (80h - FFh). For distinguishing these two physically separate memory areas, different methods of addressing are used: if SFR registers are in question, direct addressing is used; for extra RAM locations, indirect addressing is used.

Memory Expanding

In case the built-in amount of memory (either RAM or ROM) is not sufficient for your needs, there is always an option of adding two external 64KB memory chips. When added, they are addressed and accessed via I/O ports P2 and P3. From user's point of view it's all very simple, because if properly connected most of the job is carried out automatically by MCU.

8051 MCU has two separate read signals, RD# (P3.7) and PSEN#. The first one is active when reading byte from the external data memory (RAM), and the second one is active when reading byte from the external program memory (ROM). Both signals are active on low logical level. The following image shows a typical scheme for such expansion using separate chips for RAM and ROM, known as Harvard architecture



Memory can be also mapped as a single block, functioning as both data memory and program memory simultaneously (only one memory chip is used). This approach is known as Von Neumann architecture. To be able to read the same block using RD# or PSEN#, these two signals were combined via logical AND. In this way, output of AND circuit is low if any of the two inputs is low.

Using the Harvard architecture effectively doubles MCU memory, but that's not the only advantage offered by the method. Keeping the program code separated from the data makes the controller more reliable since there is no writing to the program memory.

SFR Registers (Special Function Registers)

SFR registers can be seen as a sort of control panel for managing and monitoring the microcontroller. Every register and each of the belonging bits has its name, specified address in RAM and strictly defined role (e.g. controlling the timer, interrupt, serial connection, etc). Although there are 128 available memory slots for allocating SFR registers, the basic core shared by 8051 MCUs has but 22 registers. The rest has been left open intentionally to allow future upgrades while retaining the compatibility with earlier models. This fact makes possible to use programs developed for obsolete models long ago.

Addr. (Hex.)	Mark	Full name
80	P0	Port 0
81	SP	Stack Pointer
	DPTR	Data Pointer
82	DPL	Data Low Pointer
83	DPH	Data High Pointer
87	PCON	Power Control
88	TCON	Timer/Counter Control
89	TMOD	Timer/Counter Mode Control
8A	TL0	Timer/Counter0 Low Byte
8B	TL1	Timer/Counter1 Low Byte
8C	TH0	Timer/Counter0 High Byte

Addr. (Hex.)	Mark	Full name
8D	TH1	Timer/Counter1 High Byte
90	P1	Port 1
98	SCON	Serial Port Control
99	SBUF	Serial Data Port
A0	P2	Port 2
A8	IE	Interrupt Enable
B0	P3	Port3
B8	IP	Interrupt Priority Control
D0	PSW	Program Status Word
E0	ACC(A)	Accumulator
F0	B	B Register

Notes