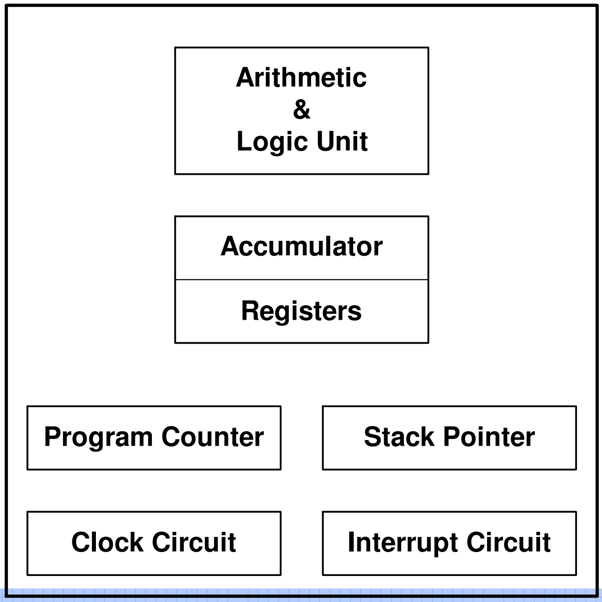
General Purpose CPU

- Have discussed the 4004 was the first developed microprocessor by Intel in 1971.
  - was a 4-bit microprocessor (The data bus was 4-bits wide).
- A microprocessor is a general purpose digital computer central processing unit.
  - But in no sense is the microprocessor a complete digital computer



CPU has its
- ALU
- Program Counter
- Stack Pointer
- Some Registers
- A clock Timing Syste
- Interrupt Circuits

General Purpose CPU

To make this a complete computer one needs to add memory

- usually read-only **program** memory (ROM)
- Random Access **data** memory (RAM)
- Memory decoders
- An Oscillator
- A number of I/O devices   e.g. Serial/Parallel data ports

Additional function features can also be used to free-up CPU time from time-consuming tasks

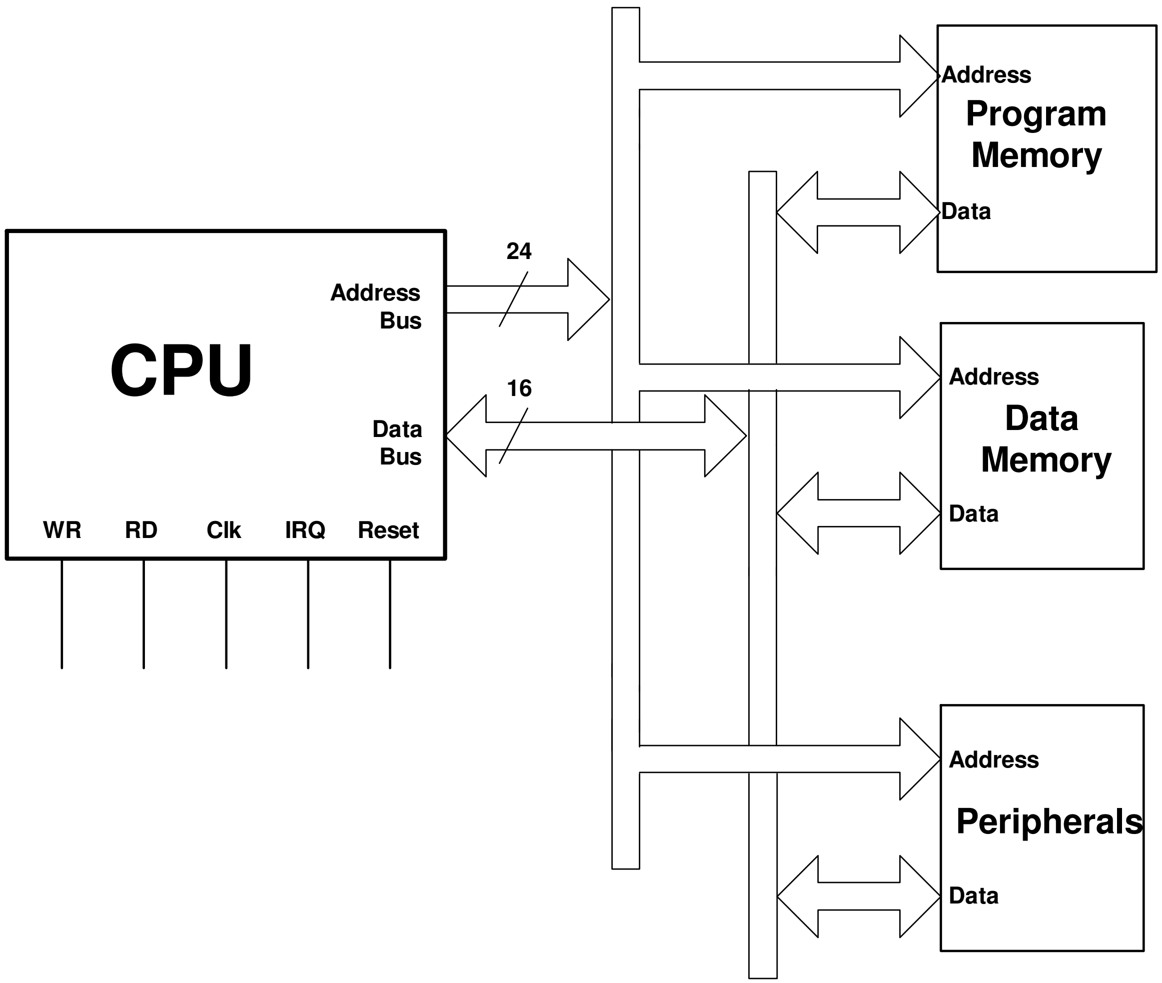
- interrupt handlers
- timers/counters

Equipping the microprocessor with

- mass storage in the form of a harddisk
- I/O peripherals such as a keyboard, CRT display

  ⇒ yields a small computer for general purpose applications.

General Purpose CPU



General Purpose CPU

The key term in the describing this processor is "*general purpose*" .

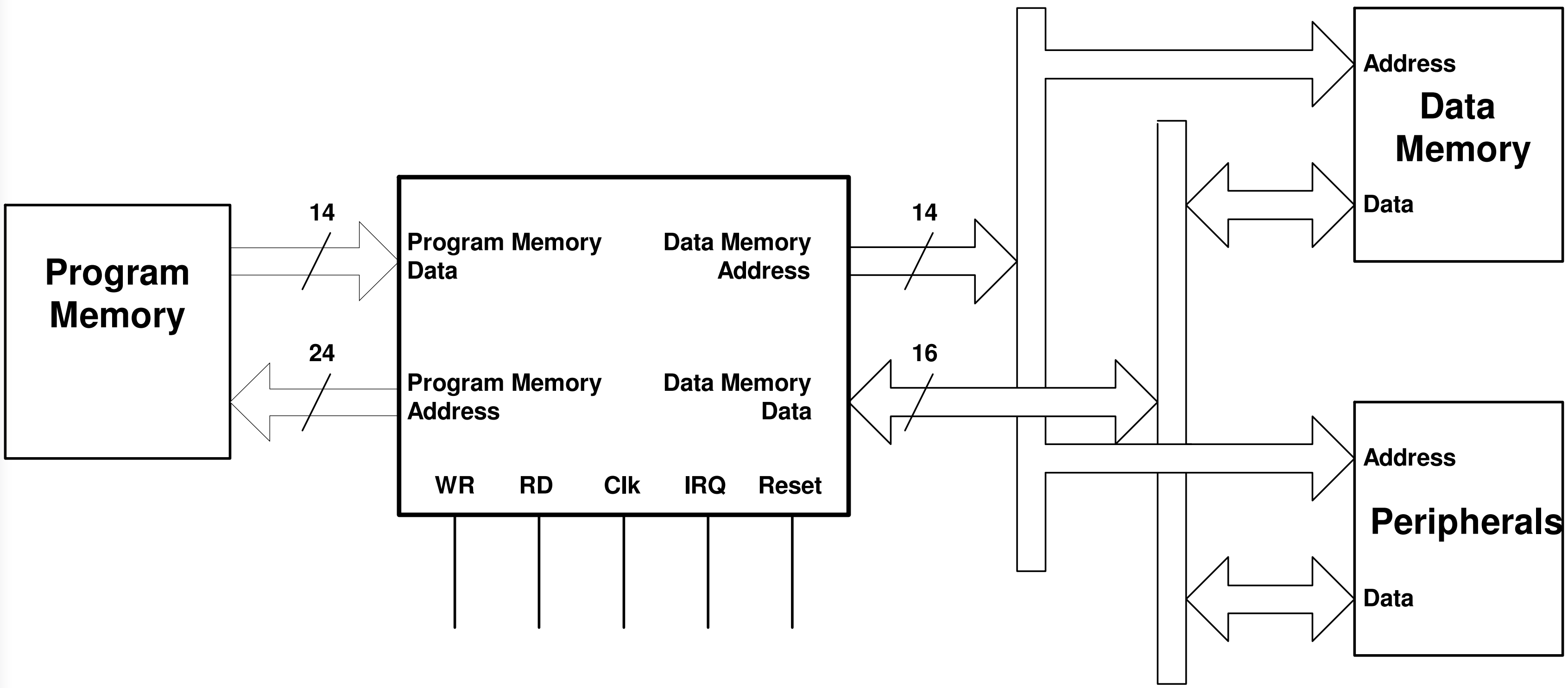
- 16-bit or 32-bit design
- hardware design of a microprocessor CPU is arranged so that a system can be configured around the CPU as the application demands.

The prime use of a microprocessor is to

- fetch instruction
- fetch data,
- execute the instruction,
- display and store the result.

For special applications, need alternative designs

HARVARD ARCHITECTURE



Dedicated Applications

- **Even through the CPU has evolved from a 4-bit in 1972 to 32-bit and 64-bit devices**
  - most engineering applications have not changed
  - these applications continue to be best served by 8-bit CPUs with limited memory size and I/O capability.

- **Cost/unit also continues to dominate processing applications**
  - using an expensive 32-bit processor to perform functions that can be efficiently served by an inexpensive 8-bit processor

- The vast majority of microprocessor production is not for the general purpose computers
  - but dedicated microprocessor applications

Definition of a Single Chip CPU

- New technology in the fabrication of integrated circuits has made it possible to make a better low-bit processor one with not only the CPU but the following incorporated on-chip
  - ROM
  - RAM
  - I/O ports
  - Serial ports
  - Times
  - A/D converters
  - etc.

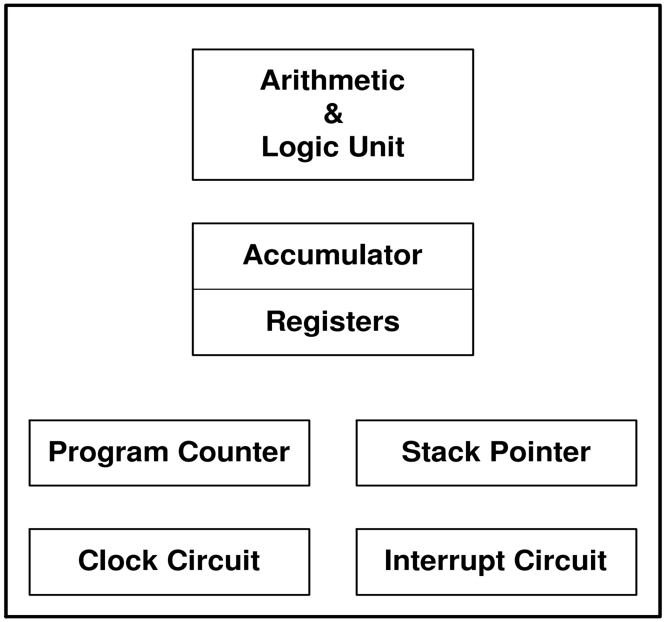
- Such a microprocessor has become known as a ***microcontroller***
  - The term microcontroller is used to describe single chip CPU's aimed at embedded control applications.

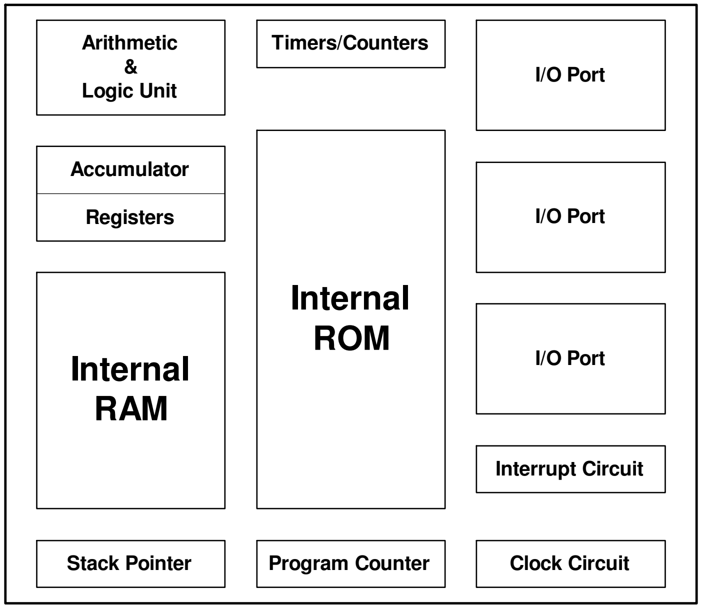
MICROCONTROLLERS

- Embedded control applications are those where the CPU + system is self contained and controls all the inputs and outputs itself

  - **modems, alarm systems**
  - **car engine management system, robotics**
  - **TV remote controls**
  - **answer machines, thermostats**
  - **vending machines.**
  - **etc.**

Microcontrollers





The contrast is best exemplified by the fact that

1. **Most microprocessors have many opcodes for moving data from external memory to the CPU. Microcontrollers will have <u>one</u> to two opcodes for this.**
2. **Microprocessors may have one or two types of bit handling instructions. Microcontrollers will have many.**

Microcontrollers

- Microprocessor is concerned with rapid movement of code and data from external address to the chip.
  - microcontrollers is concerned with movement of bits within the chip.
- These applications are distinct from applications where a PC or host computer control a system,
  - in these cases user input via a keyboard/VDU is possible and off-line storage is available.
- The majority of microcontrollers are derived from earlier CISC CPU’s
  - are consequently CISC
  - However some RISC like microcontrollers are now manufactured.

Microcontrollers

- Key distinguishing feature of a micro-controller based system is very low cost.

  $\Rightarrow$ 4-bit microcontrollers < 50c in volumes of 1000.

- A definition of a microcontroller would be :

  a complete CPU + Peripherals + Memory on a single chip.

- Like a microprocessor, a microcontroller is a general purpose devices but one which is meant to
  - fetch data
  - perform limited calculations on that data
  - control its environment based on these calculations

Microcontrollers

The prime use of a microcontroller :

- **to control the operation of a machine using a fixed program that is stored in ROM and that does not change over the lifetime of the system**

Typical Features of a Microcontrollers

- Single chip CPU
- Internal Data RAM + Registers (128 bytes)
- Internal Program ROM/EPROM/EEPROM (1K - 8K ROM)
- Programmable I/O Ports
- Typical 64K external address/data bus
- On-chip timers
- On-chip Serial Ports (UART)
- On-chip watchdog
- Interrupts

Other Peripherals

- A/D's, Pulse Accumulator, Pulse-Width Modulation

Motorola 68HC11

TIMER SYSTEM

Packed with features but easy to use, the M68HC11 timer architecture is widely recognised as the leader among 8-bit MCUs. Includes Input Capture, Output Compare, Real Time Interrupt, Pulse Accumulator and COP Watchdog. All unused timer pins may be configured as normal digital I/O.

CPU CORE

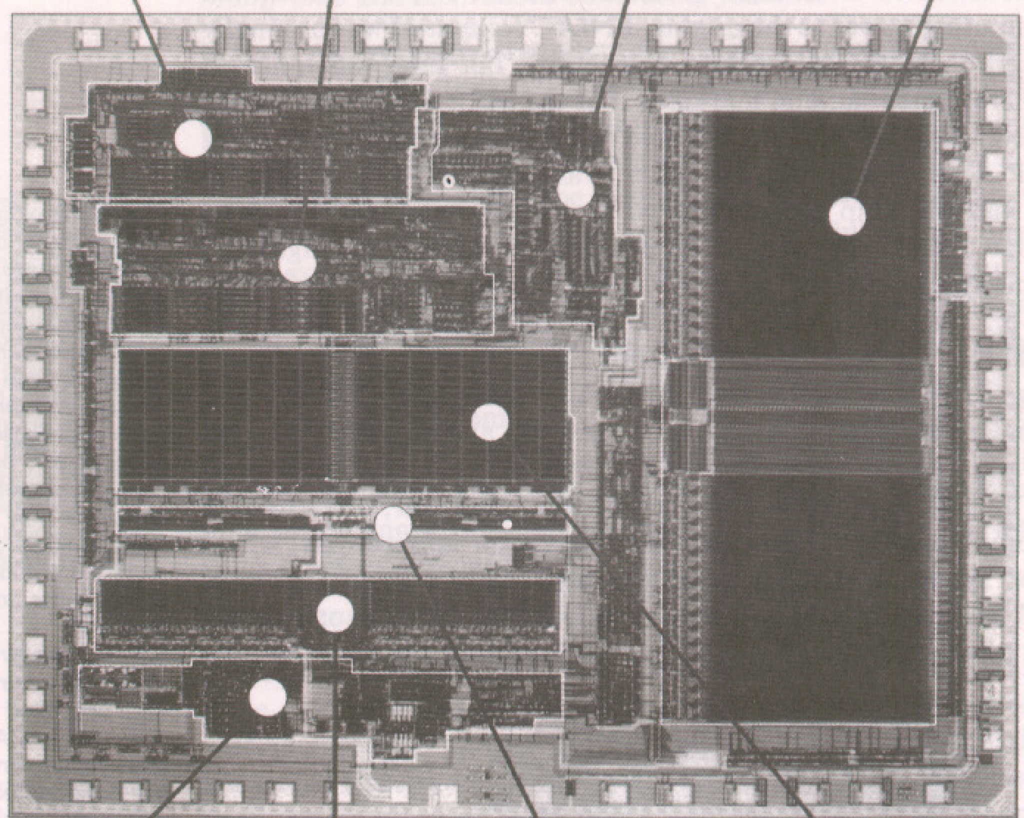
A powerfully enhanced version of the industry standard M6801 core, optimized for low power consumption and providing 91 new opcodes. Operating frequencies up to 4.0MHz for high performance, power-saving STOP and WAIT modes for economy. Six addressing modes, and a fully featured interrupt system.

COMMUNICATIONS

All versions of the M68HC11 include a Serial Peripheral Interface (SPI) and Serial Communications Interface (SCI) as standard. Both have software-selectable baud rates, and are designed to require minimum attention from the CPU to minimise processing overhead. The SPI operates at up to 2Mbits/sec.

EPROM

The latest memory introduction to the M68HC11, the first Motorola MCU family to offer both EPROM and EEPROM. Provides a cost-effective, user-programmable ROM facility for small volume prototype and development runs. Optionally available in windowed packages for UV erasure and reprogramming.



Die photo of the MC68HC711E9, the first MCU with on-chip EPROM and EEPROM.

Typical Microcontrollers

- The most common microcontrollers are 8-bit.
- 4-bit are used in high volume very low cost applications
- 16-bit are used in high-end applications.

Examples:

• Motorola	68HC05, 68HC11
• Intel	80C48 , 80C51
• Zilog	Z80

- Typical clock frequencies are 12 - 16 MHz
- The instruction set typically consists of a large number of single byte instructions but also can allow some multi-byte ones.
- Typical 1 micro-second per instruction eg. ADD, MOV
  - $\Rightarrow$ many clock cycles per instuction
- Microcontrollers are usually a 1-address machine with registers.

Development Systems for Microcontrollers

What is needed to apply a microcontroller to an application ?

1. A development system
2. A device capable of programming EPROM's.
   - many microcontrollers have not only a ROM based version but also a EPROM version and EEPROM version.
   - ⇒ both allow the designer to debug the hardware and software prototype before committing to full-scale production.
3. Software

- The minimum software package consists of a machine assembler. This can be purchased from the microcontroller vendor or independent developer

8051 Microcontroller

- Intel 8051 is a very widely used microcontroller. In fact it includes a whole family of microcontrollers from the 8031 to the 8751.
  - depending on the construction
  - N-Channel MOS ***NMOS*** or Complementary MOS ***CMOS.***
  - variety of packaging types.

- A block diagram of the 40-pin Intel 8051 microcontroller and its functional diagram can be seen below.

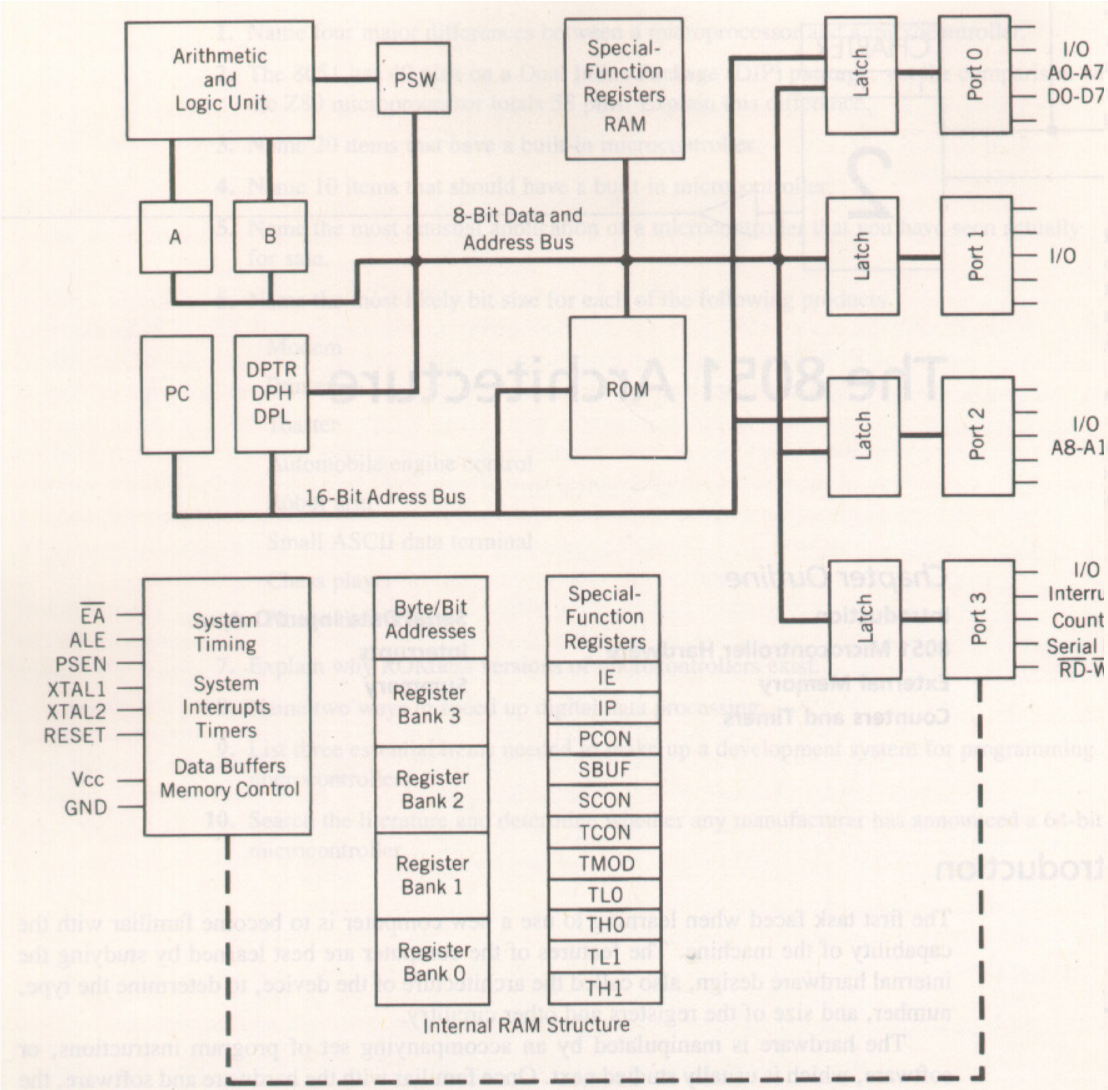
- The operation of the CPU core is much the same as the simple CPU architecture already considered.
- The memory and peripherals are integrated on chip.

8051 Microcontroller

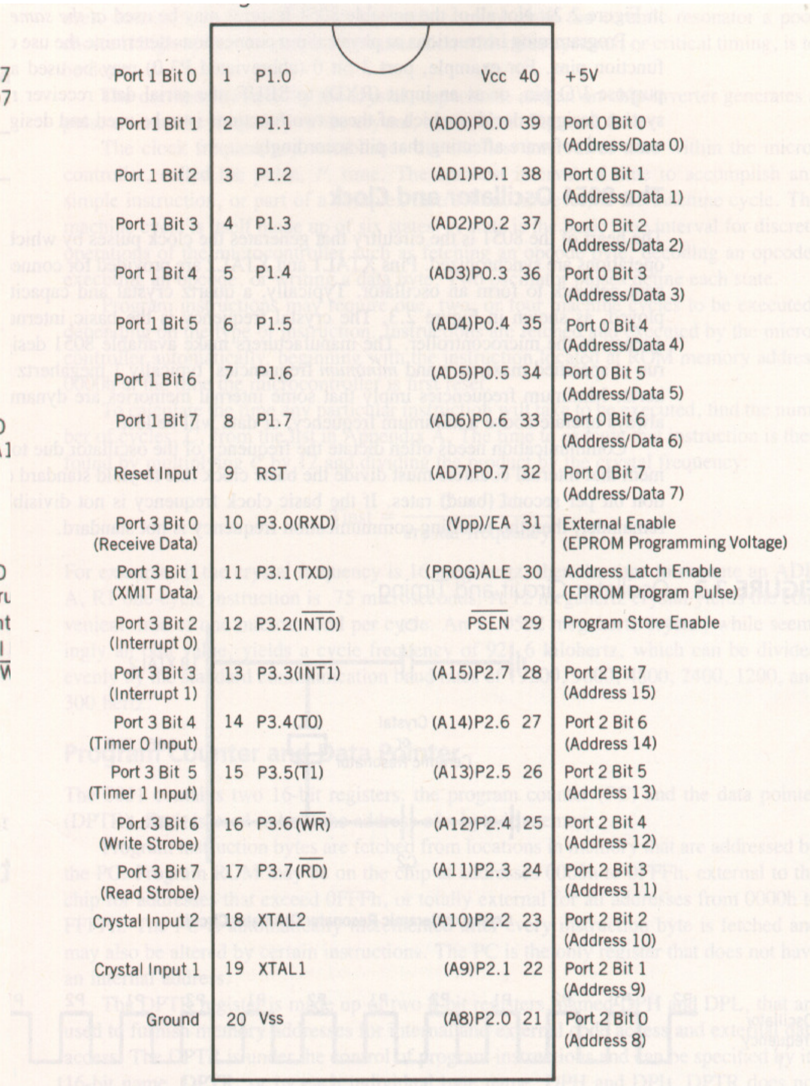
The core of the 8051's 8-bit CPU is made up of

- 8-bit Register A (accumulator) and Register B
- 16-bit Program Counter (PC) and Data Pointer (DPTR)
- Program Address Register
- Program Status Word (PSW)
    - contains the mathematical flags and the register select bits
        - identifies which register is currently being used
- Instruction Decoder
- Timing Control

Intel 8051



Arithmetic and Logic Unit
PSW
Special-Function Registers RAM
Latch
Port 0
I/O
A0-A7
D0-D7
A
B
8-Bit Data and Address Bus
Latch
Port 1
I/O
PC
DPTR
DPH
DPL
ROM
Latch
Port 2
I/O
16-Bit Adress Bus
EA
ALE
PSEN
XTAL1
XTAL2
RESET
Vcc
GND
System Timing
System Interrupts Timers
Data Buffers Memory Control
Byte/Bit Addresses
Register Bank 3
Register Bank 2
Register Bank 1
Register Bank 0
Special-Function Registers
IE
IP
PCON
SBUF
SCON
TCON
TMOD
TL0
TH0
TL1
TH1
Internal RAM Structure
Latch
Port 3
I/O
Serial



Port 1 Bit 0 1 P1.0
Port 1 Bit 1 2 P1.1
Port 1 Bit 2 3 P1.2
Port 1 Bit 3 4 P1.3
Port 1 Bit 4 5 P1.4
Port 1 Bit 5 6 P1.5
Port 1 Bit 6 7 P1.6
Port 1 Bit 7 8 P1.7
Reset Input 9 RST
Port 3 Bit 0 (Receive Data) 10 P3.0(RXD)
Port 3 Bit 1 (XMIT Data) 11 P3.1(TXD)
Port 3 Bit 2 (Interrupt 0) 12 P3.2(INT0)
Port 3 Bit 3 (Interrupt 1) 13 P3.3(INT1)
Port 3 Bit 4 (Timer 0 Input) 14 P3.4(T0)
Port 3 Bit 5 (Timer 1 Input) 15 P3.5(T1)
Port 3 Bit 6 (Write Strobe) 16 P3.6(WR)
Port 3 Bit 7 (Read Strobe) 17 P3.7(RD)
Crystal Input 2 18 XTAL2
Crystal Input 1 19 XTAL1
Ground 20 Vss
Vcc 40 +5V
(AD0)P0.0 39 Port 0 Bit 0 (Address/Data 0)
(AD1)P0.1 38 Port 0 Bit 1 (Address/Data 1)
(AD2)P0.2 37 Port 0 Bit 2 (Address/Data 2)
(AD3)P0.3 36 Port 0 Bit 3 (Address/Data 3)
(AD4)P0.4 35 Port 0 Bit 4 (Address/Data 4)
(AD5)P0.5 34 Port 0 Bit 5 (Address/Data 5)
(AD6)P0.6 33 Port 0 Bit 6 (Address/Data 6)
(AD7)P0.7 32 Port 0 Bit 7 (Address/Data 7)
(Vpp)/EA 31 External Enable (EPROM Programming Voltage)
(PROG)ALE 30 Address Latch Enable (EPROM Program Pulse)
PSEN 29 Program Store Enable
(A15)P2.7 28 Port 2 Bit 7 (Address 15)
(A14)P2.6 27 Port 2 Bit 6 (Address 14)
(A13)P2.5 26 Port 2 Bit 5 (Address 13)
(A12)P2.4 25 Port 2 Bit 4 (Address 12)
(A11)P2.3 24 Port 2 Bit 3 (Address 11)
(A10)P2.2 23 Port 2 Bit 2 (Address 10)
(A9)P2.1 22 Port 2 Bit 1 (Address 9)
(A8)P2.0 21 Port 2 Bit 0 (Address 8)

8051 Microcontroller

Features of Memory

- The 8051 is organised so that program memory and data memory can be in two entirely different physical memory entities, but each has the ***same*** address range

Program Memory is

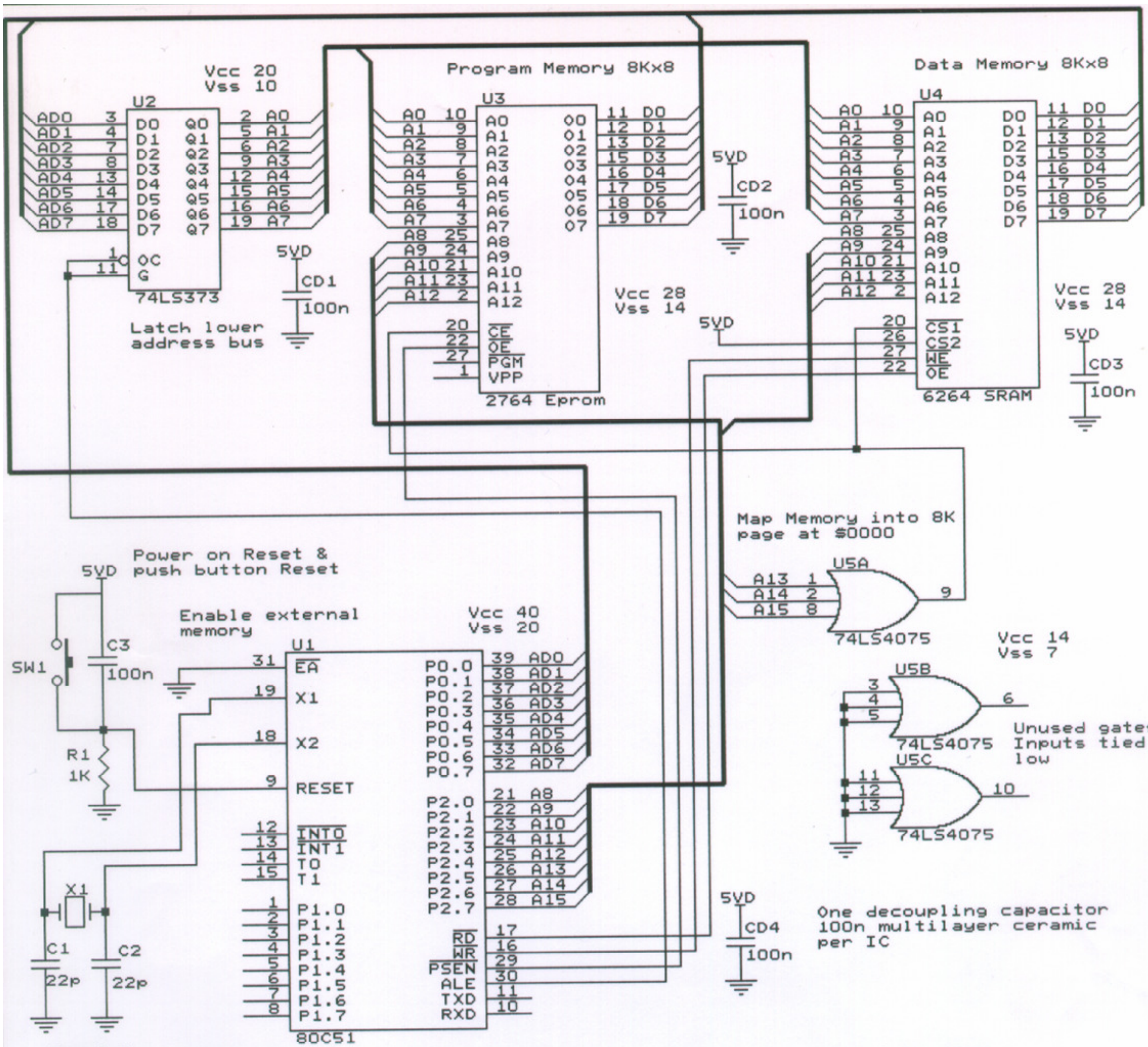
- in the ROM/EPROM block $\Rightarrow$ \$0000 $\rightarrow$ \$0FFF

or can

- be external through Port 0 and Port 2 $\Rightarrow$ \$0000 $\rightarrow$ \$FFFF

Data Memory is

- in the RAM block



80C51 with External Memory
Program Memory 8K x 8 Eprom
Data Memory 8K x 8 SRAM

MICROCONTROLLERS

Registers

The mathematical core of the 8051 is formed by two registers

- Register A is the accumulator
- Register B, used in Multiply and Divide Operations

- Register A is also used in data transfers between the 8051 and external memory.

- The first 32 Bytes if RAM are also mapped as 4 banks of 8 registers (R0 - R7) each.

Hence these memory areas can be accessed as a register or as Data Memory. Likewise other registers :

- ACC,
- B,
- DPTR

can be accessed as RAM or Registers

MICROCONTROLLERS

8051 Instruction Set

Immediate Addressing

The 8051 supports immediate addressing

MOV A, #FF

Register Addressing

Direct register addressing for the 4 banks of Registers and other registers is provided.

MOV A, Rn

MOV Rn, A

MICROCONTROLLERS

Direct Addressing

Direct Addressing is provided for the internal Data Ram and Special Function Registers (for programming the peripherals) using the single byte address assigned to each RAM location.

e.g.	A	$0E0
	B	$0F0
	DPL	$82
	DPH	$83

e.g.	MOV	A, addr
	MOV	addr, A

Indirect Addressing

The indirect addressing mode uses a register to “hold” the actual address

```
MOV     A,@R0
```

MICROCONTROLLERS

OPCODES

The 8051 provides the usual set of arithmetic and logical instructions including :

- ***INC*** and ***DEC*** of a register.
- ***SHIFT*** ACC left or right

It also includes the following BIT operations :

- Set Bit
- Clear Bit
- NOT bit
- AND bit with Carry
- OR bit with Carry
- MOVE carry $\rightarrow$ bit
- MOVE bit $\rightarrow$ carry

MICROCONTROLLERS

Architecture Summary

The 8051 addressing modes and instruction set is fairly typical of 8-bit microcontrollers.

- CISC like CPU, with simple (limited and restrictive) instruction set, limited addressing.

Peripherals

In addition to the basic CPU and internal RAM, microcontrollers have a number of hardware peripherals integrated on-chip.

e.g. The 8051 has

- 4 8-Bit I/O ports, P0 - P3
- 2 16-bit timers
- 1 Serial Port

There are also 2 external hardware interrupts supported

- INT0
- INT1

MICROCONTROLLERS

The on-chip resources of the 80C51 (and all microcontrollers) are multiplexed.

- Therefore if external program memory is used

⇒ PORT 0 and PORT 2 are **NOT** available as I/O ports

PORT 3 shares its pins with :

- The serial port RxD, TxD timer inputs T0 and T1
- The interrupts INT0 and INT1
- The External Data Memory RD and WR lines

⇒ if any of these functions is used

⇒ the corresponding I/O pin of PORT 3 is **NOT** available.

Thus with the 80C51 each of the 24 pins PORT 0, PORT 2 and PORT 3 can be configured for one of 2 possible functions.

The peripherals are programmed and accessed with Special Function Registers (SFR)

MICROCONTROLLERS

I/O Ports

- Each port can be used as a programmable I/O.
- Each port pin has a D-type flip-flop to store the output value of the pin

⇒ Hence each port is made up of 8 D-Flip-Flops which form a register.

In the 80C51 the registers are at the following address :

- PORT 0 $80
- PORT 1 $90
- PORT 2 $A0
- PORT 3 $B0

MICROCONTROLLERS

So if the BYTE $8C is written to the SFR $90
⇒ the I/O pins will have the following value.

• Pin 1	0
• Pin 2	1
• Pin 3	0
• Pin 4	1
• Pin 5	0
• Pin 6	0
• Pin 7	0
• Pin 8	1

Each I/O pin can also be used as an input.
To do this we must write a **1** to the corresponding D-flip-flop.
⇒ when you read the SFR you read the value of the pin *.
⇒ If the pin is not connected (***floating***) it will be read as a **1**.

* read Modify write op-code : read the latch and not the pin.

MICROCONTROLLERS

I/O pins are very useful for the following :

- reading switches
- reading keypads
- writing to LED's
- controlling Motors (PWM) etc.

MICROCONTROLLERS

Timers

Many microcontroller applications require the counting of external events, such as :

- the frequency of a pulse train
- the generation of precise internal time delays between computer actions

To provide a solution to this, the 80C51 has two 16-Bit timers.

- Both are used to count a certain number of internal of external pulses.

The counter is loaded with a number

$\Rightarrow$ when running each high-to-low transition of the pulse increments the counter.

- This continues until it reaches the maximum count ().
- On the next pulse edge it rolls over to zero and sets a timer flag.
- The flag may be read by the program or if the Timer interrupt is enabled

$\Rightarrow$ the flag with generate an interrupt.

MICROCONTROLLERS

To count N pulses the counter should be load with :

MAX_Count -N +1

The source of the pulses may be :

- the internal crystal clock/12
- the external timer input pin T0/T1 (max input = Crystal/24)

The timers are programmed with 6 SFR

- TCON Timer Control Register (Timer Enable bits and Timer Flags)
- TMOD Timer Mode Register (set Timer mode, selects int/ext pulses)
- TL0, TL1 Low 8 bits of Timers 0 and 1
- TH0, TH1 High 8 bits of Timers 0 and 1

MICROCONTROLLERS

Serial Port

The 80C51 has an internal serial port.

- It is configured as a standard asynchronous ***UART*** (*Universal Asynchronous Receive Transmit*)

- Timer 1 must be used to generate the BAUD clock.

- SBUF is the Transmit and Receive Register.
- 2 separate registers, one write only, one read only by program
- When a byte is written to SBUF it is transmitted serially out of Pin TxD
- at the BAUD rate
- preceded by a start bit
- and followed by a stop bit. (total of 10 bits).
- A flag is set when transmission is complete (TI)

MICROCONTROLLERS

- When a byte is received (including the start bit and stop bit)
- it is written into the SBUF Receive Register
- And a Flag is set (RI).

- The TI an RI flags are Or’ed and generate an interrupt.

This is just one example mode of serial port operation. There are other modes of operation of the serial port.