

Example

We need to transfer content of register R0 to register R1. How many ways can we do this?

Write a code fragment to save CY flag in bit location 4, AC flag in bit location 10H and parity flag in location 22H.

Example

Two strings are stored one after another starting from external memory location 6000H. Each string is terminated by a NULL character (an all-zero byte). Write separate subprograms to perform

- (a) String comparison
- (b) String length calculation

The subprograms should use different register banks. Main program should collect the results and comparison result in R0, length of first string in R1 and length of second string in R2 of the default register bank.

Example

Write a program to find LCM of two numbers stored at memory addresses 6000H and 6001H, with the smaller number at 6000H.

Example

Write a program to read an 8-bit pattern from Port1 and output it to Port 2 if the number is odd, to Port 3 if the number is even.

Example

Write a code fragment to output bit 3 of accumulator A to LSB of Port 1. Then rfead MSB of Port 1 and set the LSB of the accumulator to be same as the bit read. Other bits should be left unaffected.

Example

Write a code fragment to forcefully turn off the least significant 4 bits of Port 1, turn on the most significant 2 bits of Port 2 and complement all bits of port 3, irrespective of previous outputs to the ports.

Example

Write a program to transmit continually the string “Hello” serially at 9600 baud rate with 8 bit data and 1 stop bit.

Write a program to operate serial port in mode 1 for receiving serial byte through RxD pin and send the byte to Port 2.

Example

Write a program to get data from P1 and send to P2 continually. Incoming data from serial port be sent to Port P0 simultaneously.

Example

Write a program to receive 1-bit data from Port P0.3 and P1.6. Perform AND of these two bits and output the result to Port P2.3

Write a program that checks the content of Port P0. If the number read is equal to 20H, it sends FFH to Port P0. If it is equal to 30H, 40H or 50H, it sends FFH to port P1, P2 or P3 respectively.

Example

Write a program to read port P0 on occurrence of a level triggered interrupt on INT0 line. Compute the square of the number and store the value in memory locations 50H and 51H.

Example

Write a program to read data from Port P1 and write it serially at baud rate 9600, using interrupt

Example

Write a code fragment to generate 2 sec delay using 12MHz crystal

Write a code fragment to count pulses coming on pin T1. Whenever the count value becomes 100, it complements all bits of Port P1. Initially Port P1 has pattern “10101010”.

Example

Write a program to implement a digital clock. The clock normally displays the time in 24-hour format (hh-mm-ss). On pressing a *mode* button, it shows the current date in dd-mm-yy format. On releasing the button, it comes back to display the time again. The date and time values are stored internally at some memory locations. Three bytes are used to store the date, month and year values. The values of memory locations are updated automatically through a periodic timer interrupt. Assume that there is a routine DISPLAY, which when called with DPTR pointing to the beginning, displays three bytes at 6 successive display modules. Also assume the presence of another routine CALENDER, which when called updates the date value. The bytes corresponding to time are stored at memory locations 8000H, 8001H, 8002H, with hh being in 8000H. The bytes for date are stored at locations 8003H, 8004H and 8005H, with dd being stored at 8003H.

Example

Assume that the oscillator connected to 8051 chip is operating at 12MHz. Write a program to generate 4KHz square wave on pin P1.4 using timer 0.

ARM – An Advanced Microcontroller

Santanu Chattopadhyay

Electronics and Electrical Communication Engineering

Introduction

- **32-bit RISC architecture**
- **Developed by ARM Corporation, previously known as *Acron RISC Machine***
- **Licensed to companies that want to manufacture ARM based CPUs or SOC products**
- **Helps the licensee to develop their own -processors compliant with ARM instruction set architecture**

Features

- ARM cores are very simple, require relatively lesser number of transistors, leaving enough space on die to realize other functionalities on the silicon
- Instruction set architecture and the pipeline design aimed at minimizing energy consumption
- Also capable of running 16-bit THUMB instruction set – greater code density and enhanced power saving
- Higher performance
- Highly modular architecture – the only mandatory part is the integer pipeline, all other components are optional
- Built-in JTAG debug port and on-chip embedded in-circuit emulator (ICE) that allows programs to be downloaded and fully debugged in-system

ARM Architecture History

Version	Year	Features	Implementation
V1	1985	The first commercial RISC (26-bit)	ARM1
V2	1987	Coprocessor support	ARM2, ARM3
V3	1992	32-bit, MMU, 64-bit MAC	ARM6, ARM7
V4	1996	THUMB	ARM7TDMI, ARM8, ARM9TDMI, StrongARM
V5	1999	DSP and Jazelle extensions	ARM10, XScale
V6	2001	SIMD, THUMB-2, TrustZone, Multiprocessing	ARM11, ARM11 MPCore
V7	2005	Three profiles, NEON, VFP	?

ARM7 block diagram

