Microcontroller Firmware procedures

1. Assembly code
2. Filename.asm
3. Assembler
4. Filename.HEX
5. Software Burner
6. Training-KIT

Software Burner
To HW USB
Firmware

Assembler
Assembly code
C or MikroC code
Simulator “as Proteus”

Filename.C Filename.h

To Embedded system

Microprocessor 2017-18
ORG
Directive used by assembler to define the start address of the current machine code at the program memory

Such as

ORG 0
INC A

ORG 100h
DEC A

0000h -> Machine code (INC A)

0100h -> Machine code (DEC A)

Program M
END

- Directive used by assembler to stop compiling.
- Any instruction after “End” consider as comment.

Any instruction after “End ”
Consider as comment

MOV A, R1

Program M
The machine cycle (MC) is the minimum number of CPs (12 CPs) taken by CPU to perform a single instruction cycle (F-D-E).

What is the difference between machine code and machine cycle?
The rate of machine cycles (internal clock pulses)

Crystal Oscillator (Up to 24 MHz) → Internal clock divider by 12 → Rate of the Internal clock pulses

Rate of Machine cycle = \frac{\text{Crystal frequency}}{12} \text{ [MHz]}

ONE MACHINE CYCLE

Clock pulses from the oscillator

Machine cycle pulses
The internal clock pulse

Uses by CPU to synchronize all processes in the microcontrollers

\[ F = \frac{f_{\text{crystal}}}{12} \]
Calculate the rate of the internal clock pulse of AT89C55, if the external crystal has frequency 3 MHz

The answer

The Rate of internal clock pulse = \( \frac{\text{Crystal frequency}}{12} \)

\[
= \frac{3 \times 10^6}{12} = 250 \text{ KHz}
\]
The Execution time

Is time taken by CPU to carry out one instruction

The Run-time (Delay-time)

Is time taken by CPU to carry out program or subprogram
Calculate the total run-time (total delay time) for AT89C51 during execution of 3 instructions that need 1, 2 and 4 machine cycles respectively, the connected crystal has frequency equal 6 MHz.

The answer

\[
\text{Delay time} = \frac{\text{Total number of Machine cycles} \times 12}{\text{Crystal frequency}} \\
= \frac{(4 + 2 + 1) \times 12}{6} = 14 \, \mu\text{sec}
\]
The FET as electronic switch
The internal structure of the ports

Quasi bi-directional ports
Why we write FFh to 8051 ports?
The internal structure of a pin of port1
By Setting the FF (or resetting μC)

- The bit-of-port has output “1”
- it can accept external “1” or “0” after the “READ PIN” is activate
By clearing the Flip-Flop

- The bit-of-port has output “0”
- it can’t accept any external “1”
The most port's instructions activate the "READ PIN" to accept the logical inputs from a port to any other register such as (MOV A, P1) or (OR A, P1).
Few ports' instructions activate the "READ LATCH" to accept logical inputs from a port to itself such as (ORL P1, A).

Before operation:

After operation:

An operation
Why 8051’s ports are Quasi bi-directional

The ports can output either ‘1’ or ‘0’ at any time

The ports can accept input ‘0’ at any time

but

The ports can accept input ‘1’ only if its flip-flop is loaded by ‘1’
The internal memories of 8051
### Microprocessor 2017

#### The special function registers

<table>
<thead>
<tr>
<th>Bit Addressable</th>
<th>SFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>F0</td>
<td>B</td>
</tr>
<tr>
<td>E0</td>
<td>ACC</td>
</tr>
<tr>
<td>D0</td>
<td>PSW</td>
</tr>
<tr>
<td>B8</td>
<td>IP</td>
</tr>
<tr>
<td>B0</td>
<td>P3</td>
</tr>
<tr>
<td>A8</td>
<td>IE</td>
</tr>
<tr>
<td>A0</td>
<td>P2</td>
</tr>
<tr>
<td>99</td>
<td>SBUF</td>
</tr>
<tr>
<td>98</td>
<td>SCON</td>
</tr>
<tr>
<td>90</td>
<td>P1</td>
</tr>
<tr>
<td>8D</td>
<td>TH1</td>
</tr>
<tr>
<td>8C</td>
<td>TH0</td>
</tr>
<tr>
<td>8B</td>
<td>TL1</td>
</tr>
<tr>
<td>8A</td>
<td>TL0</td>
</tr>
<tr>
<td>89</td>
<td>TMOD</td>
</tr>
<tr>
<td>88</td>
<td>TCON</td>
</tr>
<tr>
<td>87</td>
<td>PCON</td>
</tr>
<tr>
<td>83</td>
<td>DPH</td>
</tr>
<tr>
<td>82</td>
<td>DPL</td>
</tr>
<tr>
<td>81</td>
<td>SP</td>
</tr>
<tr>
<td>80</td>
<td>P0</td>
</tr>
</tbody>
</table>

*Note: Certain addresses are not bit addressable.*
The Program-Status-Word Register (PSW)

- 4 status flags (CY, AC, OV, P)
- 2 control flags (RS0, RS1)
- User bit-addressable flag (F0)
- Reversed flag (bit1)
Selection of operational bank

![Diagram showing selection of operational banks with start addresses](image-url)
The Carry and Auxiliary carry flags

Carry flag (CY)  Carry up from D7

Auxiliary Carry flag (AC)  Carry up from D3 → D4

Accumulator
The parity flag (P)

\[ P = \sum_{\text{bit 0}}^{\text{bit 7}} \text{ones in Accumulator} \]

Odd data \(\rightarrow\) \(P = 1\)

Even data \(\rightarrow\) \(P = 0\)

Or

Zero data
Find the status of the flags CY, AC and P after the addition of the two digital values ABh and 38h in AT89C51.

The answer

\[
\begin{array}{c}
\text{Hex} \\
\text{AB} \\
\text{38} \\
\hline
\text{E3}
\end{array}
\begin{array}{c}
\text{Binary} \\
10101011 \\
00111000 \\
11100011
\end{array}
\]

\text{CY} = 0 \text{ because there isn't carry-up from 8}^{\text{th}} \text{ bit of the final result.}

\text{AC} = 1 \text{ because there is carry-up from 4}^{\text{th}} \rightarrow 5^{\text{th}} \text{ bit of the final result.}

\text{P} = 1 \text{ because the Accumulator has odd number of ones (5 ones).}
The Over Flow flags (OV = 1) in three cases

• When the divisor has value zero during the division operation.
• When the multiplication product is greater than the value "FFh".
• When an error occurs during the addition and subtraction operations for the signed numbers (the result outside the signed 8-bit range "-128" to "127").
Find the status of the flags CY and OV after the addition of the two digital values \( F4h \) and \( 80h \) in the AT89C51.

The answer

\[
\begin{array}{c|c}
\text{Hex} & \text{Binary} \\
\hline
F4 & 1111 0100 \\
+ 80 & 1000 0000 \\
\hline
174 & 0111 0100 \\
\end{array}
\]

- CY = '1'
- OV = '1' xor '0' = '1'

The answer is:

- CY = negative
- OV = negative
- Result = positive
The PC points to the memory location to fetch current instruction.
Accessing the Internal Program memory of 8051

**Diagram:**

- **A+ DPTR** connected to **Internal Address bus**
- **Internal Program Memory** (Flash type)
  - (4 Kbyte)
- **Internal Data bus** connected to **Accumulator**

**Code:**

`MOVC A, @A+DPTR`

**Notes:**
- **Destination**
- **Pointers**
Example

MOVC A, @A+DPTR

The pointer $\rightarrow$ A+DPTR = 24h + 0011h = 35h
After executing the instruction `MOVC A, @A+DPTR`