

Secured 8-bit microcontroller

83C852

FEATURES

- 8-bit CPU
- 6K bytes of user program memory (ROM), no external extension
- 256 bytes of RAM data memory (RAM), no external extension
- 2K bytes EEPROM:
 - EEPROM stores data or program
 - on-chip voltage multiplier for EEPROM ERASE/WRITE
 - ERASE/WRITE cycle time independent of the clock frequency
 - 10000 ERASE/WRITE cycles per bytes
 - 10 years non-volatile data retention
 - infinite number of READ cycles
 - error code correction
- Calculation unit for cryptographic calculations
- Security features
- Power-ON/OFF reset circuit
- Low frequency detector
- Two 16-bit timers
- Clock frequency range 1 MHz to 6 MHz; 1 μ s cycle time with 6 MHz clock frequency
- Two I/O lines; only one I/O line is used in half-duplex, according to the ISO standards for the Smart Card applications; full-duplex communication can be performed with both I/O lines
- 5 interrupt sources from: I/O lines; Timer 0; Timer 1; EEPROM; Calculation unit
- Power-down and idle mode
- Two operating modes: test mode and user mode
- Single 5 volts power supply
- 6 pins: V_{DD} , V_{SS} , I/O1, I/O2, RESET, CLK

QUICK REFERENCE DATA

| SYMBOL | PARAMETER | CONDITION | MIN. | MAX. | UNIT |
|-----------|-------------------------------------|----------------------|------|------|--------------|
| V_{DD} | supply voltage range | | 4.5 | 5.5 | V |
| I_{DD} | supply current | $f_{CLK} = 3.57$ MHz | – | 10 | mA |
| I_{DD} | supply current: operating modes | $f_{CLK} = 6.0$ MHz | – | 15 | mA |
| I_{ID} | supply current: idle mode | $f_{CLK} = 6.0$ MHz | – | 3 | mA |
| P_{tot} | total power dissipation | | – | 1 | W |
| T_{stg} | storage temperature range | | –65 | 150 | $^{\circ}$ C |
| T_{amb} | operating ambient temperature range | | 0 | 70 | $^{\circ}$ C |

GENERAL DESCRIPTION

The 83C852 single chip secured microcontroller is manufactured in an advanced 1.2 μ COS process. It is a derivative of the 80C51 microcontroller family and has the same instruction set as the 80C51. It has been specially designed for conditional access in secure Smart Card applications and is implemented with the highest levels of security.

Its internal calculation unit speeds-up cryptographic calculations using Public Key Algorithms.

Cryptographic calculations

| |
|--|
| At $f_{CLK} = 6$ MHz: $X^e \text{ mod.} n$ is performed in 1.5 s typical, with 512 bit operands. |
|--|

External communications can be performed through a serial interface (I/O) according to ISO standards. The serial interface must be controlled by application software for access to the 83C852 internal memory.

The 83C852 contains a 6K bytes READ only memory (user ROM); a 256 bytes READ/WRITE data memory (RAM); 2K bytes electrically erasable programmable READ only memory (EEPROM); two I/O lines; two 16-bit timers; five vectorized interrupt sources; 33 Special Function Registers (SFRs) and a Calculation Unit to speed-up the execution time of public keys and secret keys cryptographic algorithms.

The 83C852 operates with a single 5 volts power supply and at a maximum clock frequency of 6 MHz. The instruction set consists of over 100 instructions: 49 one-byte, 45 two-byte and 17 three-byte. With an input clock frequency of 6 MHz, 58% of the instructions are executed in 1 μ s and 40% in 2 μ s.

Secured 8-bit microcontroller

83C852

ORDERING INFORMATION

| EXTENDED TYPE NUMBER | PACKAGE | | | |
|----------------------|---------|-----------------|-------------|------|
| | PADS | PAD POSITION | MATERIAL | CODE |
| 83C852 die | 6 | X Y coordinates | die | - |
| 83C852P | tbf | tbf | plastic DIL | tbf |

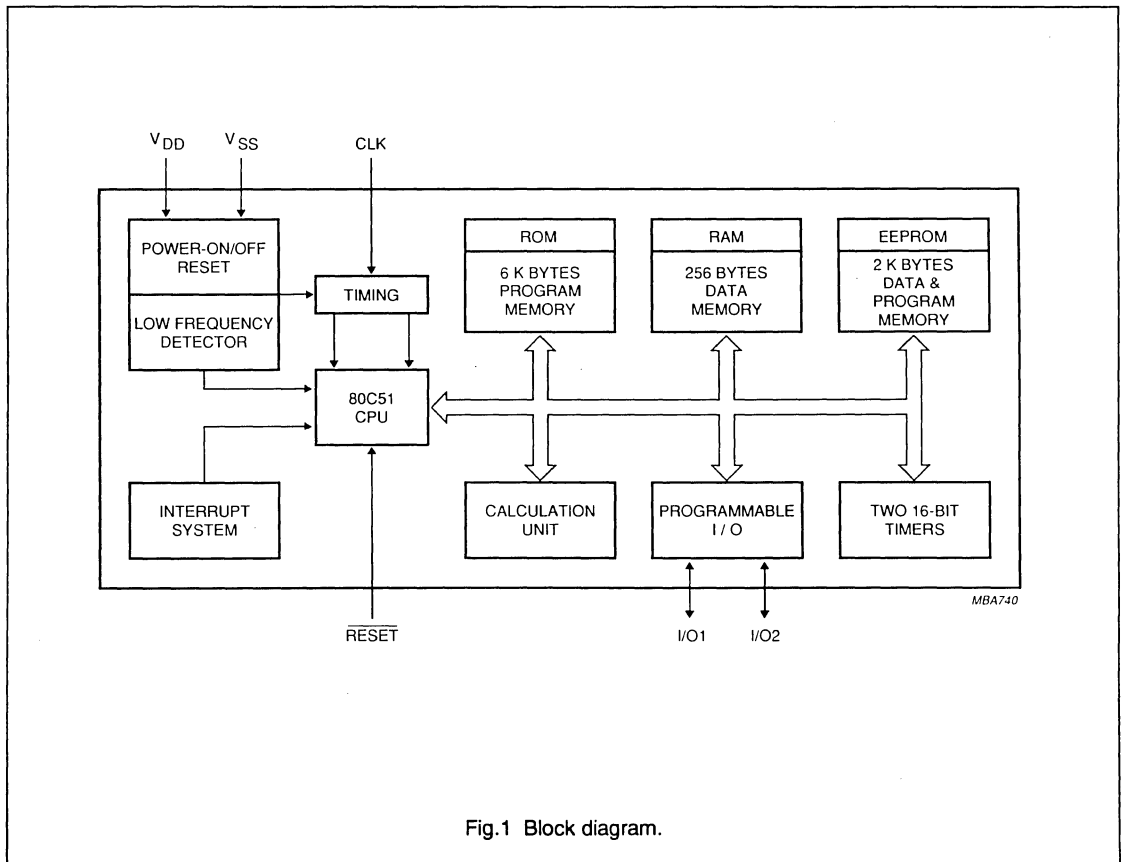


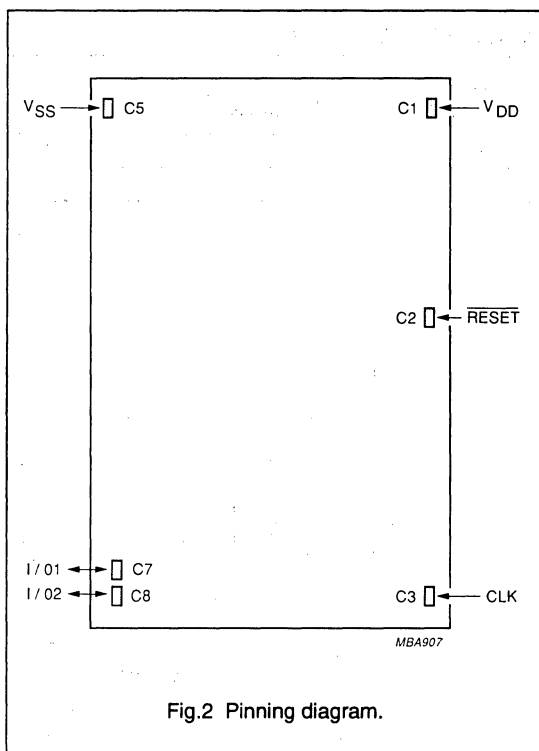
Fig.1 Block diagram.

Secured 8-bit microcontroller

83C852

PINNING

| SYMBOL | PIN (PAD) | DESCRIPTION |
|-----------------|-----------|--|
| V _{DD} | C1 | +5 volts power supply pin during normal operation, idle mode and power-down mode |
| RESET | C2 | active LOW input that initializes the processor |
| CLK | C3 | external clock input. The internal clock frequency = the external clock frequency |
| V _{SS} | C5 | ground |
| I/O1 | C7 | quasi bi-directional port (TTL compatible), the user's program must include routines able to handle an asynchronous serial communication through a single I/O port (for half-duplex) |
| I/O2 | C8 | quasi bi-directional port (TTL compatible) |



ASSIGNMENT OF ISO/83C852 SMART CARD CONTACTS

| CONTACTS | ISO ASSIGNMENTS | 83C852 ASSIGNMENTS |
|----------|-----------------|--------------------|
| C1 | V _{CC} | V _{DD} |
| C2 | RESET | RESET |
| C3 | CLK | CLK |
| C4 | reserved | not connected |
| C5 | GND | V _{SS} |
| C6 | V _{PP} | not connected |
| C7 | I/O | used for I/O1 |
| C8 | reserved | reserved for I/O2 |

| | |
|----|----|
| C1 | C5 |
| C2 | C6 |
| C3 | C7 |
| C4 | C8 |

MBA902

contact assignments are specified in part 2 of ISO 7816

Fig.3 Contact assignments.

Secured 8-bit microcontroller

83C852

FUNCTIONAL DESCRIPTION**General**

The 83C852 is specially designed for secured applications such as conditional access and transactions in a Smart Card environment. It has a Calculation Unit which makes the microcontroller dedicated to asymmetric crypto systems. Special Function Registers (SFRs) are available to the user to manipulate memory transactions and calculation unit operations.

The address bus of the EEPROM is mixed to prevent fraudulent access and optical scanning. The EEPROM has an hardware error code correction which guarantees the data content integrity. The 83C852 is able to read and modify a part of the internal program memory contained in EEPROM.

The 83C852 has two software selectable modes of reduced activity for further power reduction, the idle mode and power-down mode:

- the idle mode freezes the CPU while allowing the RAM, the timers and the interrupt system to continue functioning.
- the power-down mode saves the RAM content and disables all other chip functions.

The 83C852 has 33 SFRs available for use by the user (see Table 23). The functional descriptions and usage of the SFRs as they co-relate to the RAM, EEPROM and the calculation unit activities, are described within the following sections.

Memory organisation

The central processing unit (CPU) manipulates operands in three memory spaces, (see Fig. 4) these are:

- 6K-byte internal program memory (ROM)
- 2K-byte program and data memory (EEPROM)
- 256-byte internal data memory (RAM).

The 256-byte internal RAM memory address space is sub-divided into:

- 128-byte internal data RAM locations 00 to 7FH. This address space is accessible with direct and indirect addressing
- 128-byte internal data RAM locations 80H to FFH. This address space is accessible with indirect addressing only

- 128-byte Special Function Register (SFR) address space 80H to FFH. This address space is parallel to the upper 128 byte RAM. It is accessible with direct addressing only. 33 SFRs reside inside this area, the remaining address space in between is unused.

EEPROM

The EEPROM has a capacity of 2K bytes (words) see Fig. 5. With its built-in error correction hardware the EEPROM is a very reliable non-volatile memory. In addition to each single stored data byte, 4 extra "parity" bits are stored in EEPROM. Single-bit errors per byte are automatically corrected when reading the memory. It can be accessed by both CPU and calculation unit (however, not at the same time).

Programming of the EEPROM is completely controlled by the EEPROM's sequencer. The EEPROM can be used either both as data memory and program memory for the CPU, or as data memory for the calculation unit.

EEPROM AS DATA MEMORY:

When the CPU executes opcodes from internal ROM (program address < 8000H), the EEPROM can be used as a data memory. The communication between CPU and EEPROM is performed via 6 SFRs (see Table 1), these comprise:

- 2 EEPROM (SFRs) address registers (HIGH and LOW address byte)
- 1 EEPROM (SFR) data register for READ and WRITE operations
- 2 EEPROM (SFRs) control registers to select the various operating and test modes
- 1 EEPROM (SFR) timer register to adapt the ERASE/WRITE time to the operating clock frequency.

EEPROM AS PROGRAM MEMORY:

When the program counter is higher than 7FFFH, the EEPROM is used as a program memory. The CPU fetches opcodes directly from the EEPROM. EADRH, EADRL1 and EDAT registers cannot be written. Their contents in this mode are irrelevant.

Reading data from the EEPROM can still be done with the MOVc instruction, but EEPROM write operation is not possible. The EEPROM can only be written by software executed from the ROM area (program address < 8000H).

Secured 8-bit microcontroller

83C852

EEPROM AS DATA MEMORY FOR THE CALCULATION UNIT:

Communication between calculation unit and EEPROM is performed via SFRs (see Table 1), these comprise:

- 2 EEPROM SFRs address registers (one HIGH and one of two alternate LOW address bytes)
- calculation unit SFRs.

The EEPROM data output is directly connected via a special bus to the data registers of the calculation unit. The calculation unit has direct READ access to the EEPROM.

During memory access, the EEPROM is addressed by one of two address pointers EADRL1 or EADRL2. Both

pointers are loaded by the CPU and decremented by the calculation unit's sequencer. EADRL1 or EADRL2 supply the LOW byte (LSB) of the EEPROM address. The HIGH byte (MSB) of the EEPROM address is taken directly from the EADRH register. The access to the EEPROM from the calculation unit, is controlled by the calculation unit SFRs.

When the access to EEPROM by the calculation unit is active, the EEPROM is not accessible by the CPU, neither as data memory nor as program memory. When the calculation unit is operating, but not accessing the EEPROM, the CPU can READ, WRITE and EXECUTE EEPROM.

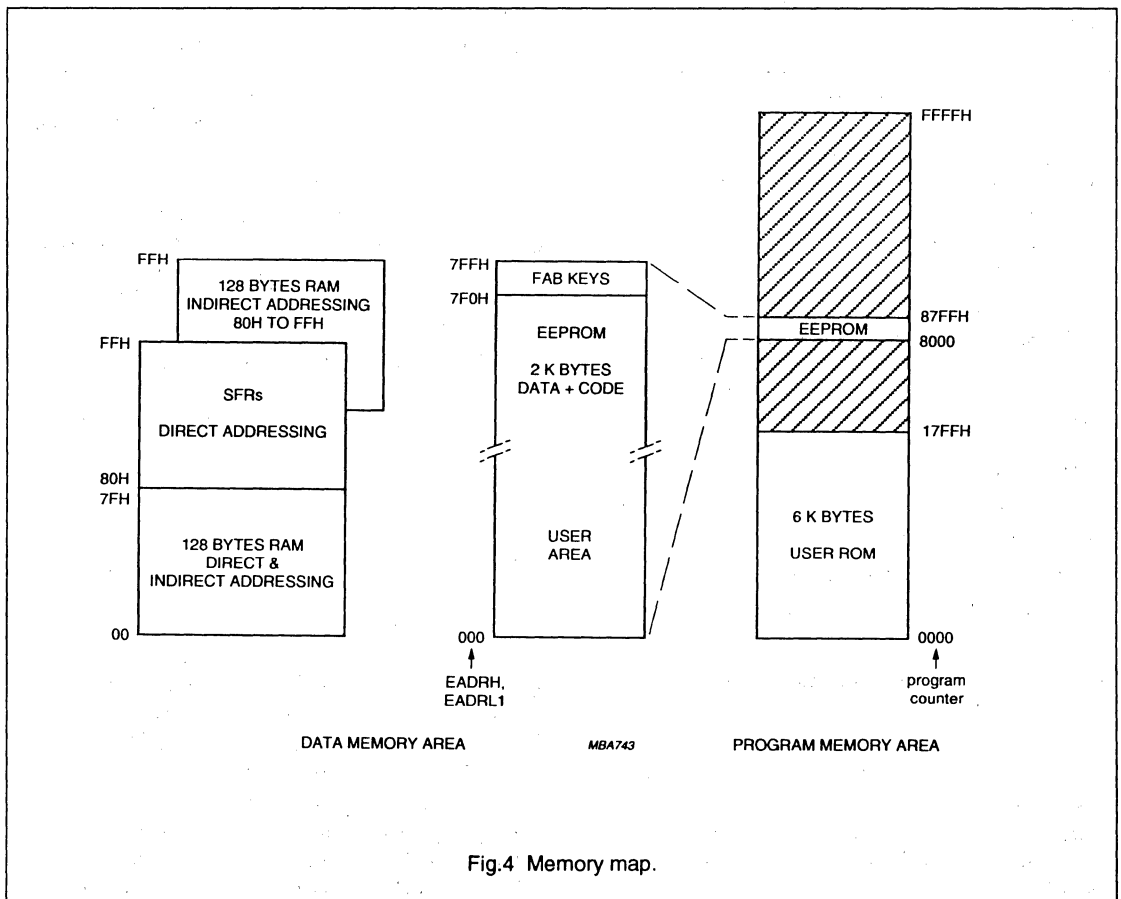


Fig.4 Memory map.

Secured 8-bit microcontroller

83C852

EEPROM SFRs

Table 1 provides a listing of the EEPROM associated SFRs:

Table 1 EEPROM SFRs

| NAME | SFR ADDRESS | FUNCTION |
|---|-------------|---|
| The communication between the CPU and the EEPROM is performed via the following SFRs: | | |
| EADRL1 | 0F2H | address register (LSB) |
| EADRH | 0F3H | address register (MSB) |
| EDAT | 0F4H | data register |
| ETIM | 0F5H | EEPROM timer register |
| ECNTRL1 | 0F6H | control register for normal operation modes |
| ECNTRL2 | 0F7H | control register for special test modes |
| The communication between the calculation unit and the EEPROM is performed via the following SFRs: | | |
| EADRL2 | 0F1H | address pointer 2: LSB of the EEPROM address (WRITE: reload EADRL2 register; READ: read counter) |
| EADRL1 | 0F2H | address pointer 1: LSB of the EEPROM address (WRITE: reload EADRL1 register; READ: read counter) |
| EADRH | 0F3H | address register (MSB) |
| Two address pointers (SFRs) are necessary for EEPROM access by the calculation unit. SFRs EADRH and either EADRL1 or EADRL2 | | |

EEPROM SFR descriptions

EADRH

EADRH is an 8-bit register (SFR), used as an address pointer, it is loaded from the CPU and contains the highest byte (MSB) of the EEPROM address. Only bits EADRH.7, 2, 1, 0 are relevant. Default value after reset is '10000000B'.

EADRL1, EADRL2

EADRL1 is an 8-bit register (SFR), used as an address pointer, it is loaded from the CPU. Its contents are transferred into an 8-bit down counter which provides the LOW byte (LSB) of the EEPROM address. The CPU has WRITE access to the EADRL1 register and READ access to the down counter.

Default value after reset of both the EADRL1 register and associated down counter are 00H. The transfer of EADRL1 to the down counter and the decrement are controlled by the calculation unit. The behaviour of EADRL1 depends on whether or not there is an access in progress from the calculation unit to the EEPROM.

No access from the calculation unit to the EEPROM:

- when there is no active access to the EEPROM from the calculation unit, the contents of the EADRL1 register are continuously loaded into its associated down counter. There is no hardware-decrement of the down counter. The combination of EADRL1 plus its down counter behaves therefore as a normal register. In this mode, the LOW address byte of the EEPROM is always supplied by EADRL1. EADRL2 behaves similar to EADRL1 but it is not used for EEPROM addressing. Its contents are irrelevant.

Calculation unit access to the EEPROM:

- EADRL1 is the EEPROM LOW address pointer used to address an operand (Ai) stored in the EEPROM. At the beginning of the calculation unit's computation cycle, the address content of the EADRL1 is loaded into its associated down counter. During the calculation, further EADRL1 address transfers to the down counter stop, whilst the down counter is decremented by the calculation unit's sequencer. During a calculation, the EADRL1 register can be reloaded from the CPU with a new address. This new address will then be used during the next calculation.

Secured 8-bit microcontroller

83C852

- EADRL2 (SFR) is the second EEPROM LOW address pointer which is required for some operations of the calculation unit. It is used to address an operand (Xi) stored in the EEPROM. The function of EADRL2 is similar to EADRL1 function. EADRL2 cannot be used as a second address register for normal CPU access to the EEPROM.

Data register EDAT

This register (SFR) is used to read data from the currently addressed EEPROM byte. When EDAT is written during BYTE MODE, it's contents will be programmed into the addressed EEPROM byte. When EDAT is written during ROW ERASE or BLOCK ERASE mode, the ROW ERASE or BLOCK ERASE operation is

started. In this mode, the data written to EDAT is then irrelevant. The ECNTL1 status bits EWP and IFE indicate whether the EEPROM ERASE/WRITE operation is still active. Whilst the EEPROM programming is in progress, rewriting data to EDAT is not allowed.

Timer register ETIM

The ETIM timer register (SFR) is required to adapt the ERASE/WRITE time to the clock frequency. ERASE (t_e) and WRITE (t_w) times of 5 ms each are required. The user has to ensure that the ERASE or WRITE time is neither too short nor too long. Table 2 gives values for ETIM register for given clock frequencies. ETIM has to be loaded by software in advance to the first ERASE/WRITE operation. ETIM's default value after reset is '08H'.

Table 2 ETIM timer values

The general formula is: Value (decimal) = $(f_{CLK} \text{ kHz}/102.4) - 2$

| OPERATING f_{CLK} MHz | VALUES FOR ETIM | | | |
|----------------------------|-----------------|------|-------------|---------|
| | BINARY | | HEXADECIMAL | DECIMAL |
| | MSB | LSB | | |
| 1.0 | 0000 | 1000 | 8 | 8 |
| 2.0 | 0001 | 0010 | 12 | 18 |
| 3.0 | 0001 | 1011 | 1B | 27 |
| 3.57 | 0010 | 0001 | 21 | 33 |
| 4.0 | 0010 | 0101 | 25 | 37 |
| 4.92 | 0010 | 1110 | 2E | 46 |
| 5.0 | 0010 | 1111 | 2F | 47 |
| 6.0 | 0011 | 1001 | 39 | 57 |

Secured 8-bit microcontroller

83C852

Control register ECNTRL1

ECNTRL1 is the control register (SFR) for the several user operation modes of the EEPROM.

Table 3 ECNTRL1 SFR

| IFE | EEINT | EWP | – | OPERATION MODE SELECT | | | |
|-----|-------|---------------|---|-----------------------|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | BYTE MODE → | | 0 | 0 | 0 | 0 |
| | | ROW ERASE → | | 1 | 1 | 0 | 0 |
| | | BLOCK ERASE → | | 1 | 0 | 1 | 0 |
| | | TEST MODE → | | 1 | 1 | 1 | 1 |

Table 4 Description of the ECNTRL1 bits

| SYMBOL / PARAMETER | FUNCTION |
|--|--|
| ECNTRL1.7 | |
| IFE, interrupt flag EEPROM | set by the EEPROM sequencer after completion of an EEPROM WRITE access, or set and reset by software. When (IFE is set 1 and EEINT is set 1), an interrupt request is done. Interrupt vector 0023H will be forced if the bits EA and EE inside the interrupt Enable Register IE are also set 1. |
| ECNTRL1.6 | |
| EEINT, enable EEPROM interrupt | set and reset by software. Enables an EEPROM interrupt request when HIGH. |
| ECNTRL1.5 | |
| EWP, ERASE/WRITE in progress | set and reset by the EEPROM sequencer. EWP is active HIGH during EEPROM write operations. Consecutive write operations to EDAT are not allowed as long as EWP is set. EWP cannot be set or reset by software. |
| ECNTRL1.4 | |
| – | reserved. |
| ECNTRL1 bits 3, 2, 1, 0 | |
| operation mode select: BYTE MODE (0000) | normal E ² PROM mode, default mode after reset. In this mode READ or WRITE access to one byte at a time is possible. |
| READ mode | This is the default mode when BYTE MODE is selected. The contents of the addressed byte are available in the data register EDAT. |
| WRITE mode | This mode is activated after loading of the data register EDAT with the data byte to be written. Before writing EDAT, the address registers EADRL1 and EADRH must be loaded first. Depending on the previous contents of the addressed memory cells, the EEPROM sequencer decides whether to do a WRITE cycle (t_w), or a combined ERASE/WRITE cycle ($t_e + t_w$). A WRITE cycle is carried out when the previous memory content has been 00H. Otherwise an ERASE/WRITE cycle is carried out. |

Secured 8-bit microcontroller

83C852

| SYMBOL / PARAMETER | FUNCTION |
|--|---|
| operation mode select: ROW ERASE (1100) | in this mode the contents of the addressed memory row will be erased. The three LSB's of EADRL1 are not significant, i.e. 8 bytes addressed by EADRL1 will be cleared in the same time normally needed to clear one single byte ($t_{\text{ROW ERASE}} = t_w$). The ROW ERASE operation can be started by writing the EDAT register. The data that is written to EDAT is not significant. Writing of the erased 8 memory cells then takes only 8 WRITE cycles. This is much faster than writing 8 data bytes without a previous ROW ERASE. Such an operation would have taken a total = $8 t_e + 8 t_w$. |
| operation mode select: BLOCK ERASE (1010) | in this mode all memory cells of the EEPROM will be cleared. The BLOCK ERASE operation can be started by writing EDAT. The contents of the data and address registers EDAT, EADRL1, EADRH are don't care. |
| operation mode select: TEST MODE (1111) | the selection of a specific EEPROM TEST MODE within the ECNTRL2 register is only possible when the ECNTRL1 register is switched in advance to TEST MODE. |

Control register ECNTRL2

ECNTRL2 is the control register (SFR) for the several test modes of the EEPROM.

Table 5 ECNTRL2 SFR

| READ ONLY | | | | READ/WRITE | | | |
|---------------------------------------|-----|-----|-----|------------|-----|-----|-----|
| SP3 | SP2 | SP1 | SP0 | TM3 | TM2 | TM1 | TM0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| NO TEST → | | | | 0 | 0 | 0 | 0 |
| READ EEPROM WITHOUT ERRORCORRECTION → | | | | 1 | 0 | 1 | 1 |

Table 6 Description of the ECNTRL2 bits

| SYMBOL / PARAMETER | FUNCTION |
|--------------------------------|---|
| ECNTRL2 bits 7, 6, 5, 4 | |
| SP3, SP2, SP1, SP0 | this part of the ECNTRL2 register is READ only. The upper 4 bits of ECNTRL2 carry either the syndrome word which is generated by the EEPROM error correction logic or the parity bits stored in parallel to the data word in EEPROM memory. The syndrome word is always output during BYTE MODE READ ($EWP = 0$). A value of '0000B' means that no error has been detected/corrected. The parity bits are output during READ EEPROM WITHOUT ERRORCORRECTION TESTMODE or while $EWP = 1$. |
| ECNTRL2 bits 3, 2, 1, 0 | |
| TM3, TM2, TM1, TM0 | this part of ECNTRL2 register is READ/WRITE. The lower 4 bits of ECNTRL2 are used to select one of the EEPROM test-modes. The selection of any test-mode is only possible if ECNTRL1 has been set to 'XXXX1111B' before. Otherwise TM3, TM2, TM1, TM0 are held at '0000B'. |

Secured 8-bit microcontroller

83C852

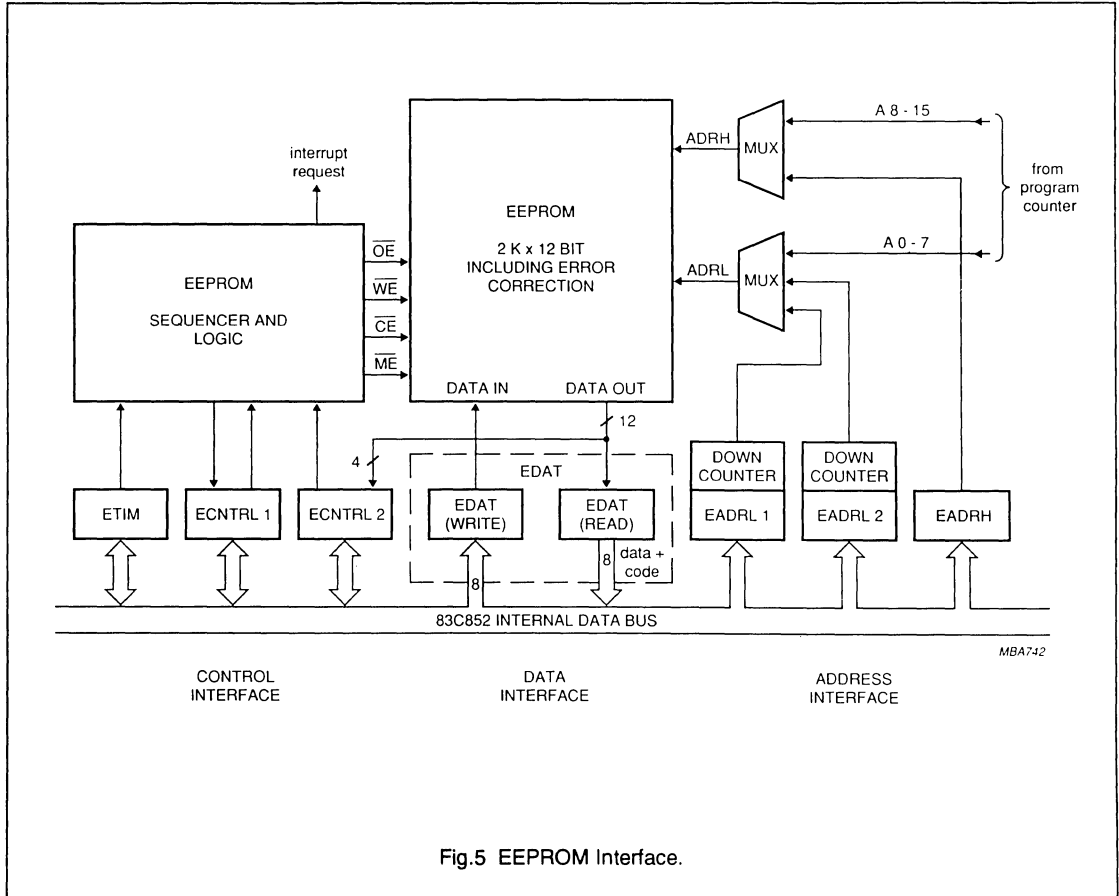


Fig.5 EEPROM Interface.

Secured 8-bit microcontroller

83C852

RAM

The RAM has a capacity of 256 bytes. It can be accessed by both CPU and calculation unit (however, not at the same time). The CPU has full READ and WRITE access to the RAM only when the calculation unit is not operating.

When the calculation unit is operating, it reads and writes data from/to the RAM via a separate channel. In order to avoid possible access conflicts, the CPU cannot read or write the RAM at this time. This condition places some restrictions to the operations of the CPU:

- no subroutine calls possible while calculation unit is active
- register operations (e.g. MOV A,R0) are not possible
- stack pointer operations (PUSH, POP) are not possible
- interrupt requests are not granted while the calculation unit is active.

In the RAM address space, only the SFRs can be read and written by the CPU whilst the calculation unit is active.

RAM address pointers

The calculation unit uses 4 RAM pointers to address the RAM during its direct memory access: AIPR, XIPR, AOPR and APR.

ADDRESS POINTERS AIPR, XIPR, AOPR

AIPR and XIPR pointers address the operand fields A_i resp. X_i inside RAM, while AOPR addresses a RAM area A_o where the calculation result is to be stored. Each pointer consists of an 8-bit register with associated 8-bit down counter. The counter provides a RAM address. It is parallel loaded from its register.

The CPU has WRITE access to the registers and READ access to the counters. Default values after reset for all registers and counters are 00H. The data transfer from the registers to their down counters and the counter decrement are controlled by the calculation unit.

ADDRESS POINTER APR

This is an 8-bit up counter which is used to address the $A_{[3-0]}$ operand field in RAM. It can be read and written by the CPU. Default value after reset is 00H. APR is incremented under the control of the calculation unit.

RAM address pointer modes

The following RAM address pointer modes apply:

The calculation unit is at standby:

- there is no direct memory access from the calculation unit
- the down counters are continuously loaded from the AIPR, XIPR and AOPR registers
- because RAM is not addressed by any of the RAM pointers, their contents are irrelevant at this time. In advance to a calculation, the CPU has to load the RAM pointers with the start addresses of the operand and result fields. AIPR, XIPR and AOPR have to be loaded with the LSB's address, while APR has to be loaded with the MSB's address of the data field.

The calculation unit is active:

- there is direct RAM memory access from the calculation unit
- the data transfer from the registers to their associated down counters is stopped
- the address pointers AIPR, XIPR and APR address the operands A_i , X_i and $A_{[3-0]}$ while AOPR addresses the A_o result area in RAM
- while the up and down counters are incremented/decremented under control of the calculation unit, the CPU can reload the registers with new addresses, ready for transfer to their down counters at the beginning of the next calculation cycle.

Secured 8-bit microcontroller

83C852

CALCULATION UNIT

This unit computes, with its associated software, any exponential functions like $x^e \text{ mod } n$. It has been designed to optimize the calculation time of exponent modulo N. It uses 196 bytes of RAM for 512-bit length operands.

CALCULATION UNIT PERFORMANCE

At $f_{\text{CLK}} = 6 \text{ MHz}$: $X^e \text{ modulo } N$ is performed in 1.5 s typical, with 512 bit operands.

To reach this speed, the calculation unit's architecture provides:

- fast multiplication and addition
- fast carry handling
- fast data transfers to fetch operands from RAM or EEPROM and to store results in RAM
- simultaneous operation of both CPU and calculation unit.

The calculation unit does not carry out a complete exponentation in one step. However, it provides a set of basic instructions, from which the complete exponentation algorithm can be built by a dedicated software. All of these basic instructions operate on data-fields inside RAM and EEPROM. The width of these data-fields is variable. A typical operand width is 512 bits.

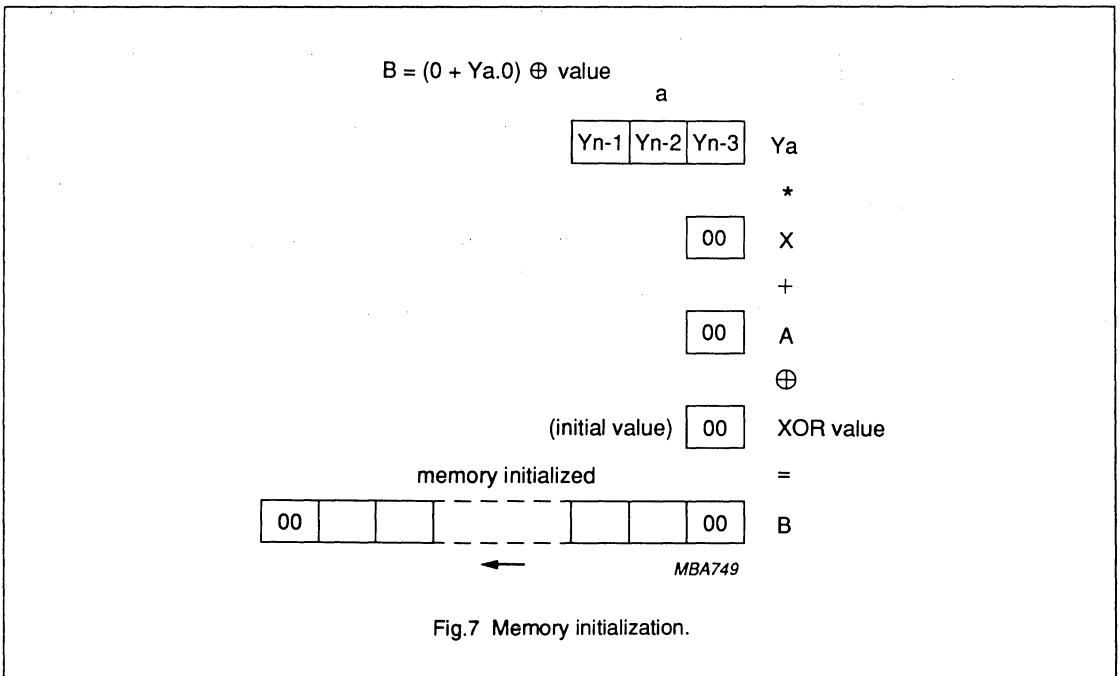
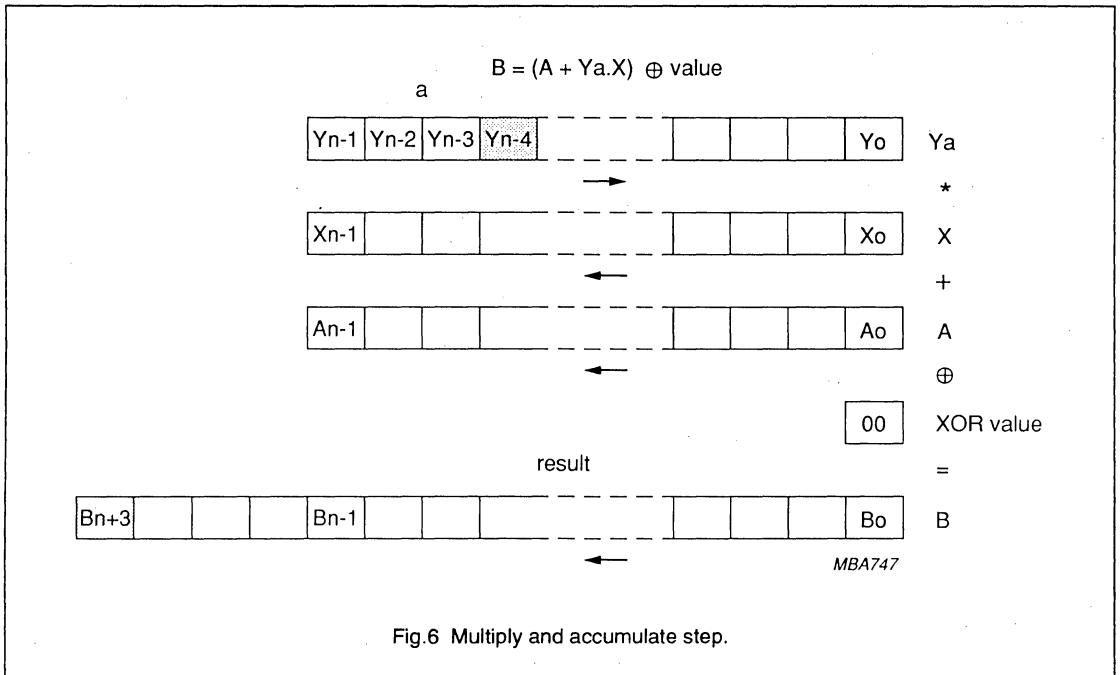
The basic operation of the calculation unit is to multiply either a 24-bit number or a 32-bit number with a long-word (e.g. 512-bit) and adding the result to another long-word. Further XOR and shift operations may be carried out to give the final result. With a 32-bit number this operation completes in typically 45 μs at 6 MHz clock frequency.

CALCULATION UNIT BASIC OPERATION

| | |
|--|--|
| The basic operation of the calculation unit is: | |
| $B = [(A + a * X) \oplus \text{value}] * 2^n$ | Where "A","B" and "X" are large numbers, "a" is a 3 or 4 byte part of the large number "Y" and "value" is either one same byte used for each result byte or the large number "X". "n" is the number of bit-shifts for the calculation result, n can be either 0 or 32. |
| Operation set of the calculation unit: | |
| $B = (A + a * X) \oplus \text{value} * 2^0$ | multiply and accumulate step (see Fig. 6). |
| $B = (0 + a * 0) \oplus \text{value} * 2^0$ | memory initialization (see Fig. 7). |
| $A = (A + a * 0) \oplus \text{value} * 2^{32}$ | 4 byte shift (see Fig. 8). |
| $B = (A + a * 0) \oplus \text{value} * 2^0$ | memory transfer (see Fig. 9). |
| $B = (A + a * X) \oplus \text{value} * 2^{32}$ | multiply, accumulate step with shift (see Fig. 10). |
| Force $A = 0$ or $X = 0$ and shift depend on the contents of the registers CMD and CMDSTAT. | |
| A calculation example of $D = M * C \text{ Mod } N$ is shown in Fig.11, where D, M, C and N are large numbers of n-bit length. | |

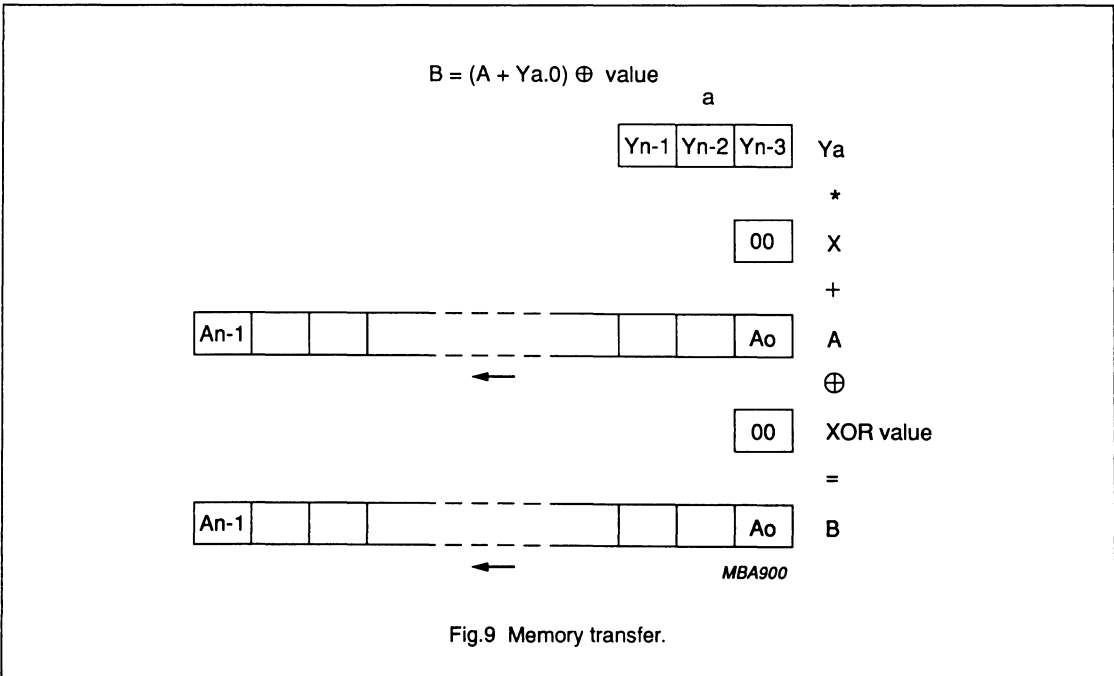
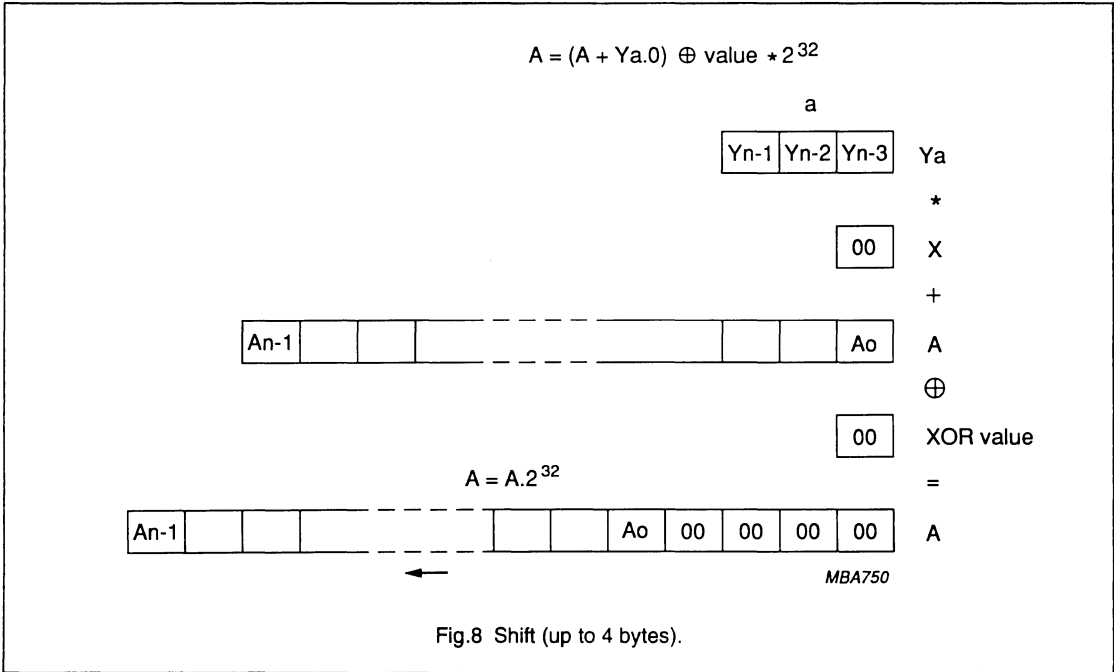
Secured 8-bit microcontroller

83C852



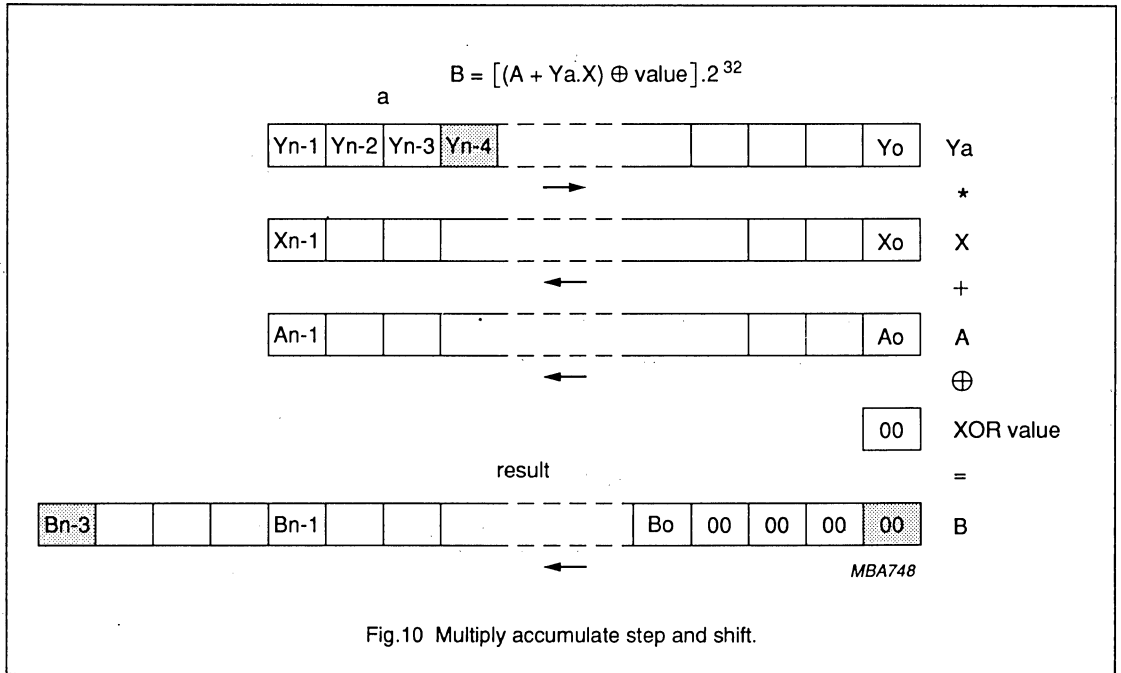
Secured 8-bit microcontroller

83C852



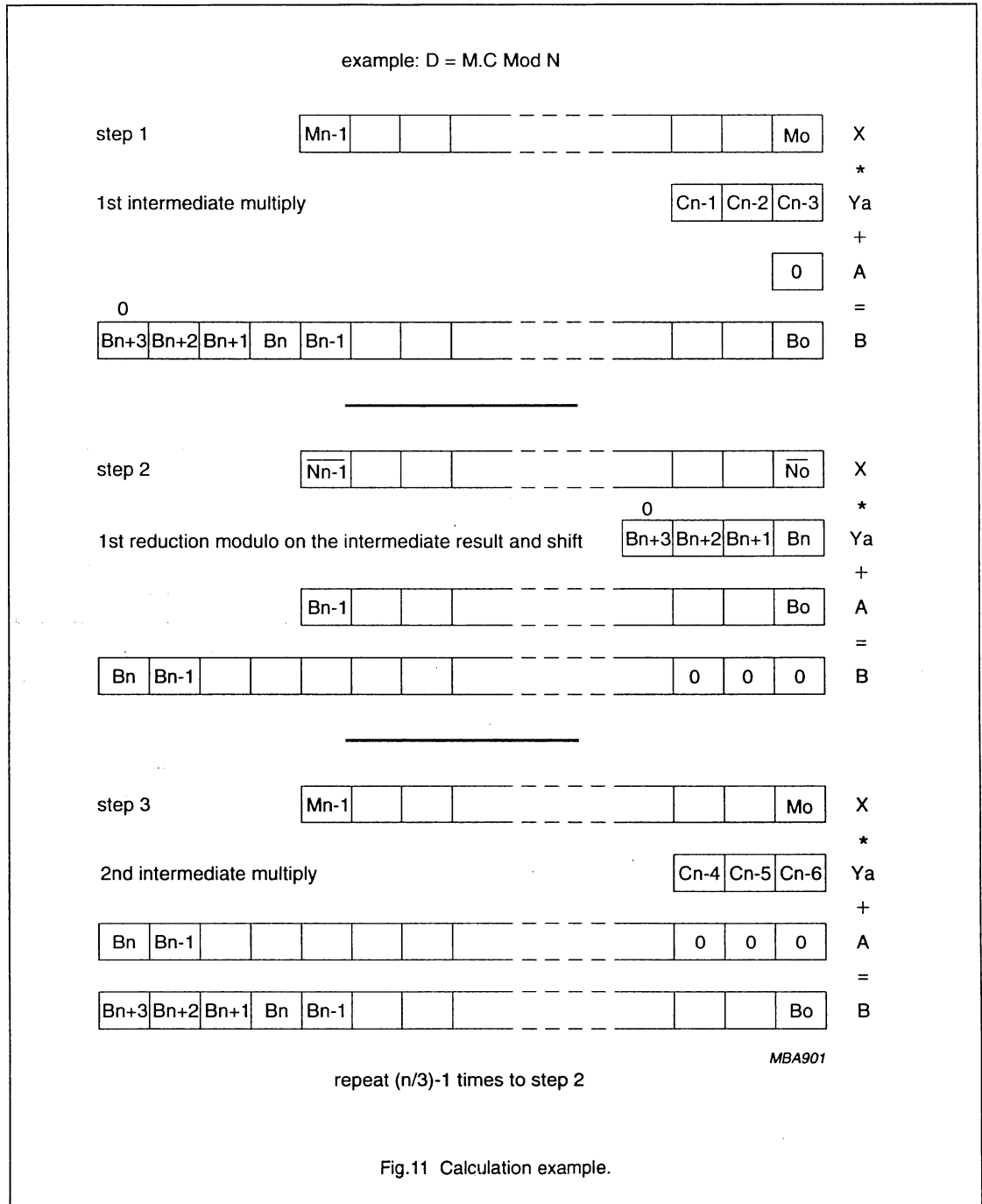
Secured 8-bit microcontroller

83C852



Secured 8-bit microcontroller

83C852



Secured 8-bit microcontroller

83C852

Calculation unit related SFRs

12 Special Function Registers (SFRs) are related to the calculation unit, see Table 7 and Figure 12.

Memory access registers

The calculation unit has direct READ and WRITE memory access to the RAM and READ-only access to the EEPROM. The unit uses four 8-bit wide RAM pointers to address operands and result inside RAM and two 8-bit pointers to address operands inside EEPROM. The MSB of the EEPROM address is supplied by the EADRH register. 5 out of the 6 pointers are pipelined. This allows the CPU to initialize these registers while a calculation is busy.

Table 7 Calculation unit related SFRs

| SYMBOL | ADDRESS | FUNCTION |
|---------|---------|---|
| AIPR | A4H | RAM address pointer for Ai input operand; note 1 |
| EADRL1 | F2H | EEPROM address pointer for Ai input operand; note 1 |
| XIPR | A5H | RAM address pointer for Xi input operand; note 1 |
| EADRL2 | F1H | EEPROM address pointer for Xi input operand; note 1 |
| APR | A6H | RAM address pointer for A[3-0] input operand |
| AOPR | A7H | RAM address pointer for Ao result output |
| CMD | 99H | command register |
| CMDSTAT | 98H | command and status register |
| CNTCYCL | F9H | cycle counter |
| CXOR | A3H | XOR operand register |
| WRLIM | FBH | WRITE operation limit register |
| RDLIM | FAH | limits register of the READ operands Xi and Ai. |

Note

1. Ai operand and Xi operand can be stored in either the RAM or the EEPROM.

Secured 8-bit microcontroller

83C852

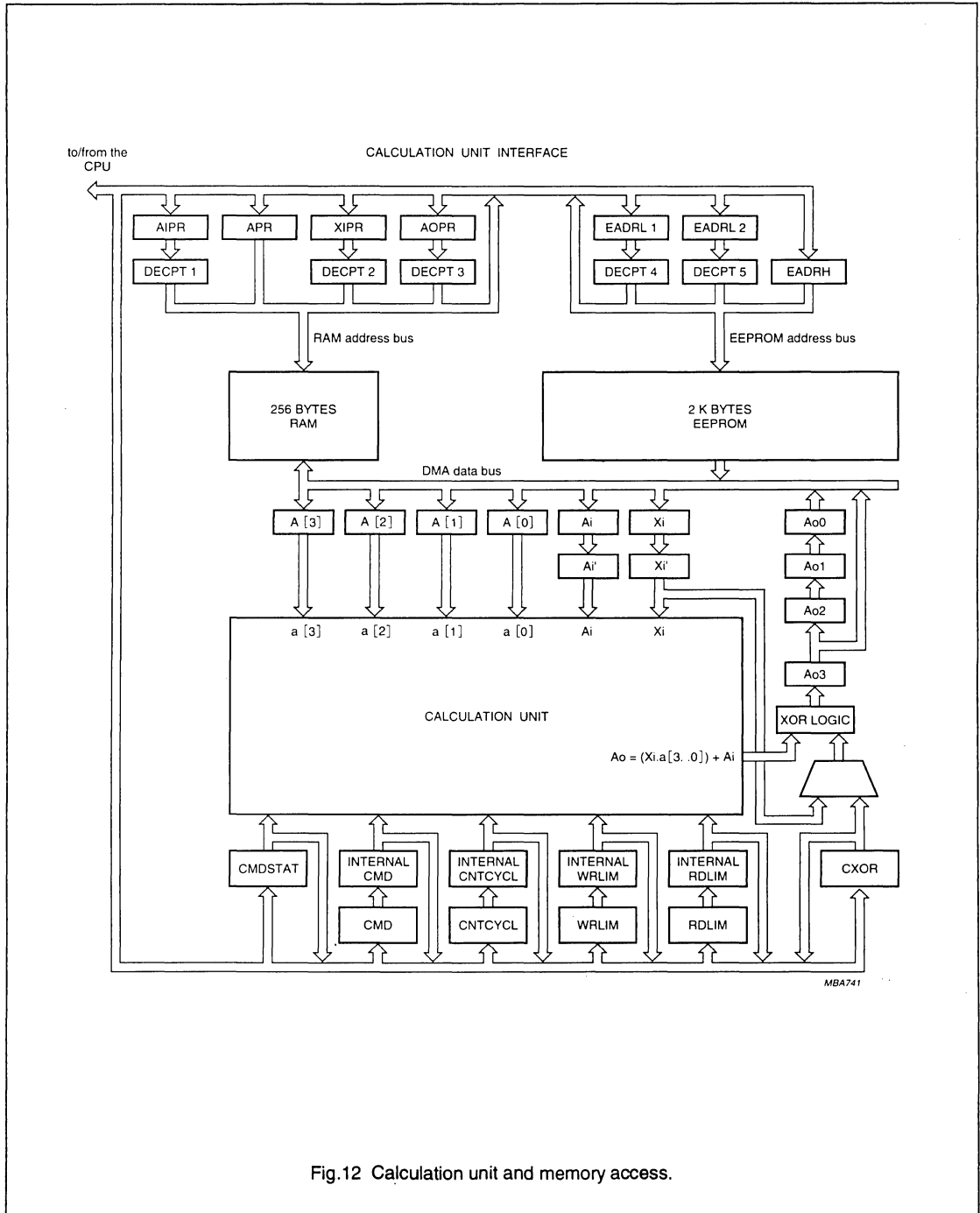


Fig.12 Calculation unit and memory access.

Secured 8-bit microcontroller

83C852

Command and status registers CMD and CMDSTAT

The calculation unit has two 8 bit registers (SFRs) for commands and status. SFR CMD is used for commands only, SFR CMDSTAT is used for both commands and status (2 command bits and 3 status bits).

7 bits of the CMD register are pipelined (see Table 8). At the beginning of a calculation, the contents of these seven bits are transferred into an internal command register that controls the calculation unit's sequencer.

This allows the CPU to re-initialize the CMD register for the next calculation while the current calculation is still busy.

The CMDSTAT SFR (see Table 10) is not pipelined and may not be reloaded by the CPU whilst the calculation unit is active. Both CMD and CMDSTAT registers are cleared at reset. Tables 9 and 11 describe the function of single status and control bits within the CMD and CMDSTAT registers.

Table 8 CMD SFR

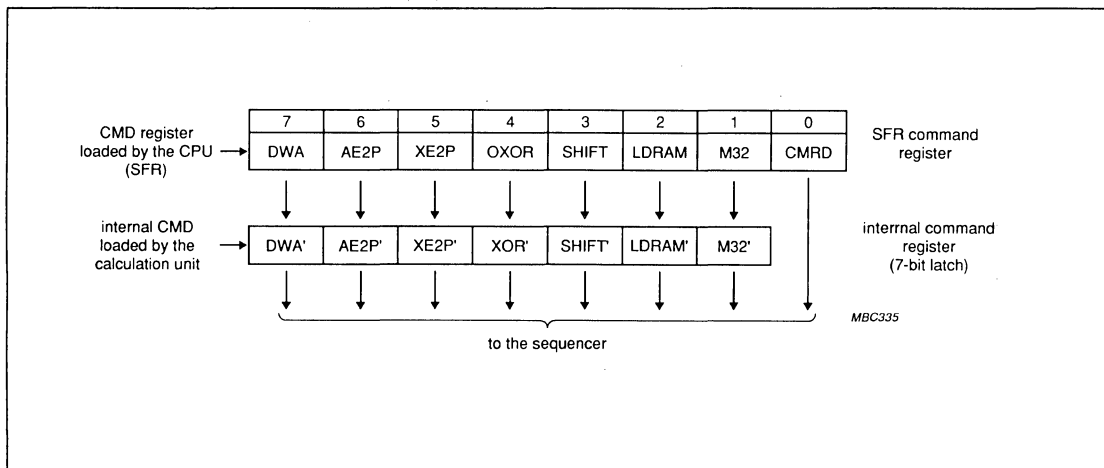


Table 9 Description of the CMD bits

| SYMBOL / PARAMETER | FUNCTION |
|---|---|
| CMD.7 | |
| DWA = disable output write | set and reset by the CPU. When the DWA bit is set 1, the writing operation of the Ao result into RAM is disabled. The AOPR address counter is not decremented. When DWA bit is reset 0, the Ao result bytes are written into RAM. |
| CMD.6 | |
| AE2P = input Ai operand from the EEPROM | set and reset by the CPU. When the AE2P bit is set 1 the EEPROM is addressed by AIPE (EADRL1) address pointer and the Ai operand is read from the EEPROM. When AE2P is reset 0 the RAM is addressed by AIPR address counter and the Ai operand is read from the RAM. |

Secured 8-bit microcontroller

83C852

| SYMBOL / PARAMETER | FUNCTION |
|--|--|
| CMD.5 | |
| XE2P = input Xi operand from the EEPROM | set and reset by the CPU. When the XE2P bit is set 1 the EEPROM is addressed by XIPE (EADRL2) address pointer and the Xi operand is read from the EEPROM. When XE2P is reset 0 the RAM is addressed by XIPR address counter and the Xi operand is read from the RAM. |
| CMD.4 | |
| OXOR = output exclusive OR | set and reset by the CPU. When the OXOR bit is set 1 the output data value is: $Xi \oplus$ calculation result. When the OXOR bit is reset 0 the output data value is: CXOR register content \oplus calculation result. If the output value must be the real calculation result: OXOR bit is reset 0 and the CXOR register content = 00H. |
| CMD.3 | |
| SHIFT = control of Ao result shift | set and reset by the CPU. When the SHIFT bit is set 1 the Ao result consists of four registers Ao3, Ao2, Ao1, Ao0, thus the output data is delayed four times which leads to a 32-bit result shift. If the SHIFT bit is reset 0, the output pipeline has only one register Ao3. The result is not shifted. |
| CMD.2 | |
| LDRAM = Load A3, A2, A1, A0 registers from the RAM | set and reset by the CPU. When the LDRAM bit is set 1 the calculation unit reloads A3, A2, A1, A0 registers from the RAM at the start of the computation. When the LDRAM bit is reset 0, the A3, A2, A1, A0 registers are reloaded from the Ao3, Ao2, Ao1, Ao0 output pipeline registers. |
| CMD.1 | |
| M32 = 32-bit operands | set and reset by the CPU. When the M32 bit is set 1 the multiplier operands are A[3..0] (32-bits), APR is incremented by four. When the M32 bit is reset 0 the multiplier operands are A[2-0] (28-bits), APR is incremented by three. |
| CMD.0 | |
| CMRD = command ready | set by the CPU and cleared by the calculation unit (see Fig. 13). To start a calculation, the CMRD bit is set 1 by the CPU. The CMRD bit will be reset 0 by hardware immediately after beginning the calculation. The CMD register is then ready to take the next command word from the CPU. When the CMRD bit is set 1 by the CPU while the calculation unit is still active, the calculation unit does not stop at the end of the current computation. The new calculation starts immediately with the new command parameters. In this case there will be no interrupt request. |

Secured 8-bit microcontroller

83C852

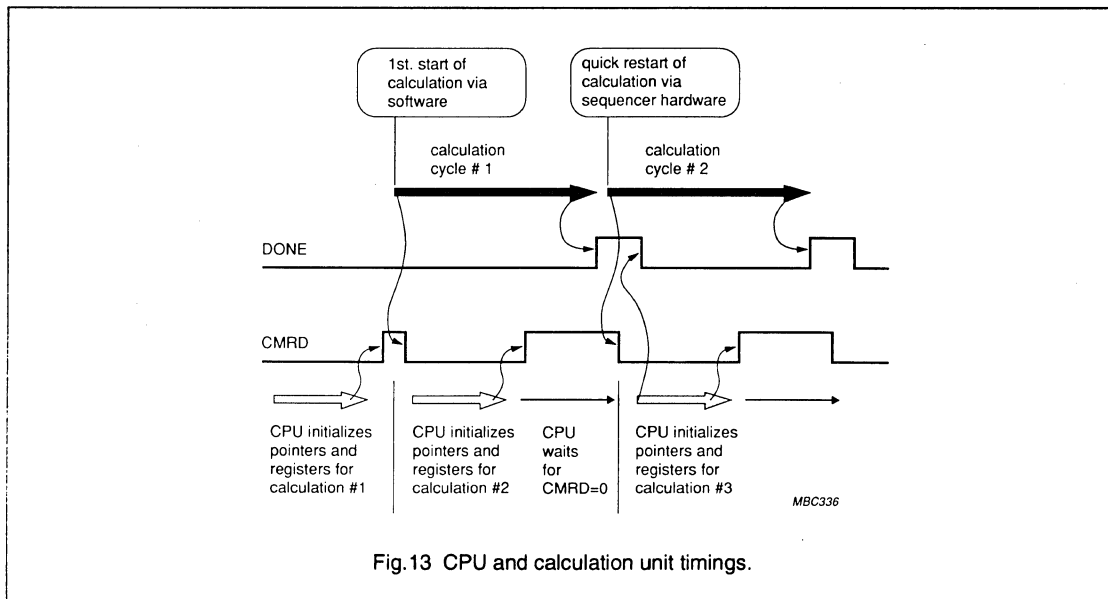


Fig.13 CPU and calculation unit timings.

Table 10 CMDSTAT SFR

| | | | | | |
|---|------------------|-----|---------------------------|-----|------|
| SFR Command and Status register (SFR) loaded and read by the CPU → | 4 | 3 | 2 | 1 | 0 |
| | DRA | DRX | RUN | CCY | DONE |
| | ↓ | ↓ | ↑ | ↑ | ↑ |
| | to the sequencer | | from the calculation unit | | |

Table 11 Description of the CMDSTAT bits

| SYMBOL / PARAMETER | FUNCTION |
|--------------------------------------|---|
| CMDSTAT bits 7, 6, 5, not applicable | |
| CMDSTAT.4 | |
| DRA = READ disable of the operand Ai | set and reset by the CPU. When the DRA bit is set 1, READ operation of the operand Ai is disabled and Ai is cleared. In this mode the AIPR address counter is not decremented. |
| CMDSTAT.3 | |
| DRX = READ disable of the operand Xi | set and reset by the CPU. When the DRX bit is set 1, READ operation of the operand Xi is disabled and Xi is cleared. In this mode the XIPR address counter is not decremented. |
| CMDSTAT.2 | |
| RUN = active calculation unit | set and reset by the calculation unit. While the calculation unit is active, the RUN bit is set 1, otherwise the RUN bit is reset 0 (cleared). |

Secured 8-bit microcontroller

83C852

| SYMBOL / PARAMETER | FUNCTION |
|--------------------------------------|---|
| CMDSTAT bits 7, 6, 5, not applicable | |
| CMDSTAT.1 | |
| CCY = carry | set and reset by the calculation unit. When the contents of Ao3, Ao2, Ao1, Ao0 output pipeline registers are greater than 0 (zero), the CCY bit is set 1, otherwise CCY is reset 0. |
| CMDSTAT.0 | |
| DONE = end of the computation | set by the calculation unit, reset by the CPU. The DONE bit is set 1 by the calculation unit at the end of computation. When set 1, an interrupt request is generated. The DONE flag has to be reset 0 by the CPU during an interrupt service routine. |

Cycle counter CNTCYCL

This is an 8-bit register with associated down counter. It counts the number of bytes of the result which the calculation unit shall carry out. The first byte of the result is always zero.

The CPU has WRITE access to the register and READ access to the down counter. CNTCYCL is cleared to 00H during reset. As long as the calculation unit is in standby, the down counter is continuously loaded with the register contents. After start of calculation, the down counter becomes separated from the register and starts counting. The register can then be reloaded by the CPU for the next calculation cycle. As soon as the down counter reaches the state 00H, the calculation cycle is terminated.

Limit registers

The calculation unit has two 8-bit limit registers (SFRs) WRLIM and RDLIM. WRLIM controls the start of Ao result output to RAM, while RDLIM controls the length of Ai and Xi input operands. RDLIM is split into two 4-bit registers. RDLIM (7-4) carry the READ limit for the Xi operand and RDLIM (3-0) the READ limit for Ai.

The contents of WRLIM, RDLIM (7-4), RDLIM (3-0) are compared to the contents of the Cycle Counter CNTCYCL during calculation. As Xi and Ai limits are both only 4-bits wide, these values are expanded to 8-bits for comparison by adding four leading zeros.

WRITE LIMIT REGISTER WRLIM:

- as long as the CNTCYCL contents are greater than the WRLIM's contents, writing of the Ao output result to RAM is inhibited.

READ LIMIT REGISTER RDLIM:

- READ Xi limit (upper 4-bits): when the CNTCYCL contents has reached the Xi limit, the READ value of the Xi operand is 0.
- READ Ai limit (lower 4-bits): when CNTCYCL contents has reached the Ai limit, the READ value of the Ai operand is 0.

The limit registers can be read and written by the CPU. Because of their pipeline structure, they can be re-loaded whilst the calculation unit is active. Both registers are cleared to 00H during reset.

CXOR register

This is an 8-bit wide SFR that provides one operand for an exclusive-OR operation on the calculation result. It can be read and written by the CPU, the default value after reset is 0.

Calculation unit interrupt

At the end of a calculation cycle, an interrupt request is generated (DONE = 1). The DONE flag has to be reset by software during the interrupt service routine. The calculation unit's operation cannot be interrupted by any other interrupt. Interrupt requests are pending until the end of the calculation cycle. They will be acknowledged during the interrupt service routine.

Parallel operation of CPU and calculation unit

The performance of the calculation unit degrades if pointer and control register initializations are done in between two consecutive calculation cycles. The full calculation speed is reached when these initializations are carried out by the CPU in parallel to a computation

Secured 8-bit microcontroller

83C852

cycle of the calculation unit (see Fig. 13). Thus with parallel initializations, when a current computation cycle is completed, all of the required initializations have already been done to enable the next computation cycle of the calculation unit to start immediately.

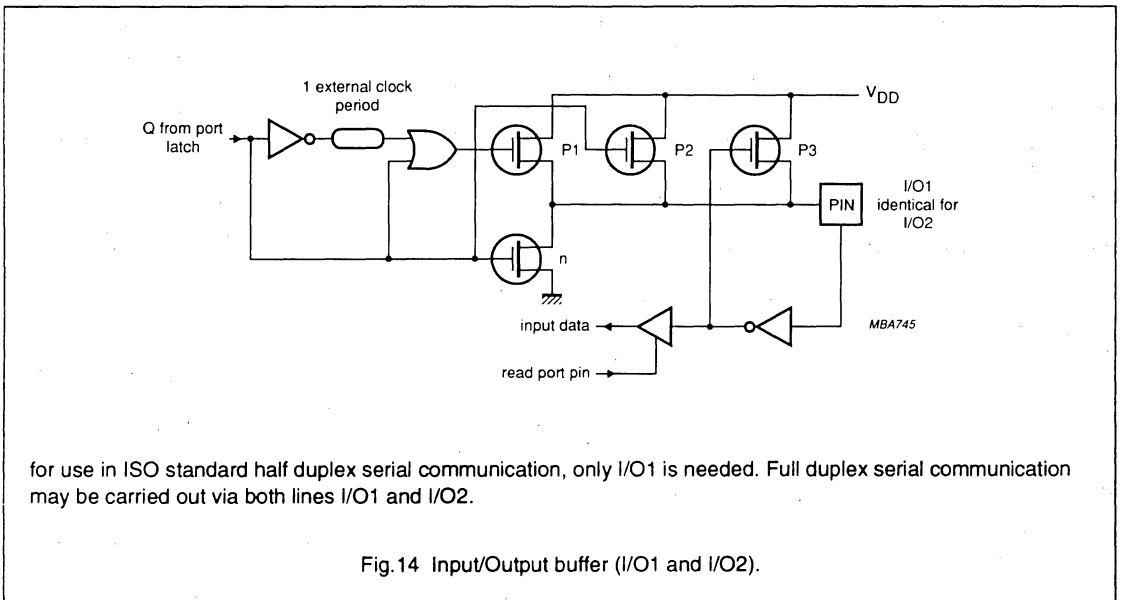
IO register

The 83C852 has 2 I/O lines: I/O1 and I/O2. Line I/O1 is represented by IO1 (bit 0) and line I/O2 is represented by IO2 (bit 1) of the IO register (SFR). IO bits: 7, 6, 5, 4, 3 and 2 are don't care.

Table 12 I/O SFR

| | | | | | | | |
|---|---|---|---|---|---|-----|-----|
| - | - | - | - | - | - | IO2 | IO1 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Either I/O line can be used independently from the other as an input or as an output. For an I/O line to be used as an input, a set 1 must first be written to its port-latch. See Fig. 14. The strong output driver FET P1 is turned off after one external clock period. The pin is then pulled HIGH by the weak pull-up FETs P2 and P3. It can be pulled LOW by an external source. After a hardware reset, both port latches contain a set 1 and both lines I/O1 and I/O2 are in Input mode.



Secured 8-bit microcontroller

83C852

Timers

The 83C852 has two 16-bit timer registers Timer 0 and Timer 1. The timer registers are incremented each machine cycle and are thus capable of counting machine cycles. Since a machine cycle consists of 6 external clock periods, the count rate is 1/6 of the clock frequency. Each timer has three operating modes:

- Mode 0 = 13-bit timer
- Mode 1 = 16-bit timer
- Mode 2 = 8-bit timer with auto-reload.

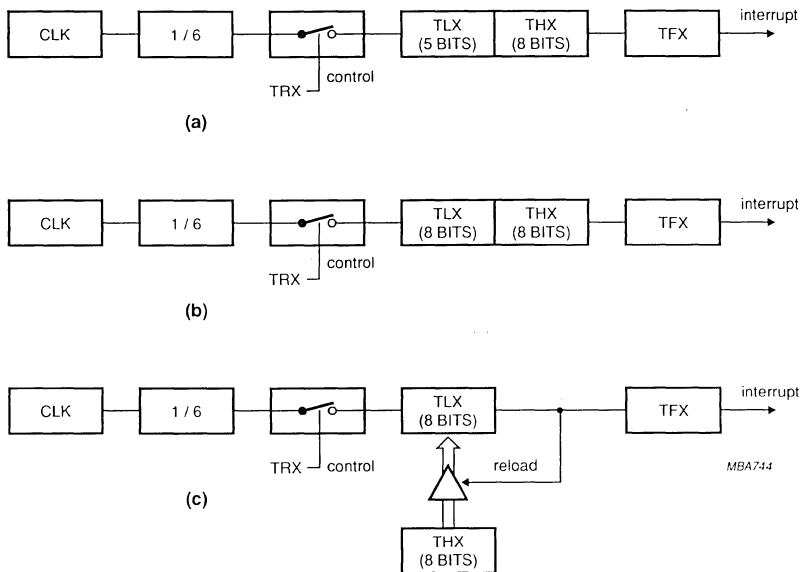


Fig.15 83C852 Timers

Secured 8-bit microcontroller

83C852

Timers 0 and 1 are controlled via the two SFRs: Timer Mode Control (TMOD) and Timer Control/External Interrupt Control (TCON).

Table 13 TMOD SFR

| TIMER 1 | | | | TIMER 2 | | | |
|---------|---|----|----|---------|---|----|----|
| – | – | M1 | M0 | – | – | M1 | M0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Table 14 Description of the TMOD bits

| SYMBOL | PARAMETER | FUNCTION |
|----------------------------------|-----------|--|
| Timer 1 | | |
| – | TMOD.7 | reserved, don't care |
| – | TMOD.6 | reserved, don't care |
| M1 | TMOD.5 | Timer 1 mode select |
| M0 | TMOD.4 | Timer 1 mode select |
| Timer 0 | | |
| – | TMOD.3 | reserved, don't care |
| – | TMOD.2 | reserved, don't care |
| M1 | TMOD.1 | Timer 0 mode select |
| M0 | TMOD.0 | Timer 0 mode select |
| M1 and M0 operating modes | | |
| 0 | 0 | 8-bit timer "THx" with "TLx" as 5-bit prescaler |
| 0 | 1 | 16-bit timer "THx" and "TLx" are cascaded. There is no prescaler |
| 1 | 0 | 8-bit auto-reload timer "THx" holds a value which is reloaded into "TLx" each time it overflows. |

Secured 8-bit microcontroller

83C852

Table 15 TCON SFR

| TIMER CONTROL | | | | EXTERNAL INTERRUPT CONTROL | | | |
|---------------|-----|-----|-----|----------------------------|------|-----|-----|
| TF1 | TR1 | TF0 | TR0 | – | IOSW | IE0 | IT0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Table 16 Description of the TCON bits

| SYMBOL | PARAMETER | FUNCTION |
|-----------------------------------|-----------|---|
| Timer control | | |
| TF1 | TCON.7 | Timer 1 overflow flag. Set by hardware on Timer 1 overflow. Cleared by hardware when processor vectors to interrupt routine. |
| TR1 | TCON.6 | Timer 1 run control bit. Set/cleared by software to turn Timer 1 ON/OFF. |
| TF0 | TCON.5 | Timer 0 overflow flag. Set by hardware on Timer 0 overflow. Cleared by hardware when processor vectors to interrupt routine. |
| TR0 | TCON.4 | Timer 0 run control bit. Set/cleared by software to turn Timer 0 ON/OFF. |
| External Interrupt Control | | |
| – | TCON.3 | reserved, don't care. |
| IOSW | TCON.2 | switch for external interrupt source: 0 = I/O1 is used as external interrupt source; 1 = I/O2 is used as external interrupt source. |
| IE0 | TCON.1 | is the external interrupt 0 edge flag. If IT0 is set 1, the IE0 bit is set 1 by hardware when the external interrupt source (either I/O1 pin or I/O2 pin) is detected to have made a 1 to 0 transition. The IE0 bit is cleared by hardware when the processor transfers control to the interrupt service routine. |
| IT0 | TCON.0 | determines whether external interrupt is edge-triggered or level-triggered. If IT0 is set 1: external interrupt 0 is edge-triggered. If IT0 is reset 0, external interrupt 0 is triggered by a detected LOW at the external interrupt source. |

Secured 8-bit microcontroller

83C852

Interrupt system

The 83C852 has five interrupt sources, each can be programmed to one of two priority interrupt levels, either HIGH or LOW. The five interrupt sources are listed below:

1. I/O: external request from either I/O line
2. Timer 0: overflow from Timer 0
3. Cell: end of calculation
4. Timer 1: overflow from Timer 1
5. EEPROM: completion of EEPROM programming.

Each interrupt source can be individually enabled or disabled, all interrupt sources can also be globally enabled or disabled.

Each interrupt source can be programmed to either a HIGH or a LOW priority interrupt level. A LOW can be interrupted by a HIGH priority interrupt, but not by another LOW priority interrupt. A HIGH priority interrupt cannot be interrupted.

Only one of the I/O lines (either I/O1 or I/O2) can be used as an external interrupt source at a time. This selection is made by IOSW bit from TCON register.

Interrupt vectors

The microcontroller acknowledges a request from an interrupt by a hardware subroutine call. It pushes the contents of the PC (program counter) into the stack, but it does not save the PSW (program status word). PC is reloaded with an address that depends on the source of the interrupt request, as shown below:

| Address | Source |
|---------|-----------------------------------|
| 0003H | I/O1 or I/O2 |
| 000BH | Timer 0 Overflow |
| 0013H | end of calculation |
| 001BH | Timer 1 Overflow |
| 0023H | completion of EEPROM programming. |

Interrupt registers IE and IP**INTERRUPT ENABLE REGISTER IE**

Each source can be individually enabled or disabled by setting or clearing the corresponding bit inside the SFR Interrupt Enable register IE. All interrupt sources can also be globally enabled or disabled.

Table 17 IE SFR

| EA | — | — | EE | ET1 | EC | ET0 | EX0 |
|----|---|---|----|-----|----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Table 18 Description of the IE bits

| SYMBOL | PARAMETER | FUNCTION |
|--------|-----------|--|
| EA | IE.7 | general enable/disable control 0 = no interrupt is enabled 1 = any individually enabled interrupt will be accepted |
| — | IE.6 | reserved, don't care |
| — | IE.5 | reserved, don't care |
| EE | IE.4 | enable EEPROM interrupt |
| ET1 | IE.3 | enable Timer 1 interrupt |
| EC | IE.2 | enable calculation unit interrupt |
| ET0 | IE.1 | enable Timer 0 interrupt |
| EX0 | IE.0 | enable external 0 interrupt (from I/O) |

Secured 8-bit microcontroller

83C852

Table 19 IP SFR

| | | | | | | | |
|---|---|---|----|-----|-----|-----|-----|
| - | - | - | PE | PT1 | PCU | PT0 | PX0 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Table 20 Description of the IP bits

| SYMBOL | PARAMETER | FUNCTION |
|--------|-----------|----------------------------------|
| - | IP.7 | reserved, don't care |
| - | IP.6 | reserved, don't care |
| - | IP.5 | reserved, don't care |
| PE | IP.4 | EEPROM interrupt level |
| PT1 | IP.3 | Timer 1 interrupt level |
| PCU | IP.2 | calculation unit interrupt level |
| PT0 | IP.1 | Timer 0 interrupt level |
| PX0 | IP.0 | external 0 interrupt level |

INTERRUPT PRIORITY REGISTER IP

The interrupt level is selected within the SFR Interrupt Priority register IP. Setting a bit to '1' selects HIGH priority.

Hardware security

OPERATING MODE

The microcontroller has two operating modes:

- User mode
- Test mode

The test mode is permanently disabled once the test has been performed.

Low frequency sensor

The low frequency detector circuit triggers a reset of the CPU when the clock frequency falls below f_{CLK} minimum (approximately 500 kHz). When the clock frequency rises above f_{CLK} minimum the reset is de-activated.

Power ON/OFF reset

The power ON/OFF reset circuit triggers a reset of the CPU when the power supply falls below V_{DD} minimum (approximately 3.5 V). When the power supply rises above V_{DD} minimum, this reset is de-activated. When the power-down mode is active (PD is set 1) the power ON/OFF reset is de-activated.

Idle mode and power-down mode

IDLE MODE

The 83C852 provides two power saving operational modes, the idle mode and the power-down mode. In the idle mode, the CPU enters a sleep routine whilst some of the on-chip peripherals (timers and interrupt system) remain active. The contents of the RAM and SFRs remain unchanged during the duration of an idle mode. The idle mode can be terminated by an enabled interrupt or by a hardware reset. Besides stopping the CPU, the idle mode terminates EEPROM write operations and stops operation of the calculation unit.

POWER-DOWN MODE

In the power-down mode, all on-chip internal clocks are frozen. The CPU and all on-chip peripherals stop working. The only exit from a power-down mode is by a hardware reset. The on-chip RAM and SFRs retain their values until the power-down mode is terminated. Reset redefines the SFRs, but does not change the RAM contents. The V_{DD} supply can be reduced to 2 V whilst the power-down mode is active. Both modes are activated by software via the SFR Power Control register PCON. PCON is not bit addressable.

Secured 8-bit microcontroller

83C852

Table 21 PCON SFR

| | | | | | | | |
|---|---|---|---|-----|-----|----|-----|
| – | – | – | – | GF1 | GF0 | PD | IDL |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Table 22 Description of the PCON bits

| SYMBOL | PARAMETER | FUNCTION |
|--------|-----------|--|
| – | PCON.7 | reserved, don't care |
| – | PCON.6 | reserved, don't care |
| – | PCON.5 | reserved, don't care |
| – | PCON.4 | reserved, don't care |
| GF1 | PCON.3 | general purpose flag bit |
| GF0 | PCON.2 | general purpose flag bit |
| PD | PCON.1 | enter power-down mode when set; note 1 |
| IDL | PCON.0 | enter idle mode when set; note 1 |

Note

1. If a logic 1 is written to PD and IDL at the same time, PD takes precedence.

Secured 8-bit microcontroller

83C852

SFRs memory mapping

Table 23

The 83C852 has the following 33 Special Function Registers (SFRs) available to the user.

| SFRs ADDRESS | SYMBOL | RESET VALUE | FUNCTION |
|--------------|---------|-------------|--|
| FBH | WRLIM | 0000.0000B | WRITE limit register for calculation unit |
| FAH | RDLIM | 0000.0000B | READ limit for calculation unit |
| F9H | CNTCYCL | 0000.0000B | cycle counter for calculation unit |
| F7H | ECNTRL2 | XXXX.0000B | EEPROM control register (test modes) |
| F6H | ECNTRL1 | 0000.0000B | EEPROM control register (user modes) |
| F5H | ETIM | 0000.1000B | EEPROM timer register |
| F4H | EDAT | XXXX.XXXXB | EEPROM data register |
| F3H | EADRH | 1000.0000B | EEPROM address register HIGH |
| F2H | EADRL1 | 0000.0000B | EEPROM address register 1 LOW, address pointer AIPE for calculation unit |
| F1H | EADRL2 | 0000.0000B | EEPROM address register 2 LOW, address pointer XIPE for calculation unit |
| F0H | B | 0000.0000B | B register |
| E0H | ACC | 0000.0000B | accumulator |
| D0H | PSW | 0000.0000B | program status word |
| B8H | IP | XXX0.0000B | interrupt priority register |
| B0H | IO | XXXX.XX11B | I/O register |
| A8H | IE | 0XX0.0000B | interrupt enable register |
| A7H | AOPR | 0000.0000B | AOPR register for calculation unit |
| A6H | APR | 0000.0000B | APR register for calculation unit |
| A5H | XIPR | 0000.0000B | XIPR register for calculation unit |
| A4H | AIPR | 0000.0000B | AIPR register for calculation unit |
| A3H | CXOR | 0000.0000B | CXOR register for calculation unit |
| 99H | CMD | 0000.0000B | command register for calculation unit |
| 98H | CMDSTAT | XXX0.0000B | command and status register for calculation unit |
| 8DH | TH1 | 0000.0000B | Timer 1 HIGH |
| 8CH | TH0 | 0000.0000B | Timer 0 HIGH |
| 8BH | TL1 | 0000.0000B | Timer 1 LOW |
| 8AH | TL0 | 0000.0000B | Timer 0 LOW |
| 89H | TMOD | XX00.XX00B | Timer 0 and 1 mode control |
| 88H | TCON | 0000.X000B | Timer 0 and 1 control and external interrupt control |
| 87H | PCON | XXXX.0000B | power control register |
| 83H | DPH | 0000.0000B | data pointer HIGH |
| 82H | DPL | 0000.0000B | data pointer LOW |
| 81H | SP | 0000.0111B | stack pointer |

Secured 8-bit microcontroller

83C852

| | | 8 Bytes | | | | | | | |
|----|---------|-------------------|--------|-------|------|------|---------|---------|----|
| | | ↙ Bit addressable | | | | | | | |
| F8 | | CNTCYCL | RDLIM | WRLIM | | | | | FF |
| F0 | B | EADRL2 | EADRL1 | EADRH | EDAT | ETIM | ECNTRL1 | ECNTRL2 | F7 |
| E8 | | | | | | | | | EF |
| E0 | ACC | | | | | | | | E7 |
| D8 | | | | | | | | | DF |
| D0 | PSW | | | | | | | | D7 |
| C8 | | | | | | | | | CF |
| C0 | | | | | | | | | C7 |
| B8 | IP | | | | | | | | BF |
| B0 | IO | | | | | | | | B7 |
| A8 | IE | | | | | | | | AF |
| A0 | | | | CXOR | AIPR | XIPR | APR | AOPR | A7 |
| 98 | CMDSTAT | CMD | | | | | | | 9F |
| 90 | | | | | | | | | 97 |
| 88 | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | | | 8F |
| 80 | | SP | DPL | DPH | | | | PCON | 87 |

MBA746


 83C852 specific SFR's

Fig.16 SFRs mapping.

Secured 8-bit microcontroller

83C852

INSTRUCTION SET

The instruction set consists of 49 single-byte, 46 two-byte and 16 three-byte instructions. When using a 6 MHz external clock, 64 instructions execute in 1 cycle (1 μ s) and 45 instructions execute in 2 cycles (2 μ s). Multiply and divide instructions execute in 4 cycles (4 μ s).

INSTRUCTION SET DESCRIPTION

For data addressing modes, Hexadecimal opcode cross-reference and invalid instructions, see Table notes.

| MNEMONIC | DESCRIPTION | BYTES | CYCLES | OPCODE (HEX) |
|-----------------------------|--|-------|--------|--------------|
| Arithmetic operation | | | | |
| ADD A,Rr | Add register to A | 1 | 1 | 2* |
| ADD A,direct | Add direct byte to A | 2 | 1 | 25 |
| ADD A,@Ri | Add indirect RAM to A | 1 | 1 | 26, 27 |
| ADD A,#data | Add immediate data to A | 2 | 1 | 24 |
| ADDC A,Rr | Add register to A with carry flag | 1 | 1 | 3* |
| ADDC A,direct | Add direct byte to A with carry flag | 2 | 1 | 35 |
| ADDC A,@Ri | Add indirect RAM to A with carry flag | 1 | 1 | 36, 37 |
| ADDC A,#data | Add immediate data to A with carry flag | 2 | 1 | 34 |
| SUBB A,Rr | Subtract register from A with borrow | 1 | 1 | 9* |
| SUBB A,direct | Subtract direct byte from A with borrow | 2 | 1 | 95 |
| SUBB A,@Ri | Subtract indirect RAM from A with borrow | 1 | 1 | 96, 97 |
| SUBB A,#data | Subtract immediate data from A with borrow | 2 | 1 | 94 |
| INC A | Increment A | 1 | 1 | 04 |
| INC Rr | Increment register | 1 | 1 | 0* |
| INC direct | Increment direct byte | 2 | 1 | 05 |
| INC @Ri | Increment indirect RAM | 1 | 1 | 06, 07 |
| DEC A | Decrement A | 1 | 1 | 14 |
| DEC Rr | Decrement register | 1 | 1 | 1* |
| DEC direct | Decrement direct byte | 2 | 1 | 15 |
| DEC @Ri | Decrement indirect RAM | 1 | 1 | 16, 17 |
| INC DPTR | Increment data pointer | 1 | 2 | A3 |
| MUL AB | Multiply A & B | 1 | 4 | A4 |
| DIV AB | Divide A by B | 1 | 4 | 84 |
| DA A | Decimal adjust A | 1 | 1 | D4 |

Secured 8-bit microcontroller

83C852

Instruction set description (continued)

| MNEMONIC | DESCRIPTION | BYTES | CYCLES | OPCODE (HEX) |
|-------------------------|--|-------|--------|--------------|
| Logic operations | | | | |
| ANL A,Rr | AND register to A | 1 | 1 | 5* |
| ANL A,direct | AND direct byte to A | 2 | 1 | 55 |
| ANL A,@Ri | AND indirect RAM to A | 1 | 1 | 56, 57 |
| ANL A,#data | AND immediate data to A | 2 | 1 | 54 |
| ANL direct,A | AND A to direct byte | 2 | 1 | 52 |
| ANL direct,#data | AND immediate data to direct byte | 3 | 2 | 53 |
| ORL A,Rr | OR register to A | 1 | 1 | 4* |
| ORL A,direct | OR direct byte to A | 2 | 1 | 45 |
| ORL A,@Ri | OR indirect RAM to A | 1 | 1 | 46, 47 |
| ORL A,#data | OR immediate data to A | 2 | 1 | 44 |
| ORL direct,A | OR A to direct byte | 2 | 1 | 42 |
| ORL direct,#data | OR immediate data to direct byte | 3 | 2 | 43 |
| XRL A,Rr | Exclusive-OR register to A | 1 | 1 | 6* |
| XRL A,direct | Exclusive-OR direct byte to A | 2 | 1 | 65 |
| XRL A,@Ri | Exclusive-OR indirect RAM to A | 1 | 1 | 66, 67 |
| XRL A,#data | Exclusive-OR immediate data to A | 2 | 1 | 64 |
| XRL direct,A | Exclusive-OR A to direct byte | 2 | 1 | 62 |
| XRL direct,#data | Exclusive-OR immediate data to direct byte | 3 | 2 | 63 |
| CLR A | Clear A | 1 | 1 | E4 |
| CPL A | Complement A | 1 | 1 | F4 |
| RL A | Rotate A left | 1 | 1 | 23 |
| RLC A | Rotate A left through the carry flag | 1 | 1 | 33 |
| RR A | Rotate A right | 1 | 1 | 03 |
| RRC A | Rotate A right through the carry flag | 1 | 1 | 13 |
| SWAP A | Swap nibbles within A | 1 | 1 | C4 |

Secured 8-bit microcontroller

83C852

Instruction set description (continued)

| MNEMONIC | DESCRIPTION | BYTES | CYCLES | OPCODE (HEX) |
|--|--|-------|--------|--------------|
| Data transfer | | | | |
| MOV A,Rr | Move register to A | 1 | 1 | E* |
| MOV A,direct** | Move direct byte to A | 2 | 1 | E5 |
| MOV A,@Ri | Move indirect RAM to A | 1 | 1 | E6, E7 |
| MOV A,#data | Move immediate data to A | 2 | 1 | 74 |
| MOV Rr,A | Move A to register | 1 | 1 | F* |
| MOV Rr,direct | Move direct byte to register | 2 | 2 | A* |
| MOV Rr,#data | Move immediate data to register | 2 | 1 | 7* |
| MOV direct,A | Move A to direct byte | 2 | 1 | F5 |
| MOV direct,Rr | Move register to direct byte | 2 | 2 | 8* |
| MOV direct,direct | Move direct byte to direct | 3 | 2 | 85 |
| MOV direct,@Ri | Move indirect RAM to direct byte | 2 | 2 | 86, 87 |
| MOV direct,#data | Move immediate data to direct byte | 3 | 2 | 75 |
| MOV @Ri,A | Move A to indirect RAM | 1 | 1 | F6, F7 |
| MOV @Ri,direct | Move direct byte to indirect RAM | 2 | 2 | A6, A7 |
| MOV @Ri,#data | Move immediate data to indirect RAM | 2 | 1 | 76, 77 |
| MOV DPTR,#data 16 | Load data pointer with a 16-bit constant | 3 | 2 | 90 |
| MOVC A,@A+DPTR | Move code byte relative to DPTR to A | 1 | 2 | 93 |
| MOVC A,@A+PC | Move code byte relative to PC to A | 1 | 2 | 83 |
| PUSH direct | Push direct byte onto stack | 2 | 2 | C0 |
| POP direct | Pop direct byte from stack | 2 | 2 | D0 |
| XCH A,Rr | Exchange register with A | 1 | 1 | C* |
| XCH A,direct | Exchange direct byte with A | 2 | 1 | C5 |
| XCH A,@Ri | Exchange indirect RAM with A | 1 | 1 | C6, C7 |
| XCHD A,@Ri | Exchange LOW-order digit indirect RAM with A | 1 | 1 | D6, D7 |
| Note: the following MOVX instructions of 80C51 set are not applicable to 83C852 | | | | |
| MOVX A,@Ri | Move external RAM (8-bit address) to A | 1 | 2 | E2, E3 |
| MOVX A,@DPTR | Move external RAM (16-bit address) to A | 1 | 2 | E0 |
| MOVX @Ri,A | Move A to external RAM (8-bit address) | 1 | 2 | F2, F3 |
| MOVX @DPTR,A | Move A to external RAM (16-bit address) | 1 | 2 | F0 |

Secured 8-bit microcontroller

83C852

Instruction set description (continued)

| MNEMONIC | DESCRIPTION | BYTES | CYCLES | OPCODE (HEX) |
|--------------------------------------|--|-------|--------|--------------|
| Boolean variable manipulation | | | | |
| CLR C | Clear carry flag | 1 | 1 | C3 |
| CLR bit | Clear direct bit | 2 | 1 | C2 |
| SETB C | Set carry flag | 1 | 1 | D3 |
| SETB bit | Set direct bit | 2 | 1 | D2 |
| CPL C | Complement carry flag | 1 | 1 | B3 |
| CPL bit | Complement direct bit | 2 | 1 | B2 |
| ANL C,bit | AND direct bit to carry flag | 2 | 2 | 82 |
| ANL C,/bit | AND complement of direct bit to carry flag | 2 | 2 | B0 |
| ORL C,bit | OR direct bit to carry flag | 2 | 2 | 72 |
| ORL C,/bit | OR complement of direct bit to carry flag | 2 | 2 | A0 |
| MOV C,bit | Move direct bit to carry flag | 2 | 1 | A2 |
| MOV bit,C | Move carry flag to direct bit | 2 | 2 | 92 |
| Program and machine control | | | | |
| ACALL addr11 | Absolute subroutine call | 2 | 2 | •1addr |
| LCALL addr16 | Long subroutine call | 3 | 2 | 12 |
| RET | Return from subroutine | 1 | 2 | 22 |
| RETI | Return from interrupt | 1 | 2 | 32 |
| AJMP addr11 | Absolute jump | 2 | 2 | ♦1addr |
| LJMP addr16 | Long jump | 3 | 2 | 02 |
| SJMP rel | Short jump (relative address) | 2 | 2 | 80 |
| JMP @A+DPTR | Jump indirect relative to the DPTR | 1 | 2 | 73 |
| JZ rel | Jump if A is zero | 2 | 2 | 60 |
| JNZ rel | Jump if A is not zero | 2 | 2 | 70 |
| JC rel | Jump if carry flag is set | 2 | 2 | 40 |
| JNC rel | Jump if carry flag is not set | 2 | 2 | 50 |
| JB bit,rel | Jump if direct bit is set | 3 | 2 | 20 |
| JNB bit,rel | Jump if direct bit is not set | 3 | 2 | 30 |
| JBC bit,rel | Jump if direct bit is set and clear bit | 3 | 2 | 10 |
| CJNE A,direct,rel | Compare direct to A and jump if not equal | 3 | 2 | B5 |
| CJNE A,#data,rel | Compare immediate to A and jump if not equal | 3 | 2 | B4 |
| CJNE Rr,#data,rel | Compare immed. to reg. and jump if not equal | 3 | 2 | B* |
| CJNE @Ri,#data,rel | Compare immed. to ind. and jump if not equal | 3 | 2 | B6, B7 |
| DJNZ Rr,rel | Decrement register and jump if not zero | 2 | 2 | D* |
| DJNZ direct,rel | Decrement direct and jump if not zero | 3 | 2 | D5 |
| NOP | No operation | 1 | 1 | 00 |

Secured 8-bit microcontroller

83C852

NOTES TO INSTRUCTION SET TABLE

| MNEMONIC | DESCRIPTION |
|---|---|
| Data addressing modes | |
| Rr | working register R0-R7. |
| direct | 128 internal RAM locations and any special function register (SFR). |
| @Ri | indirect internal RAM location addressed by register R0 or R1 of the actual register bank. |
| #data | 8-bit constant included in instruction. |
| #data 16 | 16-bit constant included as bytes 2 and 3 of instruction. |
| bit | direct addressed bit in internal RAM or SFR. |
| addr16 | 16-bit destination address. Used by LCALL and LJMP. The branch will be anywhere within the 64K byte program memory address space. |
| addr11 | 11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2K byte page of program memory as the first byte of the following instruction. |
| rel | Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is -128 to +127 bytes relative to first byte of the following instruction. |
| Hexadecimal opcode cross-reference | |
| * | 8, 9, A, B, C, D, E, F. |
| • | 11, 31, 51, 71, 91, B1, D1, F1. |
| ♦ | 01, 21, 41, 61, 81, A1, C1, E1. |
| Invalid instructions: | |
| note ** | MOV A, ACC is not a valid instruction. |
| MOVX | MOVX instructions of 80C51 set are not applicable to 83C852. |

INSTRUCTION MAP

MOVX instructions not applicable

| | first hexadecimal character of opcode | | | | second hexadecimal character of opcode | | | | | | | | | | | |
|---|---------------------------------------|-----------------|---------------------|-------------------|--|-------------------|-------------------------|---|------------------------------------|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 0 | NOP | AJMP addr11 | LJMP addr16 | RR A | INC A | INC dir | INC @Ri 0 | 1 | INC Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 1 | JBC bit,rel | ACALL addr11 | LCALL addr16 | RRC A | DEC A | DEC dir | DEC @Ri 0 | 1 | DEC Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 2 | JB bit,rel | AJMP addr11 | RET | RL A | ADD A,#data | ADD A,dir | ADD A,@Ri 0 | 1 | ADD A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 3 | JNB bit,rel | ACALL addr11 | RETI | RLC A | ADDC A,#data | ADDC A,dir | ADDC A,@Ri 0 | 1 | ADDC A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 4 | JC rel | AJMP addr11 | ORL dir,A | ORL dir,#data | ORL A,#data | ORL A,dir | ORL A,@Ri 0 | 1 | ORL A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 5 | JNC rel | ACALL addr11 | ANL dir,A | ANL dir,#data | ANL A,#data | ANL A,dir | ANL A,@Ri 0 | 1 | ANL A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 6 | JZ rel | AJMP addr11 | XRL dir,A | XRL dir,#data | XRL A,#data | XRL A,dir | XRL A,@Ri 0 | 1 | XRL A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 7 | JNZ rel | ACALL addr11 | ORL C,bit | JMP @A+DPTR | MOV A,#data | MOV dir,#data | MOV @Ri,#data 0 | 1 | MOV Rr,#data 0 1 2 3 | 4 | 5 | 6 | 7 | | | |
| 8 | SJMP rel | AJMP addr11 | ANL C,bit | MOVC A,@A+PC | DIV AB | MOV dir,dir | MOV dir,@Ri 0 | 1 | MOV dir,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| 9 | MOV DPTR, #data | ACALL addr11 | MOV bit,C | MOVC A,@A+DPTR | SUBB A,#data | SUBB A,dir | SUBB A,@Ri 0 | 1 | SUBB A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| A | ORL C,/bit | AJMP addr11 | MOV C,bit | INC DPTR | MUL AB | | MOV @Ri,dir 0 | 1 | MOV Rr,dir 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| B | ANL C,/bit | ACALL addr11 | CPL bit | CPL C | CJNE A, #data,rel | CJNE A,dir,rel | CJNE @Ri,#data,rel 0 | 1 | CJNE Rr,#data,rel 0 1 2 3 | 4 | 5 | 6 | 7 | | | |
| C | PUSH dir | AJMP addr11 | CLR bit | CLR C | SWAP A | XCH A,dir | XCH A,@Ri 0 | 1 | XCH A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| D | POP dir | ACALL addr11 | SETB bit | SETB C | DA A | DJNZ dir,rel | XCHD A,@Ri 0 | 1 | DJNZ Rr,rel 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| E | MOVX A,@DPTR | AJMP addr11 | MOVX A,@Ri 0 1 | | CLR A | MOV A,dir | MOV A,@Ri 0 | 1 | MOV A,Rr 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |
| F | MOVX @DPTR,A | ACALL addr11 | MOVX @Ri,A 0 1 | | CPL A | MOV dir,A | MOV @Ri,A 0 | 1 | MOV Rr,A 0 1 2 | 3 | 4 | 5 | 6 | 7 | | |

* MOV A,ACC is not a valid instruction.

Secured 8-bit microcontroller

83C852

ISO INFORMATION

The following ISO characteristics information as applicable to this data sheet may be superseded. Please ensure that the latest version of ISO information is studied for relevant features.

ISO ELECTRICAL CHARACTERISTICS

| SYMBOL | PARAMETER | CONDITIONS | MIN. | TYP. | MAX. | UNIT |
|--------------|------------------------------|---|---------------------|------|---|---------|
| I/O | | | | | | |
| V_{IH} | input voltage HIGH | $I_{IH (max.)} = \pm 20 \mu A$ | $0.7 \times V_{DD}$ | – | $V_{DD} + 0.3$ | V |
| V_{IL} | input voltage LOW | $I_{IL (max.)} = -1 \text{ mA}$ | -0.3 | – | 0.8 | V |
| V_{OH} | output voltage HIGH | $I_{OH (max.)} = -20 \mu A$; note 1 | 3.8 | – | V_{DD} | V |
| V_{OL} | output voltage LOW | $I_{OL (max.)} = +1 \text{ mA}$ | 0 | – | 0.4 | V |
| C_{VO} | input/output pin capacitance | | – | – | 30 | pF |
| $t_{r/f}$ | I/O rise/fall times | $C_{IN} = 30 \text{ pF}$; $C_{OUT} = 30 \text{ pF}$ | – | – | 1 | μs |
| CLK | | | | | | |
| V_{IH} | input voltage HIGH | $I_{IH (max.)} = \pm 20 \mu A$ | $0.7 \times V_{DD}$ | – | $V_{DD} + 0.3$ | V |
| V_{IL} | input voltage HIGH | $I_{IL (max.)} = \pm 200 \mu A$ | -0.3 | – | 0.5 | V |
| C_i | input pin capacitance | | – | – | 30 | pF |
| $t_{r/f}$ | CLK rise/fall times | $C_{IN} = 30 \text{ pF}$ | – | – | 9% of period with a max. of 0.5 μs | μs |
| RESET | | | | | | |
| V_{IH} | input voltage HIGH | $I_{IH (max.)} = \pm 20 \mu A$ | $0.7 \times V_{DD}$ | – | $V_{DD} + 0.3$ | V |
| V_{IL} | input voltage LOW | $I_{IL (max.)} = \pm 200 \mu A$ | -0.3 | – | 0.5 | V |

Note

1. It is assumed that a pull-up resistor is used in the interface device (recommend value = 20 k Ω).

LIMITING VALUES

In accordance with the Absolute Maximum System (IEC 134)

| SYMBOL | PARAMETER | MIN. | MAX. | UNIT |
|---------------|--|------|-----------|-------------|
| V_i | input voltage on any pin with respect to ground (V_{SS}) | -0.5 | ± 6.5 | V |
| I_i ; I_o | input/output current on I/O1 or I/O2 pin | – | ± 5 | mA |
| P_{tot} | total power dissipation per package | – | 1 | W |
| T_{stg} | storage temperature range | -65 | 150 | $^{\circ}C$ |
| T_{amb} | operating ambient temperature range | 0 | 70 | $^{\circ}C$ |

HANDLING

Inputs and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe, it is desirable to take precautions appropriate to handling MOS devices (see 'Handling MOS Devices').

Secured 8-bit microcontroller

83C852

CHARACTERISTICS

$V_{DD} = 5\text{ V}$ ($\pm 10\%$); $V_{SS} = 0\text{ V}$; $T_{amb} = 0\text{ to }70\text{ }^\circ\text{C}$; all voltages with respect to V_{SS} unless otherwise specified.

| SYMBOL | PARAMETER | CONDITIONS | MIN. | TYP. | MAX. | UNIT |
|-------------------|---|--|--------------------|------|--------------------|---------------|
| DC | | | | | | |
| V_{DD} | supply voltage range | | 4.5 | – | 5.5 | V |
| I_{DD} | supply current | $f_{CLK} = 3.57\text{ MHz}$ | – | – | 10 | mA |
| I_{DD} | supply current operating mode | $f_{CLK} = 6.0\text{ MHz}$ | – | – | 15 | mA |
| I_{ID} | supply current idle mode | $f_{CLK} = 6.0\text{ MHz}$ | – | – | 3 | mA |
| I_{PD} | power-down current | $2\text{ V} \leq V_{PD} \leq V_{DD}\text{ max.}$ | – | – | 100 | μA |
| I/O1; I/O2 | | | | | | |
| V_{IL} | input voltage LOW | | –0.5 | – | $0.2 V_{DD}$ | V |
| V_{IH} | input voltage HIGH | | $0.2 V_{DD} + 0.9$ | – | $V_{DD} + 0.5$ | V |
| I_{IL} | input current LOW | $V_I = +0.45\text{ V}$ | – | – | –50 | μA |
| I_{IL} | input current HIGH-to-LOW | | – | – | 650 | μA |
| V_{OH} | output voltage HIGH | $I_{OH} = -20\text{ }\mu\text{A}$ | 3.8 | – | – | V |
| V_{OL} | output voltage LOW | $I_{OL} = 1.0\text{ mA}$ | – | – | 0.4 | V |
| V_{IL} | RESET; CLK input voltage LOW | | –0.5 | – | $0.2 V_{DD} - 0.1$ | V |
| V_{IH} | RESET; CLK input voltage HIGH | | $0.7 V_{DD}$ | – | $V_{DD} + 0.5$ | V |
| I_{LI} | input leakage current (RESET; CLK) | $0.45 < V_I < V_{DD}$ | – | – | ± 10 | μA |
| ESD | ESD protection | $C = 100\text{ pF}$; $R = 1.5\text{ k}$ | – | – | 2.0 | kV |
| AC | | | | | | |
| f_{CLK} | external clock frequency | internal operating frequency = f_{CLK} | 1 | – | 6 | MHz |
| t_{CYC} | cycle time | | 1 | – | – | μs |
| t_{CLK} | clock pulse width | | 45 | – | 55 | % |
| t_r | clock rise time | | – | – | tbf | ns |
| t_f | clock fall time | | – | – | tbf | ns |
| t_{POR} | power-on reset delay | | tbf | – | tbf | tbf |
| t_{rw} | reset pulse width | | $12/f_{CLK}$ | – | – | s |
| t_e | EEPROM ERASE time | | – | 5.0 | – | ms |
| t_w | EEPROM WRITE time | | – | 5.0 | – | ms |
| t_s | EEPROM data retention time | $T_{amb} = 55^\circ\text{ C}$ | 10.0 | – | – | yrs |
| N_{ew} | EEPROM endurance (number of erase/write cycles) | $t_e = 5\text{ ms}$; $t_w = 5\text{ ms}$ | 10 000 | – | – | cycles |
| C | I/O1; I/O2; RESET; f_{CLK} pin capacitance | $f_{CLK} = 1\text{ MHz}$; $T_{amb} = 25^\circ\text{ C}$ | – | – | 10 | pF |