**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Obsolete |
| Core Processor | 80C51 |
| Core Size | 8-Bit |
| Speed | 60MHz |
| Connectivity | SPI, UART/USART |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 34 |
| Program Memory Size | 64KB (64K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | 2K x 8 |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 5.5V |
| Data Converters | - |
| Oscillator Type | External |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-LQFP |
| Supplier Device Package | 44-VQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/at89c51ed2-rltim |

**Table 7.** PCA SFRs (Continued)

| Mnemo-nic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| CCAP0H | FAh | PCA Compare Capture Module 0 H | CCAP0H7 | CCAP0H6 | CCAP0H5 | CCAP0H4 | CCAP0H3 | CCAP0H2 | CCAP0H1 | CCAP0H0 |
| CCAP1H | FBh | PCA Compare Capture Module 1 H | CCAP1H7 | CCAP1H6 | CCAP1H5 | CCAP1H4 | CCAP1H3 | CCAP1H2 | CCAP1H1 | CCAP1H0 |
| CCAP2H | FCh | PCA Compare Capture Module 2 H | CCAP2H7 | CCAP2H6 | CCAP2H5 | CCAP2H4 | CCAP2H3 | CCAP2H2 | CCAP2H1 | CCAP2H0 |
| CCAP3H | FDh | PCA Compare Capture Module 3 H | CCAP3H7 | CCAP3H6 | CCAP3H5 | CCAP3H4 | CCAP3H3 | CCAP3H2 | CCAP3H1 | CCAP3H0 |
| CCAP4H | FEh | PCA Compare Capture Module 4 H | CCAP4H7 | CCAP4H6 | CCAP4H5 | CCAP4H4 | CCAP4H3 | CCAP4H2 | CCAP4H1 | CCAP4H0 |
| CCAP0L | EAh | PCA Compare Capture Module 0 L | CCAP0L7 | CCAP0L6 | CCAP0L5 | CCAP0L4 | CCAP0L3 | CCAP0L2 | CCAP0L1 | CCAP0L0 |
| CCAP1L | EBh | PCA Compare Capture Module 1 L | CCAP1L7 | CCAP1L6 | CCAP1L5 | CCAP1L4 | CCAP1L3 | CCAP1L2 | CCAP1L1 | CCAP1L0 |
| CCAP2L | ECh | PCA Compare Capture Module 2 L | CCAP2L7 | CCAP2L6 | CCAP2L5 | CCAP2L4 | CCAP2L3 | CCAP2L2 | CCAP2L1 | CCAP2L0 |
| CCAP3L | EDh | PCA Compare Capture Module 3 L | CCAP3L7 | CCAP3L6 | CCAP3L5 | CCAP3L4 | CCAP3L3 | CCAP3L2 | CCAP3L1 | CCAP3L0 |
| CCAP4L | EEh | PCA Compare Capture Module 4 L | CCAP4L7 | CCAP4L6 | CCAP4L5 | CCAP4L4 | CCAP4L3 | CCAP4L2 | CCAP4L1 | CCAP4L0 |

**Table 8.** Serial I/O Port SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SCON | 98h | Serial Control | FE/SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
| SBUF | 99h | Serial Data Buffer | | | | | | | | |
| SADEN | B9h | Slave Address Mask | | | | | | | | |
| SADDR | A9h | Slave Address | | | | | | | | |
| BDRCON | 9Bh | Baud Rate Control | | | | BRR | TBCK | RBCK | SPD | SRC |
| BRL | 9Ah | Baud Rate Reload | | | | | | | | |

**Table 9.** SPI Controller SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SPCON | C3h | SPI Control | SPR2 | SPEN | SSDIS | MSTR | CPOL | CPHA | SPR1 | SPR0 |
| SPSTA | C4h | SPI Status | SPIF | WCOL | SSERR | MODF | | | | |
| SPDAT | C5h | SPI Data | SPD7 | SPD6 | SPD5 | SPD4 | SPD3 | SPD2 | SPD1 | SPD0 |

**Table 10.** Keyboard Interface SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| KBLS | 9Ch | Keyboard Level Selector | KBLS7 | KBLS6 | KBLS5 | KBLS4 | KBLS3 | KBLS2 | KBLS1 | KBLS0 |
| KBE | 9Dh | Keyboard Input Enable | KBE7 | KBE6 | KBE5 | KBE4 | KBE3 | KBE2 | KBE1 | KBE0 |
| KBF | 9Eh | Keyboard Flag Register | KBF7 | KBF6 | KBF5 | KBF4 | KBF3 | KBF2 | KBF1 | KBF0 |

**Table 11.** EEPROM data Memory SFR (AT89C51ED2 only)

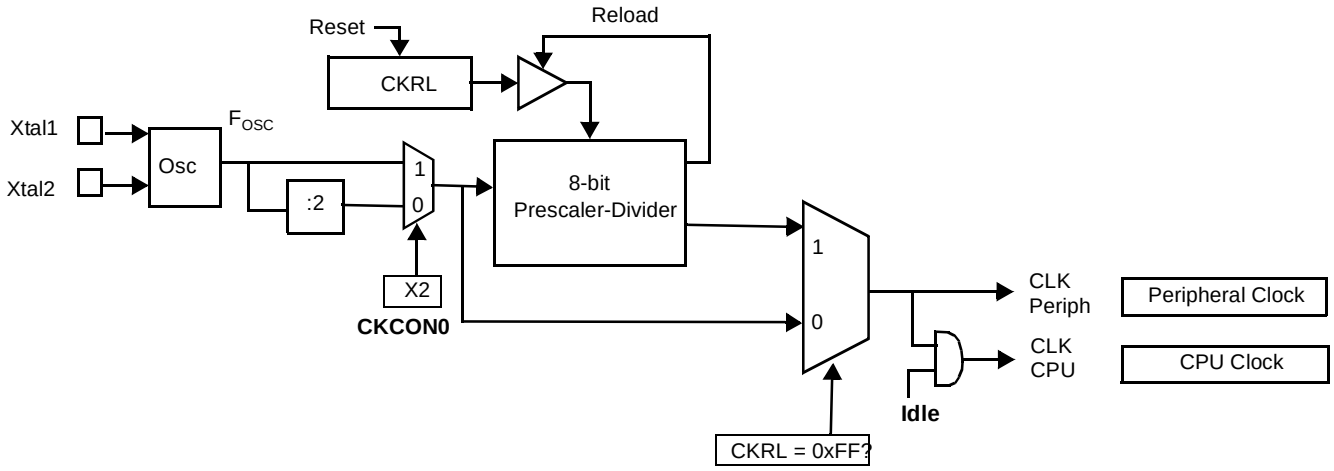
| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| EECON | D2h | EEPROM Data Control | | | | | | | EEE | EEBUSY |

**Table 13.** Pin Description

| Mnemonic | Pin Number PLCC44 | VQFP44 | PLCC68 | VQFP64 | PDIL40 | Type | Name and Function |
|---|---|---|---|---|---|---|---|
| $V_{SS}$ | 22 | 16 | 51 | 40 | 20 | I | **Ground:** 0V reference |
| $V_{CC}$ | 44 | 38 | 17 | 8 | 40 | I | **Power Supply:** This is the power supply voltage for normal, idle and power-down operation |
| P0.0 - P0.7 | 43 - 36 | 37 - 30 | 15, 14, 12, 11, 9,6, 5, 3 | 6, 5, 3, 2, 64, 61,60,59 | 32-39 | I/O | **Port 0**: Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high impedance inputs. Port 0 must be polarized to $V_{CC}$ or $V_{SS}$ in order to prevent any parasitic current consumption. Port 0 is also the multiplexed low-order address and data bus during access to external program and data memory. In this application, it uses strong internal pull-up when emitting 1s. Port 0 also inputs the code bytes during EPROM programming. External pull-ups are required during program verification during which P0 outputs the code bytes. |
| P1.0 - P1.7 | 2 - 9 | 40 - 44 1 - 3 | 19, 21, 22, 23, 25, 27, 28, 29 | 10, 12, 13, 14, 16, 18, 19, 20 | 1-8 | I/O | **Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. Port 1 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally pulled low will source current because of the internal pull-ups. Port 1 also receives the low-order address byte during memory programming and verification. |
| | | | | | | | Alternate functions for AT89C51RD2/ED2 Port 1 include: |
| | 2 | 40 | 19 | 10 | 1 | I/O | **P1.0**: Input/Output |
| | | | | | | I/O | **T2 (P1.0):** Timer/Counter 2 external count input/Clockout |
| | 3 | 41 | 21 | 12 | 2 | I/O | **P1.1:** Input/Output |
| | | | | | | I | **T2EX:** Timer/Counter 2 Reload/Capture/Direction Control |
| | | | | | | I | $\overline{\textbf{SS}}$**:** SPI Slave Select |
| | 4 | 42 | 22 | 13 | 3 | I/O | **P1.2:** Input/Output |
| | | | | | | I | **ECI:** External Clock for the PCA |
| | 5 | 43 | 23 | 14 | 4 | I/O | **P1.3:** Input/Output |
| | | | | | | I/O | **CEX0:** Capture/Compare External I/O for PCA module 0 |
| | 6 | 44 | 25 | 16 | 5 | I/O | **P1.4:** Input/Output |
| | | | | | | I/O | **CEX1:** Capture/Compare External I/O for PCA module 1 |
| | 7 | 1 | 27 | 18 | 6 | I/O | **P1.5:** Input/Output |
| | | | | | | I/O | **CEX2:** Capture/Compare External I/O for PCA module 2 |
| | | | | | | I/O | **MISO:** SPI Master Input Slave Output line<br>When SPI is in master mode, MISO receives data from the slave peripheral. When SPI is in slave mode, MISO outputs data to the master controller. |
| | 8 | 2 | 28 | 19 | 7 | I/O | **P1.6:** Input/Output |
| | | | | | | I/O | **CEX3:** Capture/Compare External I/O for PCA module 3 |
| | | | | | | I/O | **SCK:** SPI Serial Clock |

## Functional Block Diagram

**Figure 4.** Functional Oscillator Block Diagram




**Prescaler Divider**

- A hardware RESET puts the prescaler divider in the following state:
  - CKRL = FFh: $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/2$ (Standard C51 feature)
- Any value between FFh down to 00h can be written by software into CKRL register in order to divide frequency of the selected oscillator:
  - CKRL = 00h: minimum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/1020$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/510$ (X2 Mode)
  - CKRL = FFh: maximum frequency
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}/2$ (Standard Mode)
    $F_{CLK\ CPU} = F_{CLK\ PERIPH} = F_{OSC}$ (X2 Mode)

$F_{CLK\ CPU}$ and $F_{CLK\ PERIPH}$

In X2 Mode, for CKRL<>0xFF:

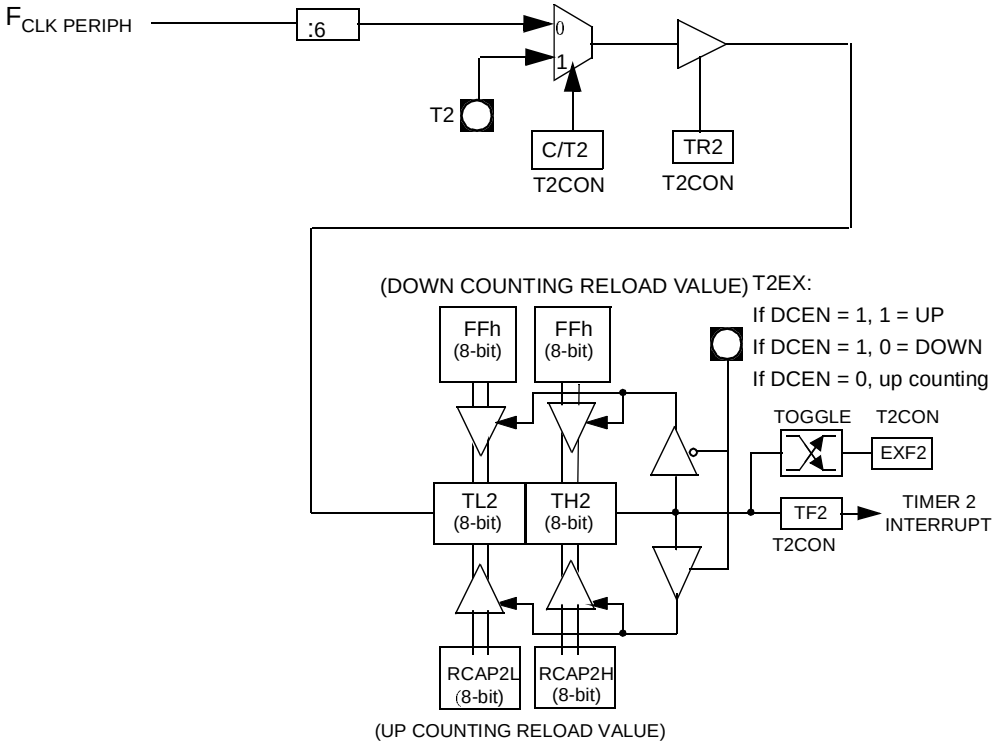
$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSC}}{2 \times (255 - CKRL)}$$

In X1 Mode, for CKRL<>0xFF then:

$$F_{CPU} = F_{CLKPERIPH} = \frac{F_{OSC}}{4 \times (255 - CKRL)}$$

INC is a short (2 bytes) and fast (12 clocks) way to manipulate the DPS bit in the AUXR1 SFR. However, note that the INC instruction does not directly force the DPS bit to a particular state, but simply toggles it. In simple routines, such as the block move example, only the fact that DPS is toggled in the proper sequence matters, not its actual value. In other words, the block move routine works the same whether DPS is '0' or '1' on entry. Observe that without the last instruction (INC AUXR1), the routine will exit with DPS in the opposite state.

**Figure 14.** Auto-reload Mode Up/Down Counter (DCEN = 1)




## Programmable Clock-output

In the clock-out mode, Timer 2 operates as a 50% duty-cycle, programmable clock generator (See Figure 15). The input clock increments TL2 at frequency $F_{CLK\ PERIPH}/2$. The timer repeatedly counts to overflow from a loaded value. At overflow, the contents of RCAP2H and RCAP2L registers are loaded into TH2 and TL2. In this mode, Timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency as a function of the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

$$Clock\text{–}OutFrequency = \frac{F_{CLKPERIPH}}{4 \times (65536 - RCAP2H/RCAP2L)}$$

For a 16 MHz system clock, Timer 2 has a programmable frequency range of 61 Hz ($F_{CLK\ PERIPH}/2^{16}$) to 4 MHz ($F_{CLK\ PERIPH}/4$). The generated clock signal is brought out to T2 pin (P1.0).

Timer 2 is programmed for the clock-out mode as follows:

- Set T2OE bit in T2MOD register.
- Clear C/$\overline{T2}$ bit in T2CON register.
- Determine the 16-bit reload value from the formula and enter it in RCAP2H/RCAP2L registers.
- Enter a 16-bit initial value in timer registers TH2/TL2. It can be the same as the reload value or a different one depending on the application.
- To start the timer, set TR2 run control bit in T2CON register.

It is possible to use Timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers.

**Table 25.** PCA Module Modes (CCAPMn Registers)

| ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMm | ECCFn | Module Function |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | No Operation |
| X | 1 | 0 | 0 | 0 | 0 | X | 16-bit capture by a positive-edge trigger on CEXn |
| X | 0 | 1 | 0 | 0 | 0 | X | 16-bit capture by a negative trigger on CEXn |
| X | 1 | 1 | 0 | 0 | 0 | X | 16-bit capture by a transition on CEXn |
| 1 | 0 | 0 | 1 | 0 | 0 | X | 16-bit Software Timer/Compare mode. |
| 1 | 0 | 0 | 1 | 1 | 0 | X | 16-bit High Speed Output |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 8-bit PWM |
| 1 | 0 | 0 | 1 | X | 0 | X | Watchdog Timer (module 4 only) |

There are two additional registers associated with each of the PCA modules. They are CCAPnH and CCAPnL and these are the registers that store the 16-bit count when a capture occurs or a compare should occur. When a module is used in the PWM mode these registers are used to control the duty cycle of the output (See Table 26 & Table 27).

**Table 26.** CCAPnH Registers (n = 0 - 4)

CCAP0H - PCA Module 0 Compare/Capture Control Register High (0FAh)

CCAP1H - PCA Module 1 Compare/Capture Control Register High (0FBh)

CCAP2H - PCA Module 2 Compare/Capture Control Register High (0FCh)

CCAP3H - PCA Module 3 Compare/Capture Control Register High (0FDh)

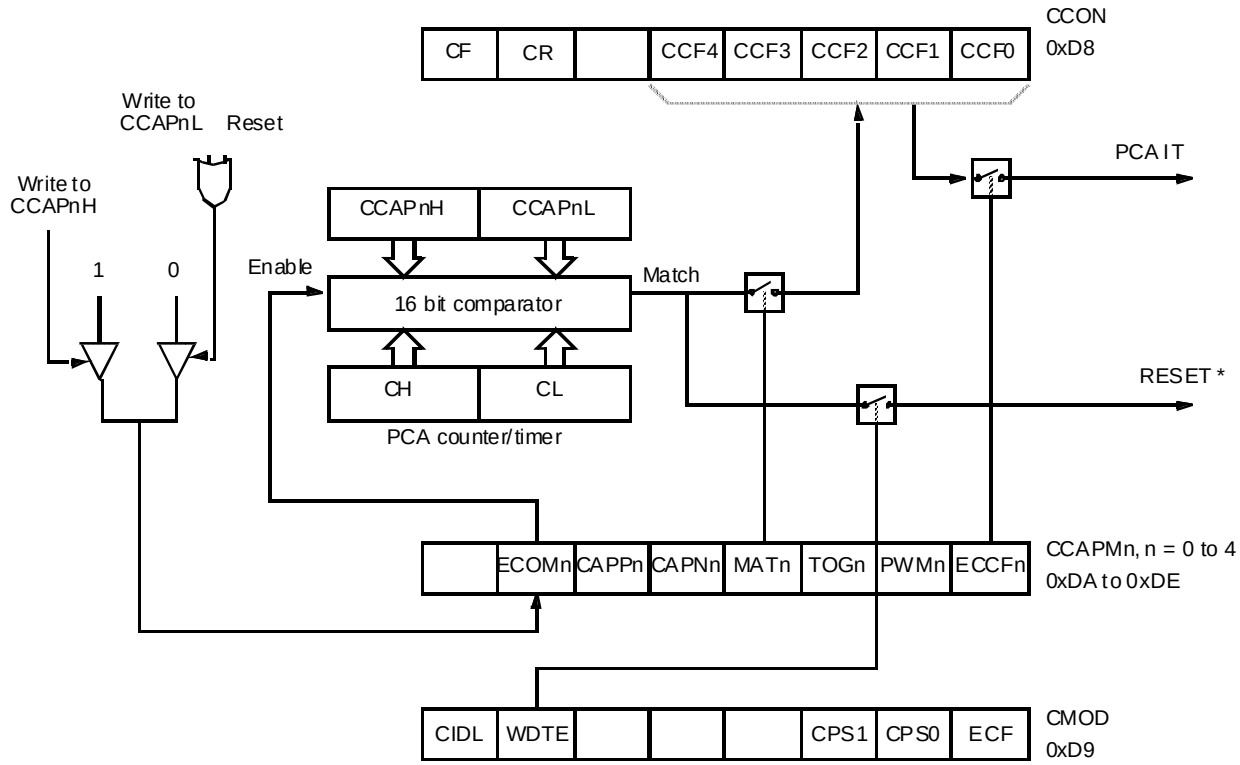
CCAP4H - PCA Module 4 Compare/Capture Control Register High (0FEh)

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

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 - 0 | - | **PCA Module n Compare/Capture Control**<br>CCAPnH Value |

Reset Value = 0000 0000b
Not bit addressable

**Figure 19.** PCA Compare Mode and PCA Watchdog Timer




Before enabling ECOM bit, CCAPnL and CCAPnH should be set with a non zero value, otherwise an unwanted match could happen. Writing to CCAPnH will set the ECOM bit.

Once ECOM is set, writing CCAPnL will clear ECOM so that an unwanted match doesn't occur while modifying the compare value. Writing to CCAPnH will set ECOM. For this reason, user software should write CCAPnL first, and then CCAPnH. Of course, the ECOM bit can still be controlled by accessing to CCAPMn register.

## High Speed Output Mode

In this mode the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the modules capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set (See Figure 20).

A prior write must be done to CCAPnL and CCAPnH before writing the ECOMn bit.

**Table 42.** BDRCON Register

BDRCON - Baud Rate Control Register (9Bh)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | BRR | TBCK | RBCK | SPD | SRC |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | BRR | **Baud Rate Run Control bit**<br>Cleared to stop the internal Baud Rate Generator.<br>Set to start the internal Baud Rate Generator. |
| 3 | TBCK | **Transmission Baud rate Generator Selection bit for UART**<br>Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator.<br>Set to select internal Baud Rate Generator. |
| 2 | RBCK | **Reception Baud Rate Generator Selection bit for UART**<br>Cleared to select Timer 1 or Timer 2 for the Baud Rate Generator.<br>Set to select internal Baud Rate Generator. |
| 1 | SPD | **Baud Rate Speed Control bit for UART**<br>Cleared to select the SLOW Baud Rate Generator.<br>Set to select the FAST Baud Rate Generator. |
| 0 | SRC | **Baud Rate Source select bit in Mode 0 for UART**<br>Cleared to select $F_{OSC}/12$ as the Baud Rate Generator ($F_{CLK\ PERIPH}/6$ in X2 mode).<br>Set to select the internal Baud Rate Generator for UARTs in mode 0. |

Reset Value = XXX0 0000b
Not bit addressable

## Keyboard Interface

The AT89C51RD2/ED2 implements a keyboard interface allowing the connection of a 8 x n matrix keyboard. It is based on 8 inputs with programmable interrupt capability on both high or low level. These inputs are available as alternate function of P1 and allow to exit from idle and power-down modes.

The keyboard interfaces with the C51 core through 3 special function registers: KBLS, the Keyboard Level Selection register (Table 45), KBE, the Keyboard interrupt Enable register (Table 44), and KBF, the Keyboard Flag register (Table 43).

### Interrupt

The keyboard inputs are considered as 8 independent interrupt sources sharing the same interrupt vector. An interrupt enable bit (KBD in IE1) allows global enable or disable of the keyboard interrupt (see Figure 27). As detailed in Figure 28 each keyboard input has the capability to detect a programmable level according to KBLS. x bit value. Level detection is then reported in interrupt flags KBF.x that can be masked by software using KBE. x bits.

This structure allow keyboard arrangement from 1 by n to 8 by n matrix and allows usage of P1 inputs for other purpose.

**Figure 27.** Keyboard Interface Block Diagram

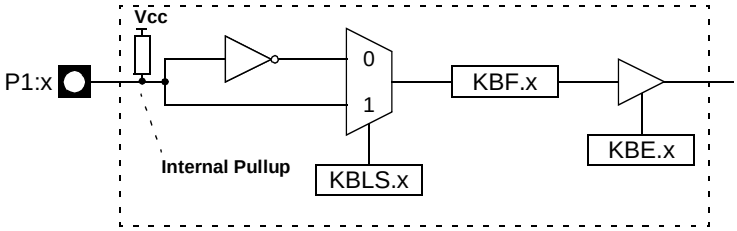
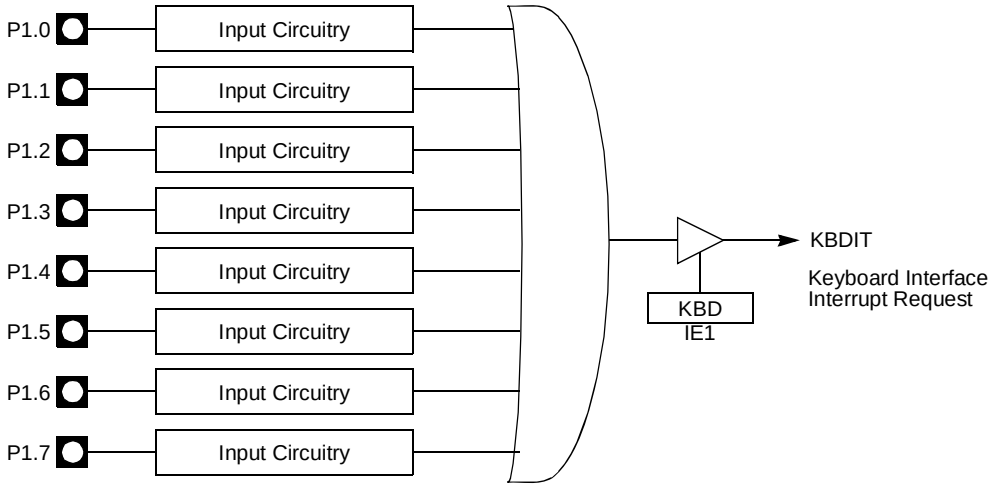



**Figure 28.** Keyboard Input Circuitry




### Power Reduction Mode

P1 inputs allow exit from idle and power-down modes as detailed in Section “Power Management”, page 82.

**Table 45.** KBLS Register

KBLS-Keyboard Level Selector Register (9Ch)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| KBLS7 | KBLS6 | KBLS5 | KBLS4 | KBLS3 | KBLS2 | KBLS1 | KBLS0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | KBLS7 | **Keyboard line 7 Level Selection bit**<br>Cleared to enable a low level detection on Port line 7.<br>Set to enable a high level detection on Port line 7. |
| 6 | KBLS6 | **Keyboard line 6 Level Selection bit**<br>Cleared to enable a low level detection on Port line 6.<br>Set to enable a high level detection on Port line 6. |
| 5 | KBLS5 | **Keyboard line 5 Level Selection bit**<br>Cleared to enable a low level detection on Port line 5.<br>Set to enable a high level detection on Port line 5. |
| 4 | KBLS4 | **Keyboard line 4 Level Selection bit**<br>Cleared to enable a low level detection on Port line 4.<br>Set to enable a high level detection on Port line 4. |
| 3 | KBLS3 | **Keyboard line 3 Level Selection bit**<br>Cleared to enable a low level detection on Port line 3.<br>Set to enable a high level detection on Port line 3. |
| 2 | KBLS2 | **Keyboard line 2 Level Selection bit**<br>Cleared to enable a low level detection on Port line 2.<br>Set to enable a high level detection on Port line 2. |
| 1 | KBLS1 | **Keyboard line 1 Level Selection bit**<br>Cleared to enable a low level detection on Port line 1.<br>Set to enable a high level detection on Port line 1. |
| 0 | KBLS0 | **Keyboard line 0 Level Selection bit**<br>Cleared to enable a low level detection on Port line 0.<br>Set to enable a high level detection on Port line 0. |

Reset Value = 0000 0000b

**Table 57.** IPL1 Register

IPL1 - Interrupt Priority Register (B2h)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | SPIL | TWIL | KBDL |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 2 | SPIL | **SPI interrupt Priority bit**<br>Refer to SPIH for priority level. |
| 1 | - | **Reserved**<br>The value read from this bit is indeterminate. Do not set this bit. |
| 0 | KBDL | **Keyboard interrupt Priority bit**<br>Refer to KBDH for priority level. |

Reset Value = XXXX X000b
Bit addressable

3. Generate an enabled external Keyboard interrupt (same behavior as external interrupt).

Note: During the time that execution resumes, the internal RAM cannot be accessed; however, it is possible for the Port pins to be accessed. To avoid unexpected outputs at the Port pins, the instruction immediately following the instruction that activated the Power-Down mode should not write to a Port pin or to the external RAM.

Note: Exit from power-down by reset redefines all the SFRs, but does not affect the internal RAM content.

**Table 59.** Pin Conditions in Special Operating Modes

| Mode | Port 0 | Port 1 | Port 2 | Port 3 | Port 4 | ALE | PSEN# |
|---|---|---|---|---|---|---|---|
| Reset | Floating | High | High | High | High | High | High |
| Idle (internal code) | Data | Data | Data | Data | Data | High | High |
| Idle (external code) | Floating | Data | Data | Data | Data | High | High |
| Power-Down (internal code) | Data | Data | Data | Data | Data | Low | Low |
| Power-Down (external code) | Floating | Data | Data | Data | Data | Low | Low |

**Table 68.** Program Lock Bits

| Program Lock Bits | | | | |
|---|---|---|---|---|
| **Security Level** | **LB0** | **LB1** | **LB2** | **Protection Description** |
| 1 | U | U | U | No program lock features enabled. |
| 2 | P | U | U | MOVC instruction executed from external program memory is disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further parallel programming of the on chip code memory is disabled.<br>ISP and software programming with API are still allowed. |
| 3 | X | P | U | Same as 2, also verify code memory through parallel programming interface is disabled. |
| 4 | X | X | P | Same as 3, also external execution is disabled (Default). |

Note: U: Unprogrammed or "one" level.
P: Programmed or "zero" level.
X: Do not care
WARNING: Security level 2 and 3 should only be programmed after Flash and code verification.

These security bits protect the code access through the parallel programming interface. They are set by default to level 4. The code access through the ISP is still possible and is controlled by the "software security bits" which are stored in the extra Flash memory accessed by the ISP firmware.

To load a new application with the parallel programmer, a chip erase must first be done. This will set the HSB in its inactive state and will erase the Flash memory. The part reference can always be read using Flash parallel programming modes.

**Default Values**

The default value of the HSB provides parts ready to be programmed with ISP:

- BLJB: Programmed force ISP operation.
- X2: Unprogrammed to force X1 mode (Standard Mode).
- XRAM: Unprogrammed to valid XRAM
- LB2-0: Security level four to protect the code from a parallel access with maximum security.

**Software Registers**

Several registers are used in factory and by parallel programmers. These values are used by Atmel ISP.

These registers are in the "Extra Flash Memory" part of the Flash memory. This block is also called "XAF" or eXtra Array Flash. They are accessed in the following ways:

- Commands issued by the parallel memory programmer.
- Commands issued by the ISP software.
- Calls of API issued by the application software.

Several software registers are described in Table 69.

**Table 69.** Default Values

| Mnemonic | Definition | Default value | Description |
|---|---|---|---|
| SBV | Software Boot Vector | FCh | |
| BSB | Boot Status Byte | 0FFh | |
| SSB | Software Security Byte | FFh | |
| | Copy of the Manufacturer Code | 58h | Atmel |
| | Copy of the Device ID #1: Family Code | D7h | C51 X2, Electrically Erasable |
| | Copy of the Device ID #2: Memories Size and Type | ECh | AT89C51RD2/ED2 64KB |
| | Copy of the Device ID #3: Name and Revision | EFh | AT89C51RD2/ED2 64KB, Revision 0 |

After programming the part by ISP, the BSB must be cleared (00h) in order to allow the application to boot at 0000h.

The content of the Software Security Byte (SSB) is described in Table 70 and Table 71.

To assure code protection from a parallel access, the HSB must also be at the required level.

**Table 70.** Software Security Byte

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | LB1 | LB0 |

| Bit Number | Bit Mnemonic | Description |
|---|---|---|
| 7 | - | **Reserved**<br>Do not clear this bit. |
| 6 | - | **Reserved**<br>Do not clear this bit. |
| 5 | - | **Reserved**<br>Do not clear this bit. |
| 4 | - | **Reserved**<br>Do not clear this bit. |
| 3 | - | **Reserved**<br>Do not clear this bit. |
| 2 | - | **Reserved**<br>Do not clear this bit. |
| 1-0 | LB1-0 | User Memory Lock Bits<br>See Table 71 |

The two lock bits provide different levels of protection for the on-chip code and data, when programmed as shown in Table 71.

**Bootloader Functionality**

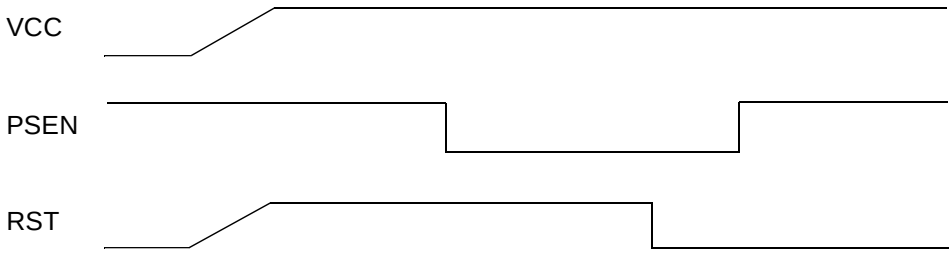
The bootloader can be activated by two means: Hardware conditions or regular boot process.

The Hardware conditions (EA = 1, PSEN = 0) during the Reset# falling edge force the on-chip bootloader execution. This allows an application to be built that will normally execute the end user's code but can be manually forced into default ISP operation.

As PSEN is a an output port in normal operating mode after reset, user application should take care to release PSEN after falling edge of reset signal. The hardware conditions are sampled at reset signal falling edge, thus they can be released at any time when reset input is low.

To ensure correct microcontroller startup, the PSEN pin should not be tied to ground during power-on (See Figure 43).

**Figure 43.** Hardware conditions typical sequence during power-on.




The on-chip bootloader boot process is shown Figure 44.

**Table 72.** Bootloader Process Description

| | **Purpose** |
|---|---|
| Hardware Conditions | The Hardware Conditions force the bootloader execution whatever BLJB, BSB and SBV values. |
| BLJB | The Boot Loader Jump Bit forces the application execution.<br>BLJB = 0 => Bootloader execution<br>BLJB = 1 => Application execution<br>The BLJB is a fuse bit in the Hardware Byte.<br>It can be modified by hardware (programmer) or by software (API).<br>Note: The BLJB test is performed by hardware to prevent any program execution. |
| SBV | The Software Boot Vector contains the high address of customer bootloader stored in the application.<br>SBV = FCh (default value) if no customer bootloader in user Flash.<br>Note: The customer bootloader is called by JMP [SBV]00h instruction. |

## Functional Description

### Software Security Bits (SSB)

The SSB protects any Flash access from ISP command.
The command "Program Software Security Bit" can only write a higher priority level.

There are three levels of security:

- level 0: ***NO_SECURITY*** (FFh)

This is the default level.
From level 0, one can write level 1 or level 2.

- level 1: ***WRITE_SECURITY*** (FEh)

For this level it is impossible to write in the Flash memory, BSB and SBV.
The Bootloader returns 'P' on write access.
From level 1, one can write only level 2.

- level 2: ***RD_WR_SECURITY*** (FCh

The level 2 forbids all read and write accesses to/from the Flash/EEPROM memory.

The Bootloader returns 'L' on read or write access.

Only a full chip erase in parallel mode (using a programmer) or ISP command can reset the software security bits.

From level 2, one cannot read and write anything.

**Table 73.** Software Security Byte Behavior

| | Level 0 | Level 1 | Level 2 |
|---|---|---|---|
| Flash/EEPROM | Any access allowed | Read-only access allowed | Any access not allowed |
| Fuse Bit | Any access allowed | Read-only access allowed | Any access not allowed |
| BSB & SBV | Any access allowed | Read-only access allowed | Any access not allowed |
| SSB | Any access allowed | Write level 2 allowed | Read-only access allowed |
| Manufacturer Info | Read-only access allowed | Read-only access allowed | Read-only access allowed |
| Bootloader Info | Read-only access allowed | Read-only access allowed | Read-only access allowed |
| Erase Block | Allowed | Not allowed | Not allowed |
| Full Chip Erase | Allowed | Allowed | Allowed |
| Blank Check | Allowed | Allowed | Allowed |

**Autobaud Performances**

The ISP feature allows a wide range of baud rates in the user application. It is also adaptable to a wide range of oscillator frequencies. This is accomplished by measuring the bit-time of a single bit in a received character. This information is then used to program the baud rate in terms of timer counts based on the oscillator frequency. The ISP feature requires that an initial character (an uppercase U) be sent to the AT89C51RD2/ED2 to establish the baud rate. Table show the autobaud capability.

**Table 74.** Autobaud Performances

| **Frequency (MHz)**<br>**Baudrate (kHz)** | **1.8432** | **2** | **2.4576** | **3** | **3.6864** | **4** | **5** | **6** | **7.3728** |
|---|---|---|---|---|---|---|---|---|---|
| 2400 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 4800 | OK | - | OK | OK | OK | OK | OK | OK | OK |
| 9600 | OK | - | OK | OK | OK | OK | OK | OK | OK |
| 19200 | OK | - | OK | OK | OK | - | - | OK | OK |
| 38400 | - | - | OK | | OK | - | OK | OK | OK |
| 57600 | - | - | - | - | OK | - | - | - | OK |
| 115200 | - | - | - | - | - | - | - | - | OK |
| | | | | | | | | | |
| **Frequency (MHz)**<br>**Baudrate (kHz)** | **8** | **10** | **11.0592** | **12** | **14.746** | **16** | **20** | **24** | **26.6** |
| 2400 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 4800 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 9600 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 19200 | OK | OK | OK | OK | OK | OK | OK | OK | OK |
| 38400 | - | - | OK | OK | OK | OK | OK | OK | OK |
| 57600 | - | - | OK | - | OK | OK | OK | OK | OK |
| 115200 | - | - | OK | - | OK | - | - | - | - |

**Command Data Stream Protocol**

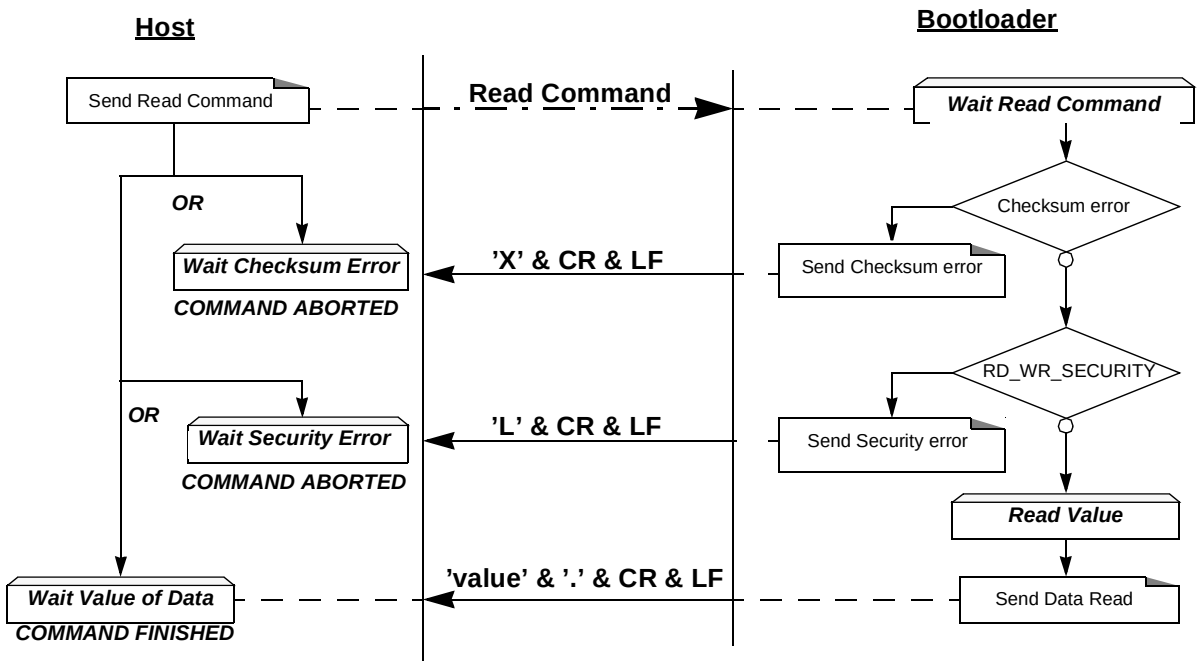
All commands are sent using the same flow. Each frame sent by the host is echoed by the bootloader.

**Read Function Description**

This flow is similar for the following frames:

- Reading Frame
- EOF Frame/ Atmel Frame (only reading Atmel Frame)

**Figure 51.** Read Flow




*Example*

Read function (read SBV)

**HOST** : 02 0000 05 07 02 F0

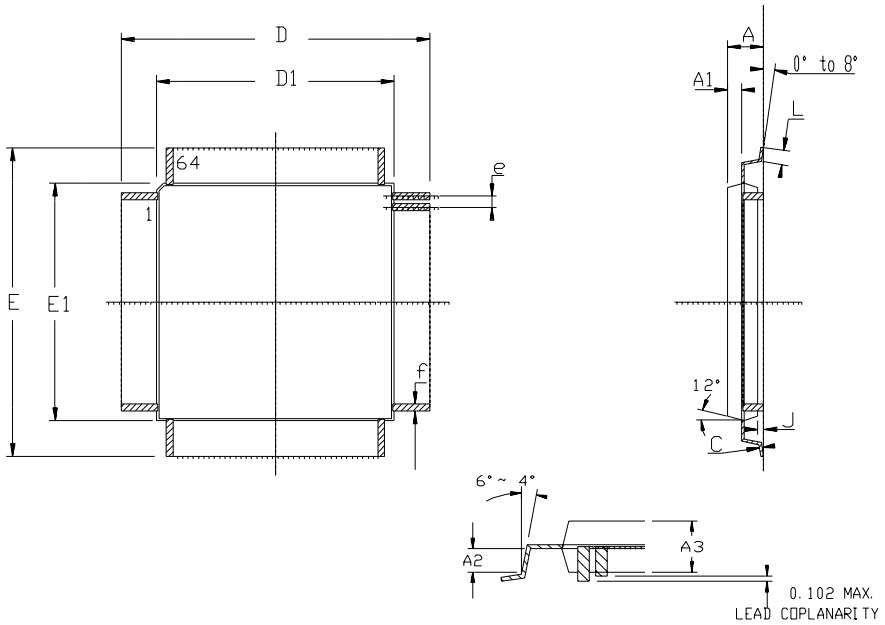
**BOOTLOADER** : 02 0000 05 07 02 F0 Value . CR LF

Atmel Read function (read Bootloader version)

**HOST** : 02 0000 01 02 00 FB

**BOOTLOADER** : 02 0000 01 02 00 FB Value . CR LF

## VQFP64




| | MM | | INCH | |
|---|---|---|---|---|
| | Min | Max | Min | Max |
| A | - | 1.60 | - | .063 |
| A1 | 0.64 REF | | .025 REF | |
| A2 | 0.64 REF | | .025 REF | |
| A3 | 1.35 | 1.45 | .053 | .057 |
| D | 11.75 | 12.25 | .463 | .483 |
| D1 | 9.90 | 10.10 | .390 | .398 |
| E | 11.75 | 12.25 | .463 | .483 |
| E1 | 9.90 | 10.10 | .390 | .398 |
| J | 0.05 | - | .002 | - |
| L | 0.45 | 0.75 | .018 | .030 |
| e | 0.50 BSC | | .0197 BSC | |
| f | 0.25 BSC | | .010 BSC | |

***Serial I/O Port ..... 49***
Framing Error Detection ..... 49
Automatic Address Recognition..... 50
Registers..... 52
Baud Rate Selection for UART for Mode 1 and 3..... 52
UART Registers..... 55

***Keyboard Interface ..... 60***
Registers..... 61

***Serial Port Interface (SPI) ..... 64***
Features..... 64
Signal Description..... 64
Functional Description ..... 66

***Interrupt System ..... 73***
Registers..... 74
Interrupt Sources and Vector Addresses..... 75

***Power Management ..... 82***
Introduction..... 82
Idle Mode..... 82
Power-Down Mode ..... 82
Registers..... 85

***Hardware Watchdog Timer ..... 86***
Using the WDT ..... 86
WDT during Power-down and Idle..... 87

***ONCE® Mode (ON- Chip Emulation) ..... 88***

***Power-off Flag ..... 89***

***Reduced EMI Mode ..... 90***

***EEPROM Data Memory ..... 91***
Write Data..... 91
Read Data..... 93
Registers..... 94

***Flash/EEPROM Memory ..... 95***
Features..... 95
Flash Programming and Erasure..... 95
Flash Registers and Memory Map..... 96
Flash Memory Status..... 99
Memory Organization ..... 99