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"

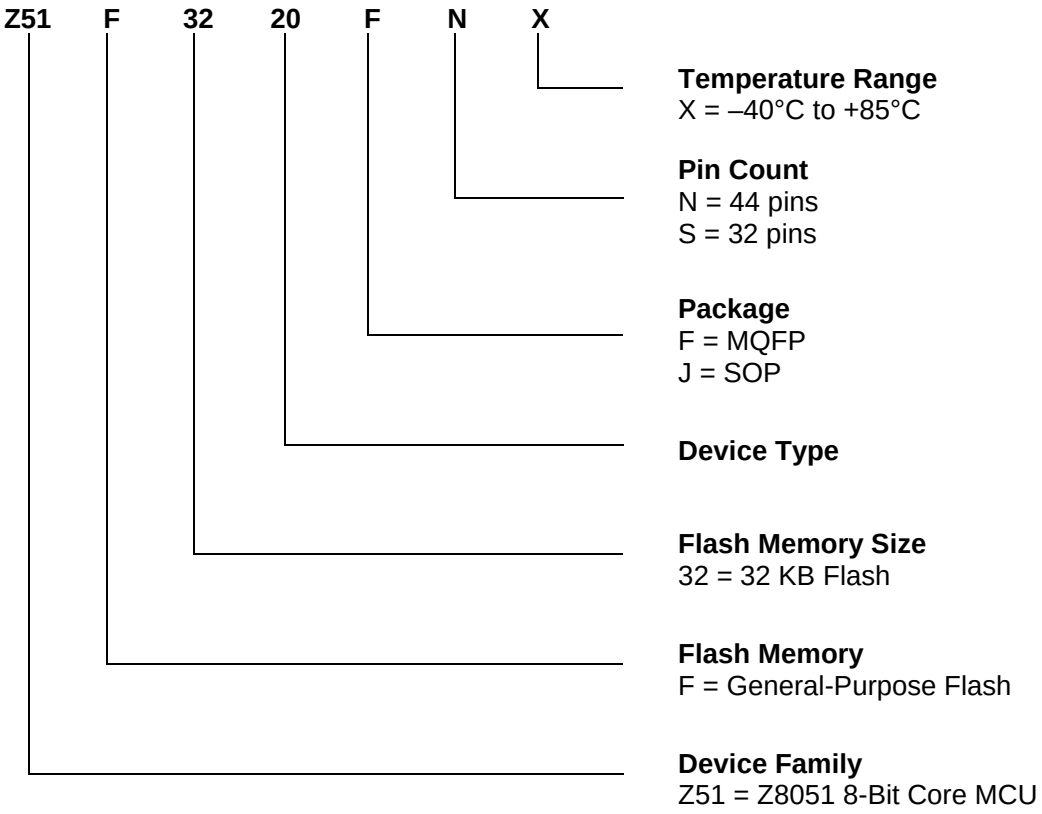
Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	16MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	LCD, LVD, POR, PWM, WDT
Number of I/O	42
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-BQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z51f3220fnx

15.1 Overview 293
16. Configure Option 304
16.1 Configure Option Control 304
17. APPENDIX 305

1.3.1 Part Number Suffix Designation

Zilog part numbers consist of a number of components, as indicated in the following example.

Example: Part number Z51F3220FNX is an 8-bit MCU with 32 KB of Flash memory and 1 KB of RAM in a 44-pin MQFP package and operating within a –40°C to +85°C temperature range. In accordance with RoHS standards, this device has been built using lead-free solder.



1.4 Development Tools

1.4.1 Compiler

We do not provide the compiler. Please contact the third parties.

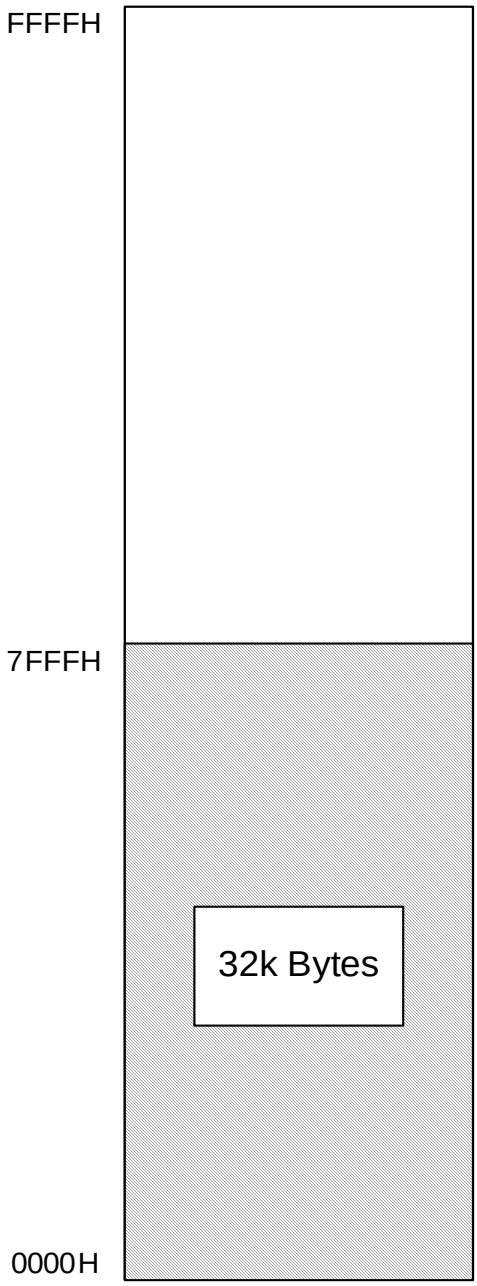
The core of Z51F3220 is Mentor 8051. And, device ROM size is smaller than 32k bytes. Developer can use all kinds of third party's standard 8051 compiler.

1.4.2 OCD Emulator and Debugger

The OCD (On Chip Debug) emulator supports Zilog's 8051 series MCU emulation.

The OCD interface uses two-wire interfacing between PC and MCU which is attached to user's system. The OCD can read or change the value of MCU internal memory and I/O peripherals. And the OCD also controls MCU internal debugging logic, it means OCD controls emulation, step run, monitoring, etc.

The OCD Debugger program works on Microsoft-Windows NT, 2000, XP, Vista (32bit) operating system.



- 32k Bytes Including Interrupt Vector Region

Figure 8.1 Program Memory

11. Peripheral Hardware

11.1 Clock Generator

11.1.1 Overview

As shown in Figure 11.1, the clock generator produces the basic clock pulses which provide the system clock to be supplied to the CPU and the peripheral hardware. It contains main/sub-frequency clock oscillator. The main/sub clock operation can be easily obtained by attaching a crystal between the XIN/SXIN and XOUT/SXOUT pin, respectively. The main/sub clock can be also obtained from the external oscillator. In this case, it is necessary to put the external clock signal into the XIN/SXIN pin and open the XOUT/SXOUT pin. The default system clock is 1MHz INT-RC Oscillator and the default division rate is eight. In order to stabilize system internally, it is used 1MHz INT-RC oscillator on POR.

- Calibrated Internal RC Oscillator (16 MHz)
  - . INT-RC OSC/1 (16 MHz)
  - . INT-RC OSC/2 (8 MHz)
  - . INT-RC OSC/4 (4 MHz)
  - . INT-RC OSC/8 (2 MHz)
  - . INT-RC OSC/16 (1 MHz, Default system clock)
  - . INT-RC OSC/32 (0.5 MHz)
- Main Crystal Oscillator (0.4~12 MHz)
- Sub Crystal Oscillator (32.768 kHz)
- Internal WDTRC Oscillator (5 kHz)

11.1.2 Block Diagram

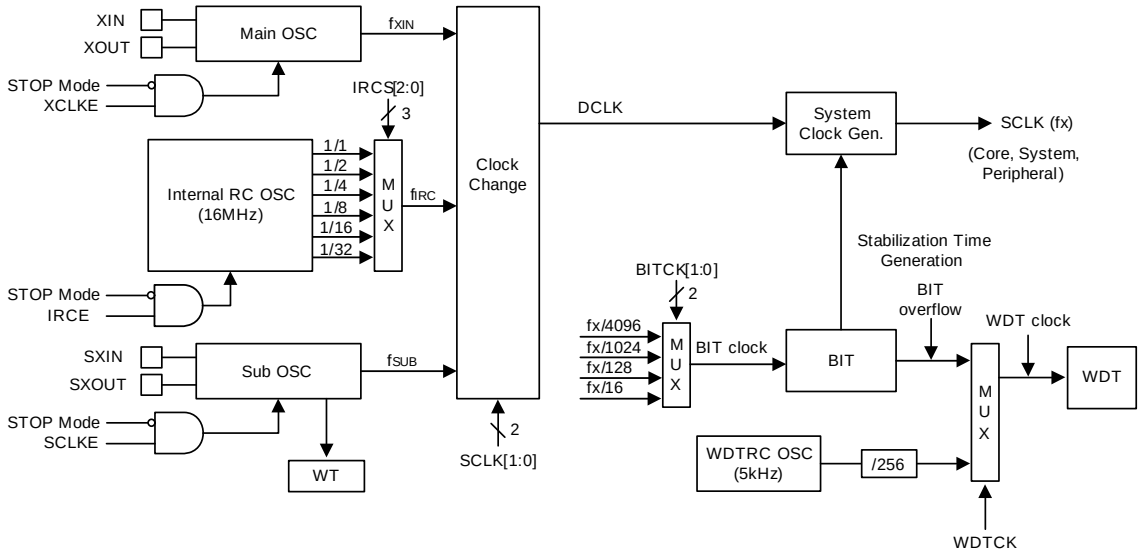
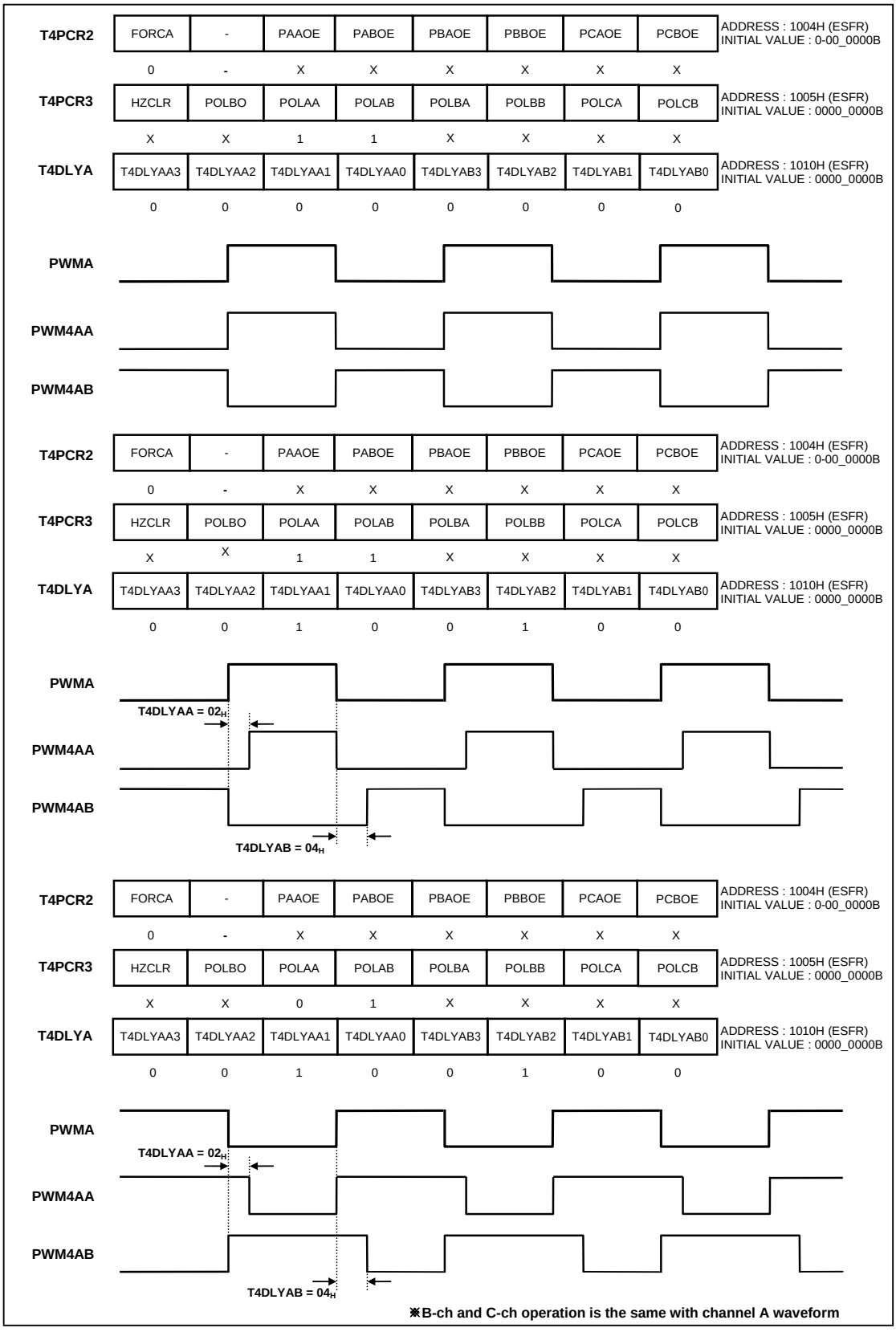


Figure 11.1 Clock Generator Block Diagram



※B-ch and C-ch operation is the same with channel A waveform

11.10.3 Data Transmit / Receive Operation

User can use SPI 2 for serial data communication by following step

1. Select SPI 2 operation mode(master/slave, polarity, phase) by control register SPICR.
2. When the SPI 2 is configured as a Master, it selects a Slave by SS2 signal (active low).

   When the SPI 2 is configured as a Slave, it is selected by SS2 signal incoming from Master
3. When the user writes a byte to the data register SPIDR, SPI 2 will start an operation.
4. In this time, if the SPI 2 is configured as a Master, serial clock will come out of SCK2 pin. And Master shifts the eight bits into the Slave (transmit), Slave shifts the eight bits into the Master at the same time (receive). If the SPI 2 is configured as a Slave, serial clock will come into SCK2 pin. And Slave shifts the eight bits into the Master (transmit), Master shifts the eight bits into the Slave at the same time (receive).
5. When transmit/receive is done, SPIIFR bit will be set. If the SPI 2 interrupt is enabled, an interrupt is requested. And SPIIFR bit is cleared by hardware when executing the corresponding interrupt. If SPI 2 interrupt is disable, SPIIFR bit is cleared when user read the status register SPISR, and then access (read/write) the data register SPIDR.

11.10.4 SS2 pin function

1. When the SPI 2 is configured as a Slave, the SS2 pin is always input. If LOW signal come into SS2 pin, the SPI 2 logic is active. And if ‘HIGH’ signal come into SS2 pin, the SPI 2 logic is stop. In this time, SPI 2 logic will be reset, and invalidated any received data.
2. When the SPI 2 is configured as a Master, the user can select the direction of the SS2 pin by port direction register (P17IO). If the SS2 pin is configured as an output, user can use general P17IO output mode. If the SS2 pin is configured as an input, ‘HIGH’ signal must come into SS2 pin to guarantee Master operation. If ‘LOW’ signal come into SS2 pin, the SPI 2 logic interprets this as another master selecting the SPI 2 as a slave and starting to send data to it. To avoid bus contention, MSB bit of SPICR will be cleared and the SPI 2 becomes a Slave and then, SPIIFR bit of SPISR will be set, and if the SPI 2 interrupt is enabled, an interrupt is requested.

NOTES)

- When the SS2 pin is configured as an output at Master mode, SS2 pin’s output value is defined by user’s software (P17IO). Before SPICR setting, the direction of SS2 pin must be defined
- If you don’t need to use SS2 pin, clear the SSENA bit of SPISR. So, you can use disabled pin by P17IO freely. In this case, SS2 signal is driven by ‘HIGH’ or ‘LOW’ internally. In other words, master is ‘HIGH’, salve is ‘LOW’
- When SS2 pin is configured as input, if ‘HIGH’ signal come into SS2 pin, SS_HIGH flag bit will be set. And you can clear it by writing ‘0’.

USI0ST2 (USI0 Status Register 2: For I2C mode) : E2H

7	6	5	4	3	2	1	0
GCALL0	TEND0	STOPD0	SSEL0	MLOST0	BUSY0	TMODE0	RXACK0
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00H

GCALL0(NOTE) This bit has different meaning depending on whether I2C is master or slave. When I2C is a master, this bit represents whether it received AACK (address ACK) from slave.

0 No AACK is received (Master mode)

1 AACK is received (Master mode)

When I2C is a slave, this bit is used to indicated general call.

0 General call address is not detected (Slave mode)

1 General call address is detected (Slave mode)

TEND0(NOTE) This bit is set when 1-byte of data is transferred completely

0 1 byte of data is not completely transferred

1 1 byte of data is completely transferred

STOPD0(NOTE) This bit is set when a STOP condition is detected.

0 No STOP condition is detected

1 STOP condition is detected

SSEL0(NOTE) This bit is set when I2C is addressed by other master.

0 I2C is not selected as a slave

1 I2C is addressed by other master and acts as a slave

MLOST0(NOTE) This bit represents the result of bus arbitration in master mode.

0 I2C maintains bus mastership

1 I2C maintains bus mastership during arbitration process

BUSY0 This bit reflects bus status.

0 I2C bus is idle, so a master can issue a START condition

1 I2C bus is busy

TMODE0 This bit is used to indicate whether I2C is transmitter or receiver.

0 I2C is a receiver

1 I2C is a transmitter

RXACK0 This bit shows the state of ACK signal

0 No ACK is received

1 ACK is received at ninth SCL period

NOTE) These bits can be source of interrupt.

When an I2C interrupt occurs except for STOP mode, the SCL0 line is hold LOW. To release SCL0, write rbitrary value to USI0ST2. When USI0ST2 is written, the TEND0, STOPD0, SSEL0, MLOST0, and RXACK0 bits are cleared.