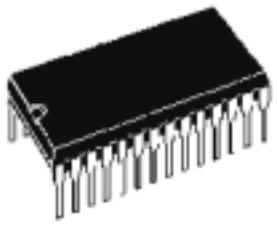
ST52T440/E440

8-Bit *DualLogic*™ MICROCONTROLLER with Timer, Triac/PWM Driver and Analog Comparator

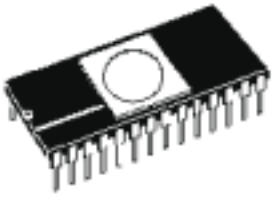
PRELIMINARY DATA

- Digital Microcontroller with embedded fuzzy computational capabilities
- Up to 8 K bytes internal EPROM, up to 256 bytes of Data RAM
- 13 or 21 Configurable I/O PINs
- Operates “STAND ALONE”
- On-Chip Clock Oscillator, with up to 20 MHz working frequencies.
- On-chip Analog Comparator and 16 bit Timer allowing to easily implement a single slope A/D converter with up to 12 bit resolution
- Bandgap reference 2.5 V
- One Programmable Timer/PWM with internal 16-bit Prescaler and 8-bit counter featuring:
  - 1 Input Capture
  - 1 Output Compare
  - External/Internal Clock
  - PWM output and Pulse generator mode
- TRIAC/PWM driver with high current capabilities
- Watchdog Timer
- On chip Power On Reset (POR) and Brown Out Detector
- Hardware multiplication and division
- 46 basic instructions
- Capability to perform boolean and arithmetic operations.
- Power Saving Features.
- Software tools and Emulators availability
- Windowed and One Time Programmable EPROM parts available for prototyping and production phases
- User programmable option byte configuration.
- Standard TTL compatible input
- CMOS compatible output



PDIP28

PS028



CDIP28W

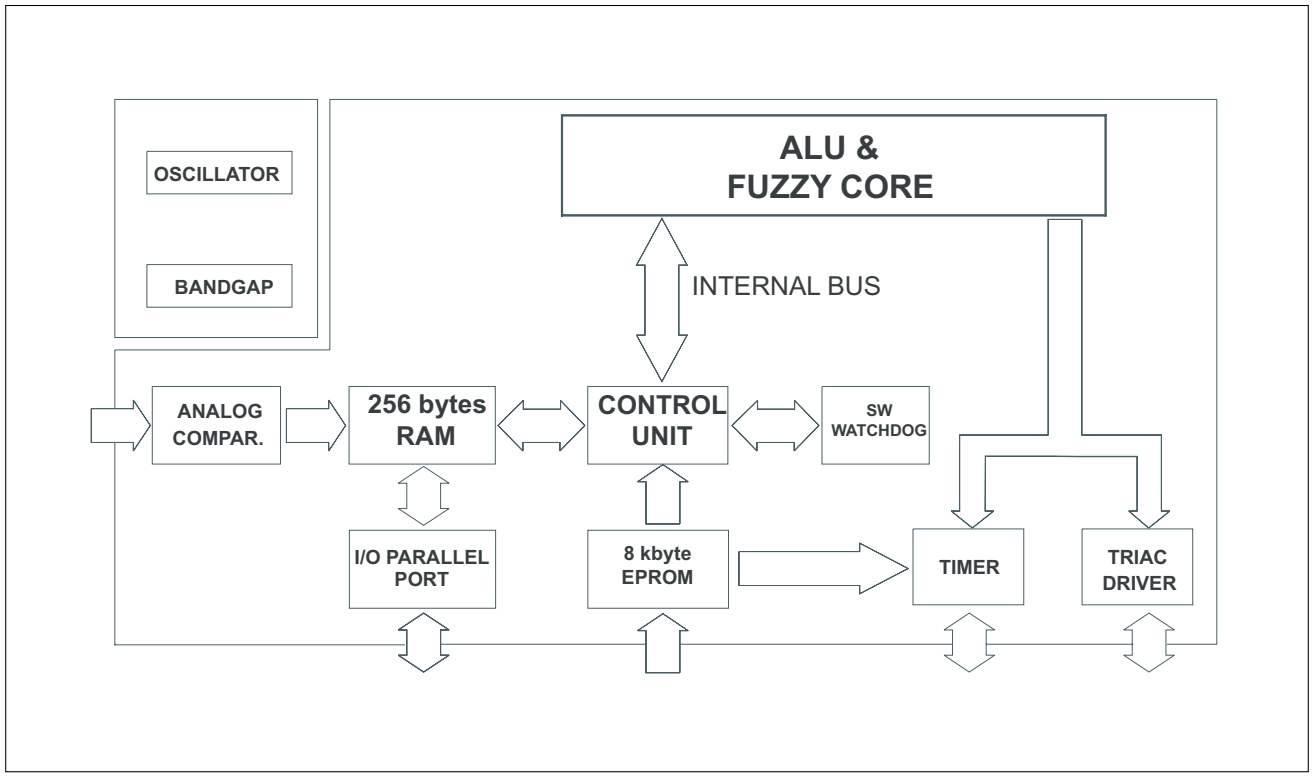
January 2001

Table 1.1 - ST52440 Device Summary

ST52 Device	NVM (bytes)	RAM (bytes)	TIMER/PWM	Analog Comparator	Triac Driver/	WDT	POR, BOR	Package
ST52T440IO	1K	128	1	4 ch	1	Yes	Yes	SO20,PDIP20
ST52T440I1	2K	128						
ST52T440I2	4K	256						
ST52T440I3	8K	256						
ST52T440GO	1K	128		6 ch				SO28,PDIP28
ST52T440G1	2K	128						
ST52T440G2	4K	256						
ST52T440G3	8K	256						

COMMON FEATURES	**ST52x440G/440I**
Temperature Range	-40 to + 85 °C
Operating Supply	5.0 V
CPU Frequency	Up to 20 MHz

Figure 1.1 ST52x440 Architectural Block Diagram



1. GENERAL DESCRIPTION

1.1 Introduction

ST52x440 are devices of **STFive** family of 8-bit *DuaLogic™* microcontrollers.They are able to perform, in an efficient way, both boolean and fuzzy algorithms, in order to reach the best performances that the two methodologies allow.

Produced by STMicroelectronics using the reliable high performance CMOS process, ST52x440 include integrated-on-chip peripherals that allow maximization of system reliability, decrease the system cost minimizing the number of external components.

The flexible I/O configuration of ST52x440 allows to interface with a wide range of external devices, like D/A converters or power control devices.

ST52x440 pins are configurable (Alternate Functions), allowing to set the input or output signals on each single pin (Figures 1.2 *and 1.3.)*

Hardware multiplier (8 bit by 8 bit with 16 bit result) and divider (16 bit over 8 bit with 8 bit result and 8 bit remainder) are available to implement complex functions by using a single instruction thus optimizing the program memory utilization and the computational speed.

Fuzzy Logic dedicated structures in ST52x440 microcontroller can be exploited to model complex system with high accuracy in a useful and easy way.

Fuzzy Expert Systems for overall system management and fuzzy Real time Controls can be designed to increase performances at very competitive costs.

The linguistic approach characterizing Fuzzy Logic is based on a set of IF-THEN rules, describing the control behavior, and on Membership Functions associated to input and output variables.

Up to 341 Membership Functions, with triangular and trapezoidal shapes, or singleton values are available to describe fuzzy variables.

The TIMER/PWM peripheral allows to manage power devices and timing signals, implementing different operating modes and high frequency PWM (Pulse With Modulation) controls. Input Capture and Output Compare functions are available on the TIMER.

The programmable Timer has a 16 bit Internal Prescaler and an 8 bit Counter. It can use internal or external START/STOP signals and clock.

An internal programmable WATCHDOG is available to avoid loop errors and to reset the microcontroller.

An Analog Comparator with a 4 or 6 channels multiplexer is available on ST52x440 family devices. This analog peripheral allows to easily implement an high resolution A/D conversion. By using only an external capacitor it is possible to configure this peripheral in order to get up to 12 bit A/D converter resolution. It includes a 2.5 V bandgap reference for A/D conversion calibration, *that can be used externally for signal conditioning.*

An on-chip TRIAC driver peripheral allows to manage directly power devices, implementing two different operating mode: Burst Mode (i.e. Thermal Applications), Phase Angle Partialization (i.e. Motors Control by Triacs). The TRIAC Driver also allows to generate a PWM signal.

ST52x440 family also includes an on-chip Power-on-Reset (POR) which provides an internal chip reset during power up situation and a Brown-Out Detector (BOR), that allows to reset the microcontroller if the voltage source V_{DD} dips below a minimum value.

To optimize energy consumption, three different power saving modes are available: Wait mode and Halt mode.

The EPROM contains the microcontroller configuration, in terms number of I/O, microcode, Fuzzy Rules and Membership Functions (MFs).

The EPROM can be locked by the user to prevent external undesired operations.

It is possible to perform operations on the data stored in the RAM, allowing to directly combine new inputs and feedback data. All bytes of RAM are used like Register File.

ST52x440 exploits a STMicroelectronics patented strategy to store the MFs in its internal memory.

OTP (One Time Programmable) version devices are fully compatible with the EPROM windowed version, which may be used forprototyping and pre-production phases of development.

A powerful development environment consisting of a board and software tools allows an easy configuration and use of ST52x440.

FUZZYSTUDIO 4.0 software tool allows development of projects through a user-friendly graphical interface and optimization of generated microcode.

1.2 FUNCTIONAL DESCRIPTION

ST52x440 microcontrollers work in two modes

- Memory Programming Phase
- Working Phase

according to control signal levels (see pins description) RESET and Vpp.

Note: ***When RESET = 0 the sequence it is advisable not to use the sequence "101010" to port PA (7 : 2).***

1.2.1 Memory Programming Phase

ST52x440 memory is loaded in Memory Programming Phase. All fuzzy and standard instructions are written inside the memory.

This phase starts with the setting of the control signals as in Table 1.2

When this phase starts, ST52x440 core is set to the RESET status. This allows to program and/or to test the internal Eprom. The signal INC_ADD is used to increment memory address (see Eprom programming).

Table 1.2 Control Signals setting

Control Signal	Programming	Reset	Working
RESET	0	0	1
Vpp	5 V /12 V	0	0

1.2.2 Working Mode

The processor starts the working phase following the instructions which have been previously loaded in the memory.

ST52x440 internal structure includes a computational block, the CONTROL UNIT (CU) / DATA PROCESSING UNIT (DPU), that allows processing of boolean functions and fuzzy algorithms.

The CU/DPU is able to manage up to 341 different Membership Functions for the fuzzy rules antecedent part. The rules consequents are "crisp" values (real numbers). The maximum number of rules that can be defined is limited by the dimensions of the implemented standard algorithm.

The Eprom is then shared between fuzzy and standard algorithms. Therefore smaller standard algorithms allow to define bigger fuzzy algorithms with more rules and viceversa.

The Control Unit (CU) read the information and the status coming from the peripherals.

The arithmetic calculus can be performed on these values by using the internal CU and the 128/256 bytes RAM, which supports all

Figure 1.2 ST52x440 Block Diagram

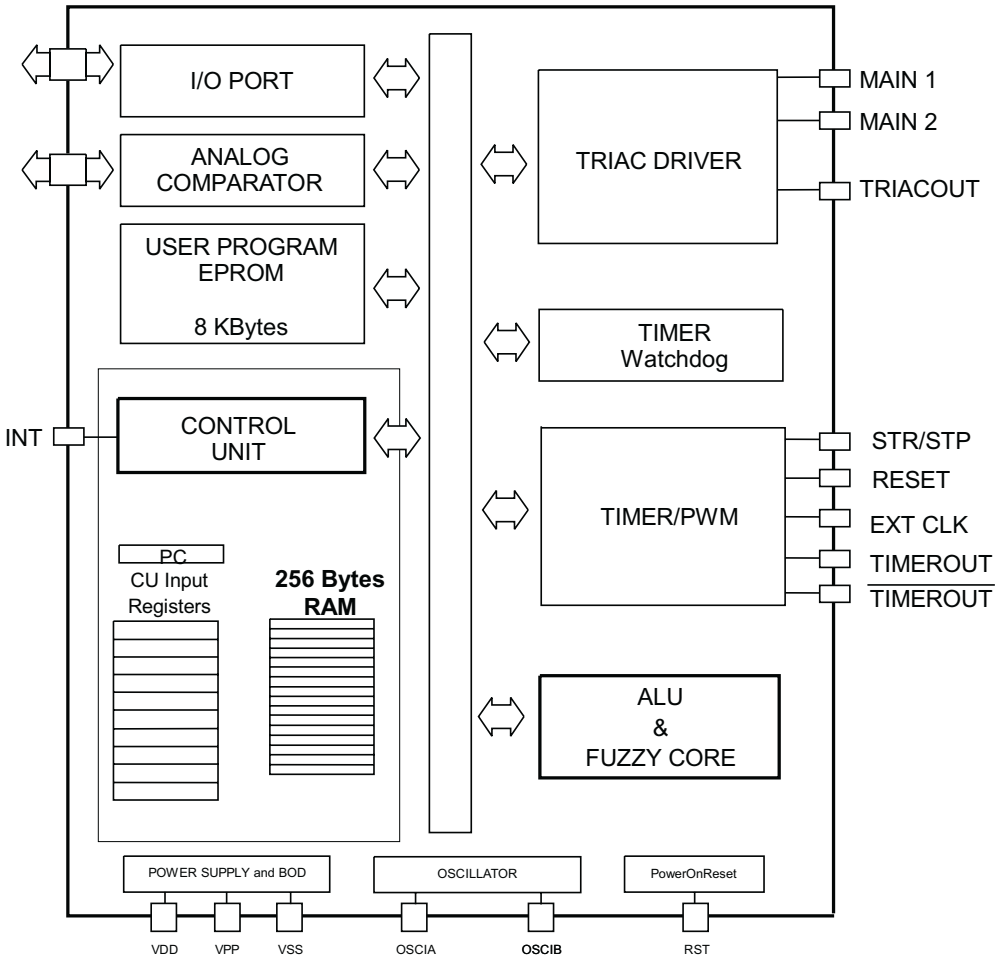


Figure 1.3. SO28 Pin Configuration

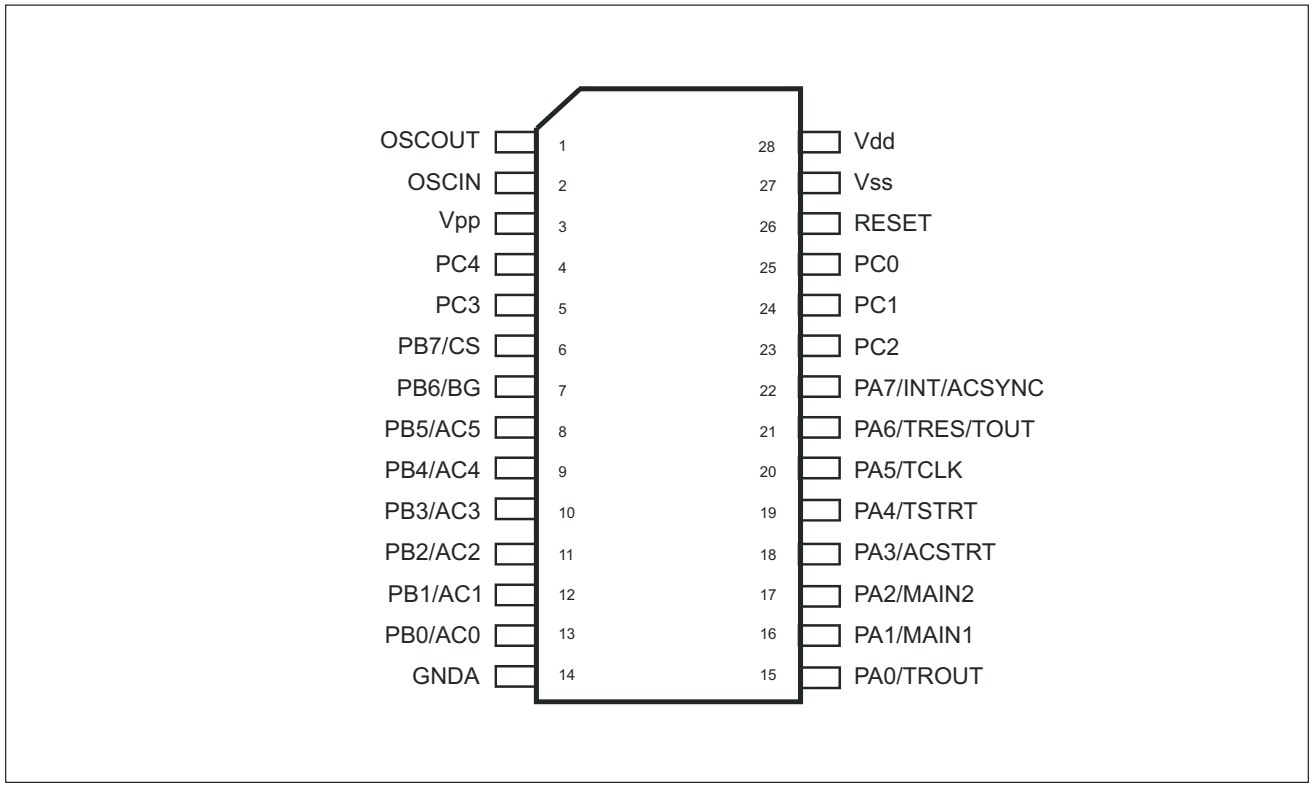


Figure 1.4 PDIP28 Pin Configuration

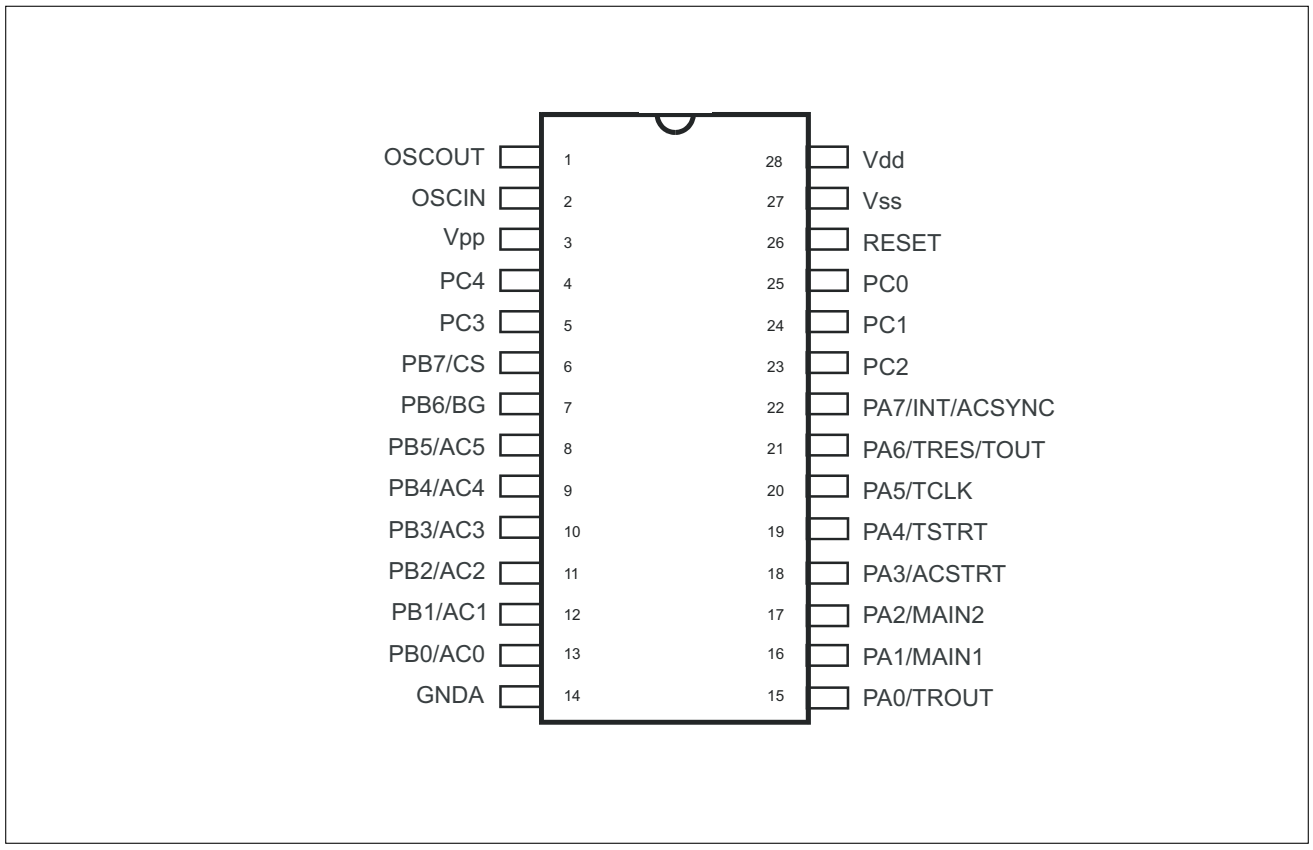


Figure 1.5. SO20 Pin Configuration

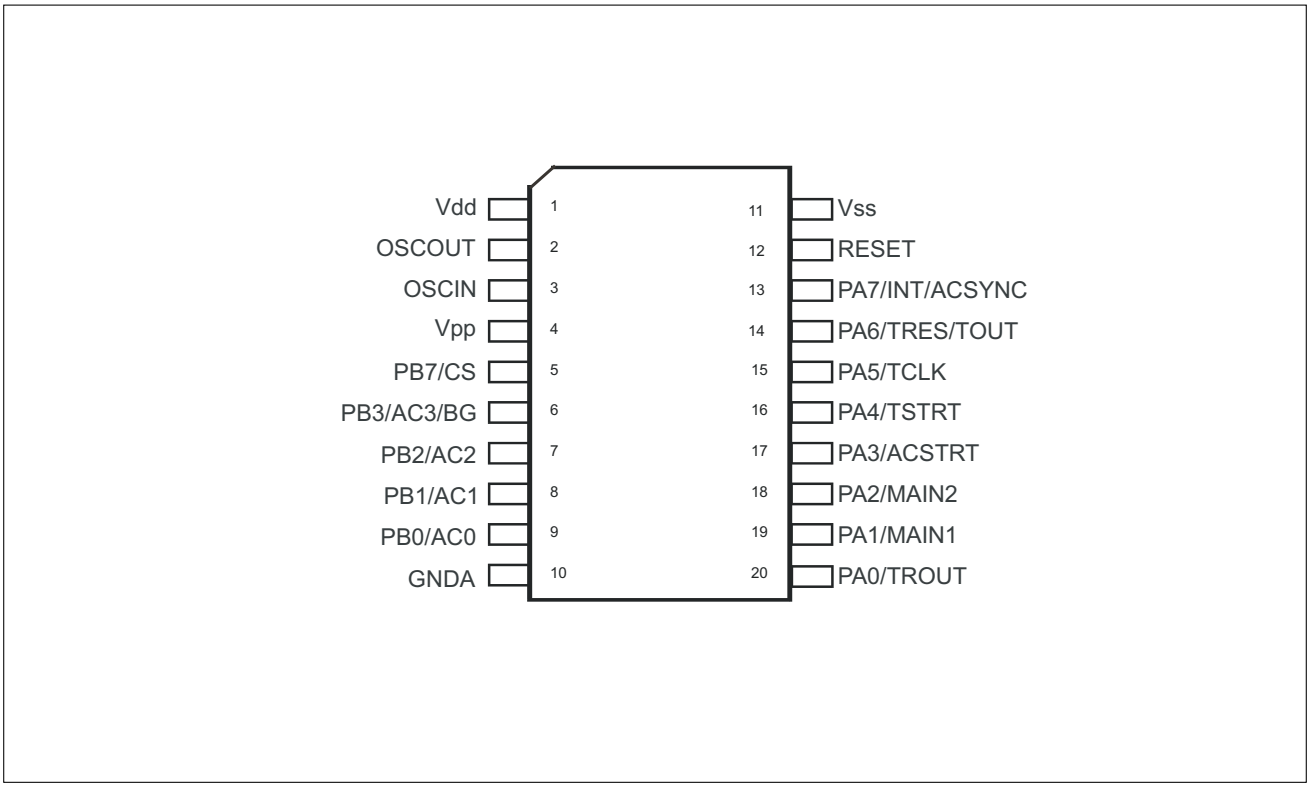


Figure 1.6 PDIP20 Pin Configuration

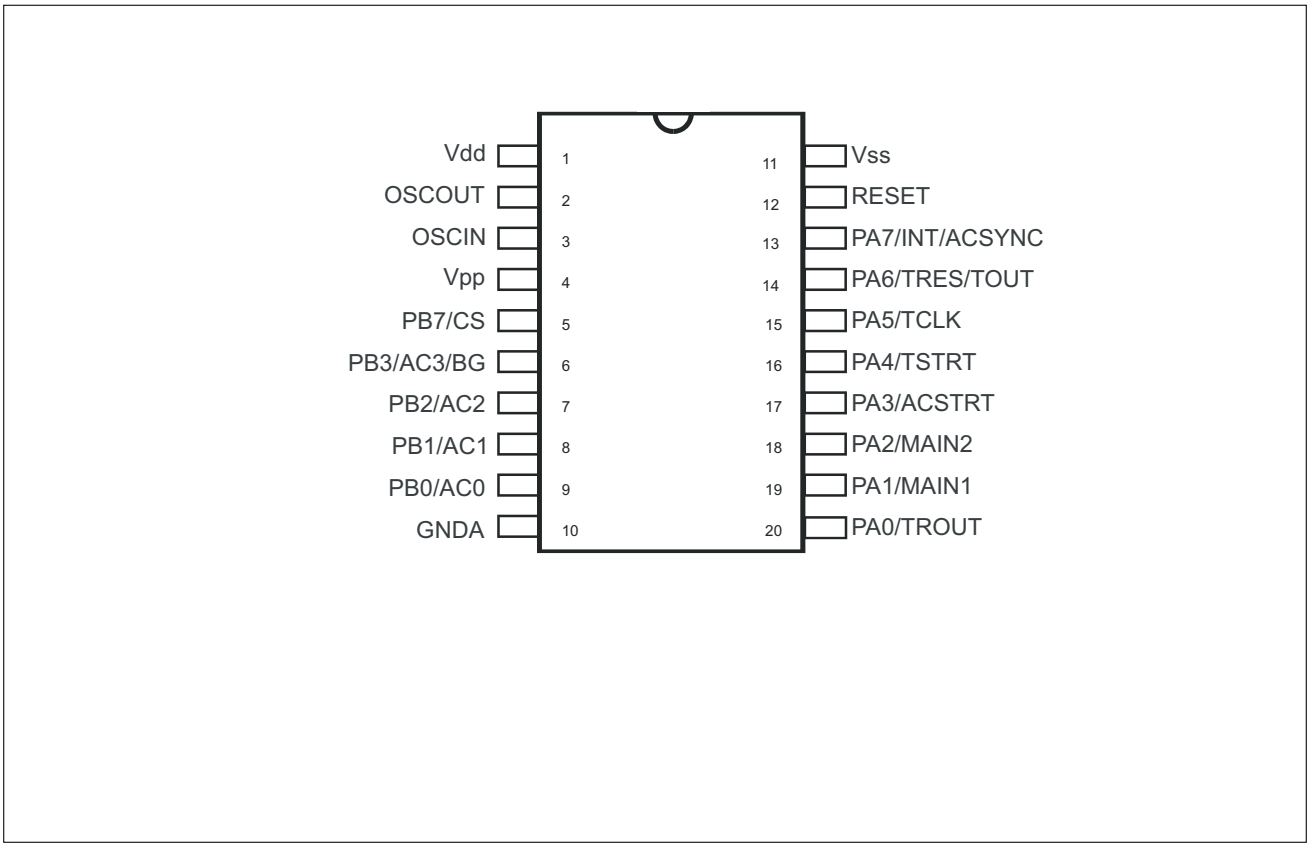


Table 1.2 - SO28 and DIP28 Pin Configuration - ST52x440G0/G1/G2/G3

PIN SO28/DIP 28	NAME	TYPE	Programming Phase	Working Phase
1	OSCOUT	O	Oscillator Output	Oscillator Output
2	OSCIN	I	Oscillator Input	Oscillator Input
3	Vpp	I	EPROM Programming Power supply (12V±5%)	EPROM V_{DD} or Vss
4	PC4	I/O		Digital I/O
5	PC3	I/O		Digital I/O
6	PB7		PHASE signal	Digital I/O / Capacitor connection
7	PB6	I/O		Digital I/O / Bandgap reference
8	PB5	I/O		Analog Comp.5
9	PB4	I/O		Analog Comp. 4
10	PB3	I/O	Configuration INCREMENT	Analog Comp. 3
11	PB2	I/O	Configuration RESET	Analog Comp. 2
12	PB1	I/O	Address INCREMENT	Analog Comp. 1
13	PB0	I/O	Address Reset	Analog Comp. 0
14	GNDA		Analog Ground	Analog Ground
15	PA0	I/O	I/O EPROM Data	Digital I/O / TRIAC_OUT
16	PA1	I/O	I/O EPROM Data	Digital I/O / MAIN1
17	PA2	I/O	I/O EPROM Data	Digital I/O/ MAIN2 / TOUT_N
18	PA3	I/O	I/O EPROM Data	Digital I/O/ ACSTRT
19	PA4	I/O	I/O EPROM Data	Digital I/O / TSTRT
20	PA5	I/O	I/O EPROM Data	Digital I/O / TCLK
21	PA6	I/O	I/O EPROM Data	Digital I/O / TRES / TOUT
22	PA7	I/O	I/O EPROM Data	Digital I/O / INT / AC_SYNC
23	PC2	I/O		Digital I/O
24	PC1	I/O		Digital I/O
25	PC0	I/O		Digital I/O
26	RESET	I	General Reset	General Reset
27	V_{SS}		Digital Ground	Digital Ground
28	V_{DD}		Digital Power Supply	Digital Power Supply

Table 1.3 ST52x440 - SO20 and DIP20 Pin Configuration - ST52x440I0/I1/I2/I3

PIN SO20/ DIP20	NAME	TYPE	Programming Phase	Working Phase
1	V_{DD}		Digital Power Supply	Digital Power Supply
2	OSCOUT	O	Oscillator Output	Oscillator Output
3	OSCIN	I	Oscillator Input	Oscillator Input
4	Vpp		EPROM Programming Power supply (12V±5%)	EPROM V_{DD} or Vss
5	PB7		PHASE signal	Digital I/O / Capacitor connection
6	PB3	I/O	Configuration INCREMENT	Analog Comp. 3/ BG
7	PB2	I/O	Configuration RESET	Analog Comp. 2
8	PB1	I/O	Address INCREMENT	Analog Comp. 1
9	PB0	I/O	Address Reset	Analog Comp. 0
10	GNDA		Analog Ground	Analog Ground
11	PA0	I/O	I/O EPROM Data	Digital I/O / TRIAC_OUT
12	PA1	I/O	I/O EPROM Data	Digital I/O / MAIN1
13	PA2	I/O	I/O EPROM Data	Digital I/O/ MAIN1 / TOUT_N
14	PA3	I/O	I/O EPROM Data	Digital I/O/ ACSTRT
15	PA4	I/O	I/O EPROM Data	Digital I/O / TSTRT
16	PA5	I/O	I/O EPROM Data	Digital I/O / TCLK
17	PA6	I/O	I/O EPROM Data	Digital I/O / TRES / TOUT
18	PA7	I/O	I/O EPROM Data	Digital I/O / INT / AC_SYNC
19	RESET	I	General Reset	General Reset
20	V_{SS}		Digital Ground	Digital Ground

1.3 PIN DESCRIPTION

ST52x440 pins support Alternate Functions and are configurable by means of configuration registers.

V_{DD}, **V_{SS}**, **GNDA, V_{PP}.** In order to avoid noise disturbances, the power supply of the digital part is kept separated from the power supply of the analog part.

V_{DD}. Main Power Supply Voltage (5V ± 10%).

V_{SS}. Digital circuit Ground.

GNDA. Analog circuit ground of the Analog Comparator. *Must be tied to V_{SS}.*

V_{PP}. Main Power Supply for the internal EPROM. (12.5V ± 5%, in programming phase) and MODE selector.During the Programming phase (programming) V_{PP} must be set at 12V. In the Working phase V_{PP} must be equal to **V_{SS}**.

OSCin and **OSCout.** These pins are internally connected to the on-chip oscillator circuit. A quartz crystal or a ceramic resonator can be connected between these two pins in order to allow the correct operations of ST52x440 with various stability/cost trade-offs. An external clock signal can be applied to OSCin: in this case OSCout must be grounded.

RESET. This signal is used to restart ST52x440 at the beginning of the program. It also allows to select the programming mode for the EPROM.

AC0-AC5, CS. In order to implement high resolution A/D conversion an Analog Peripheral is available. This peripheral contains an analog multiplexer (4 channels for ST52440Ix, 6 channels for ST52440Gx), an analog Comparator, a 16 bit Timer with output capture and 4 bit Prescaler, one output for a current generator (CS) allowing to charge an external capacitor obtaining a voltage ramp for the A/D conversion.

PA0-PA7, PB0-PB7,PC0-PC4. These lines are organized as I/O ports. Each pin can be configured as an input or an output. During the Programming phase the ports are used for the EPROM data read/write operations.

T0RES, **T0CLK**, **T0STRT**. These pins are related to the internal Programmable Timer. This Timer can be reset externally by using T0RES. In Working Mode, T0RES resets the address counter of the Timer. T0RES is active at low level.

The Timer Clock can be the internal clock or can be supplied externally by using pin T0CLK.

An external Start/Stop signal can be used to control the Timer through the pin T0STRT.

T0OUT. The TIMER outputs are available on these pins.

BG . A Bandgap Reference value of 2.5V is available on this pin. It can be used for analog signal conditioning.

MAIN1, MAIN2, TRIACOUT. ST52x440 is able to drive a TRIAC in two different modes: Burst mode and Phase Angle Partialization control mode.

The Burst mode is used mainly for thermal regulation, the Phase Angle Partialization mode for inductive load powering, typically motor controls .

MAIN1 and MAIN2 signals are used to detect the zero crossing of the mains voltage.

ST52x440 Triac driver peri[pheral allows to directly fire an external Triac device without using additional external interfaces. Pulses to drive the TRIAC device are provided by the microcontroller on TRIACOUT pin.

When the TRIAC peripheral is configured to work in PWM mode, a PWM signal is available on TRIACOUT pin.

2 INTERNAL ARCHITECTURE

ST52x440 is made up by the following blocks and peripherals:

- Control Unit (CU) and Data Processing Unit (DPU)
- ALU / Fuzzy Core
- EPROM
- 256 Byte RAM
- Clock Oscillator
- Analog Multiplexer and Analog Comparator
- 1 PWM / Timer
- 1 Triac/PWM Driver
- Digital I/O port

ST52x440 Operating Modes

ST52x440 works in two modes, Programming and Working Modes, depending on the control signals level RESET and Vpp.

The Operating modes are selected by setting the control signal level as specified in the Control Signals Setting table 1.2.

2.1 CONTROL UNIT and DATA PROCESSING UNIT

The Control Unit (CU) formally includes five main blocks. Each block decodes a set of instructions then generating the appropriate control signals.

The main parts of the CU are shown in the figure 2.1.

The five different parts of the CU manage the Loading, Logic/Arithmetic, Jump, Control and Fuzzy instructions set.

The block called “Collector” manages the signals coming from the different parts of the CU then defines the signals for the Data Processing Unit (DPU) and for the different peripherals of the microcontroller.

The block called “Arbiter” manages the different parts of the CU in order to have only one part of the system activated during the working mode.

The CU structure is very flexible. It was designed with the aim to easily adapt the core of the microcontroller to the market needs. New instructions set or new peripherals can be easily included without changing the structure of the microcontroller then maintaining the code compatibility.

The CU reads the stored instructions on the EPROM (Fetch) and decodifies them.The Arbiter according to the instructions type, activates one of the main blocks of the CU. Then all the control signals for the DPU are generated.

A set of 46 different arithmetic, fuzzy and logic instructions is available. Each instruction requires

Figure 2.1 CU Block Diagram

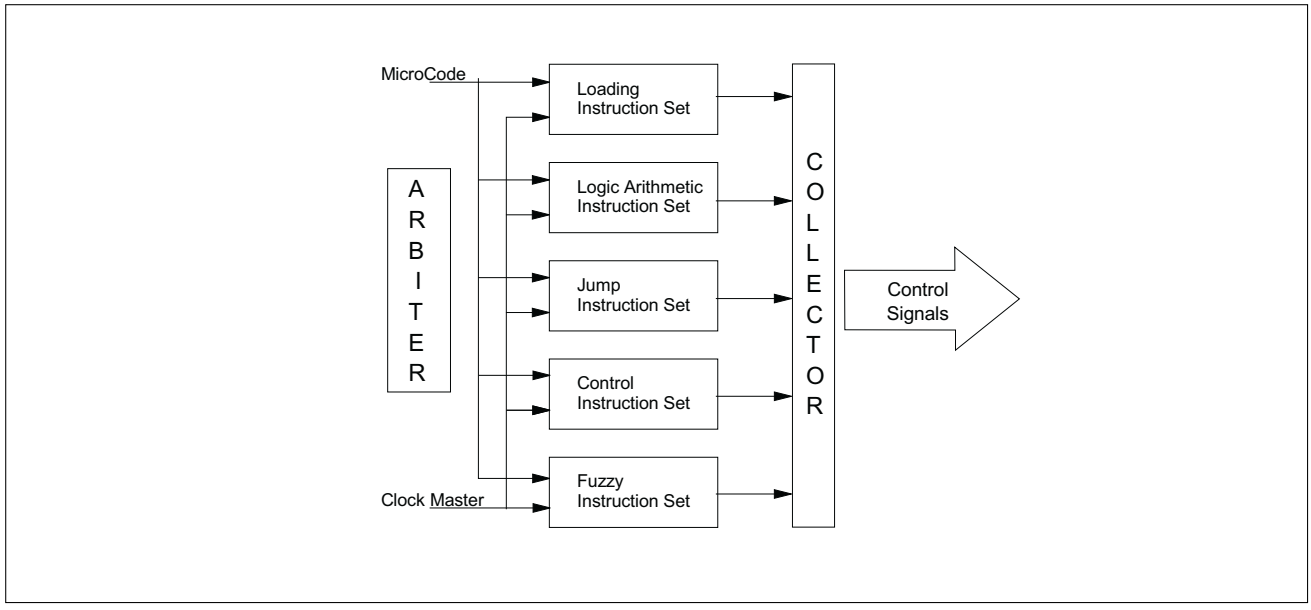


Figure 2.2 Data Processing Unit (DPU)

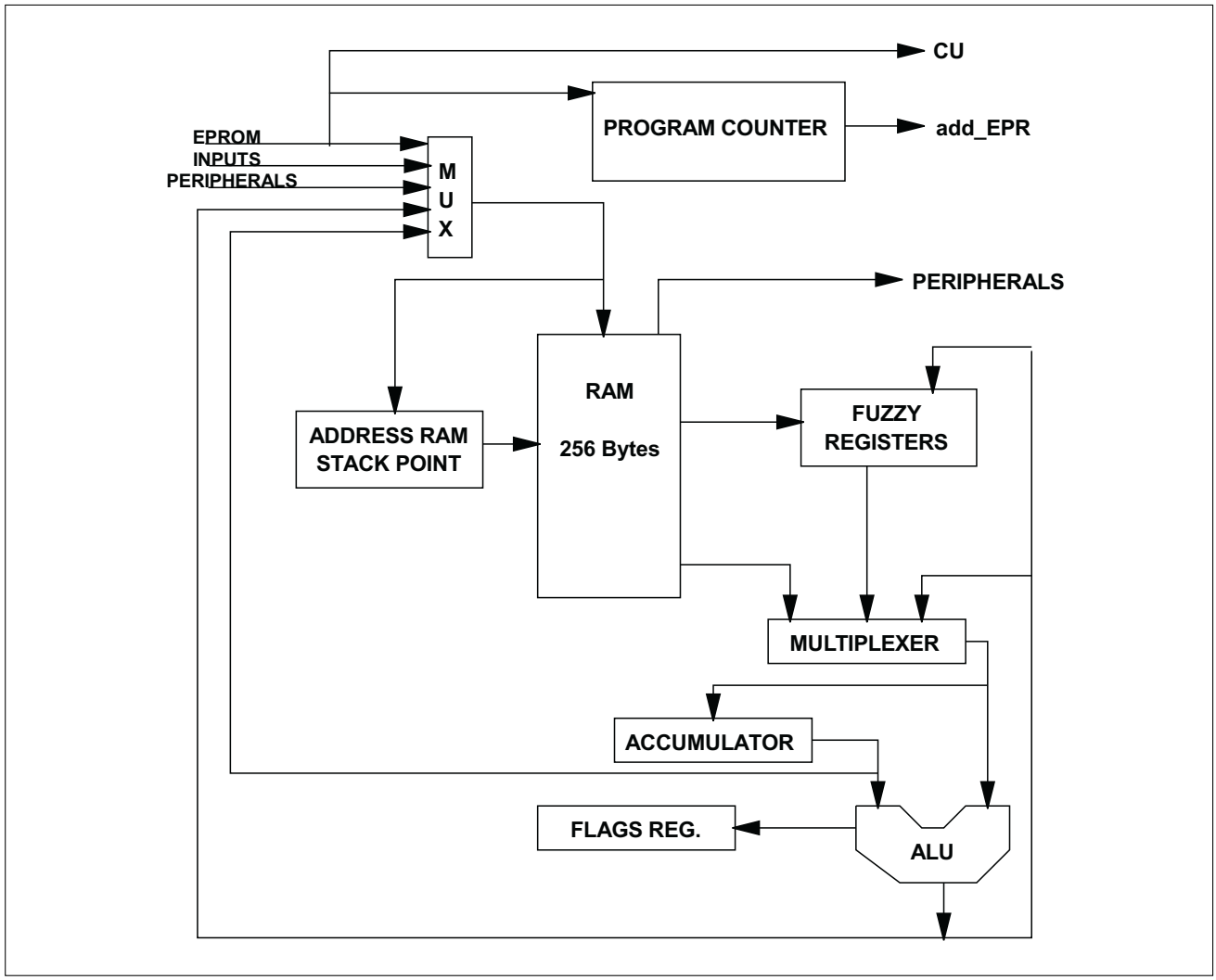


Figure 2.3 CU/DPU Block diagram

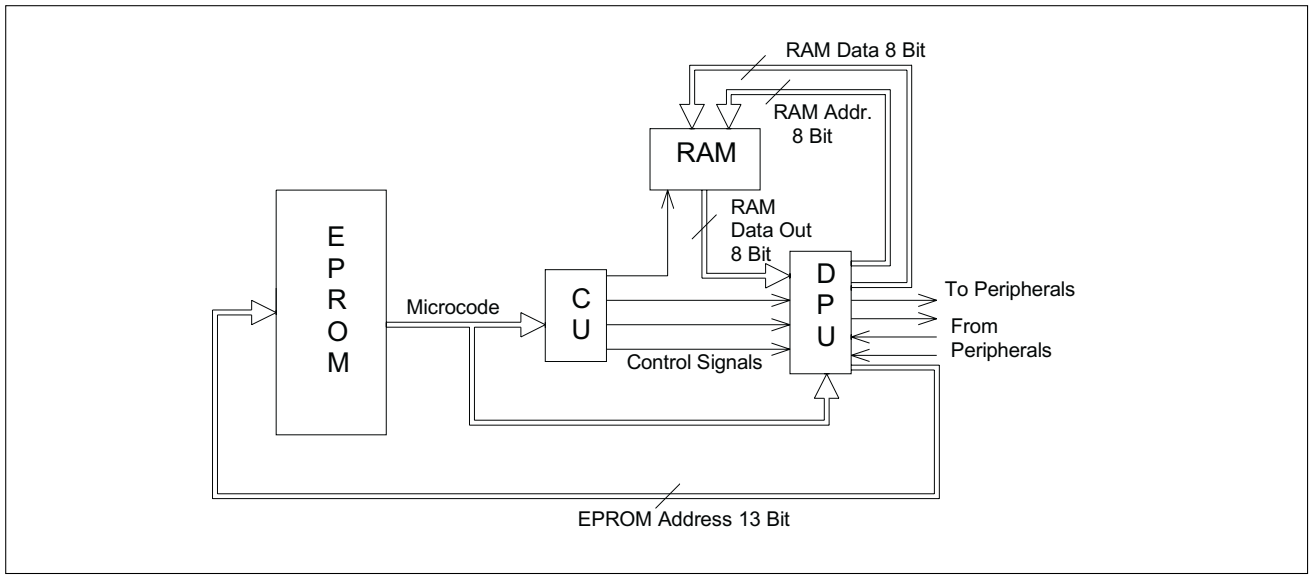
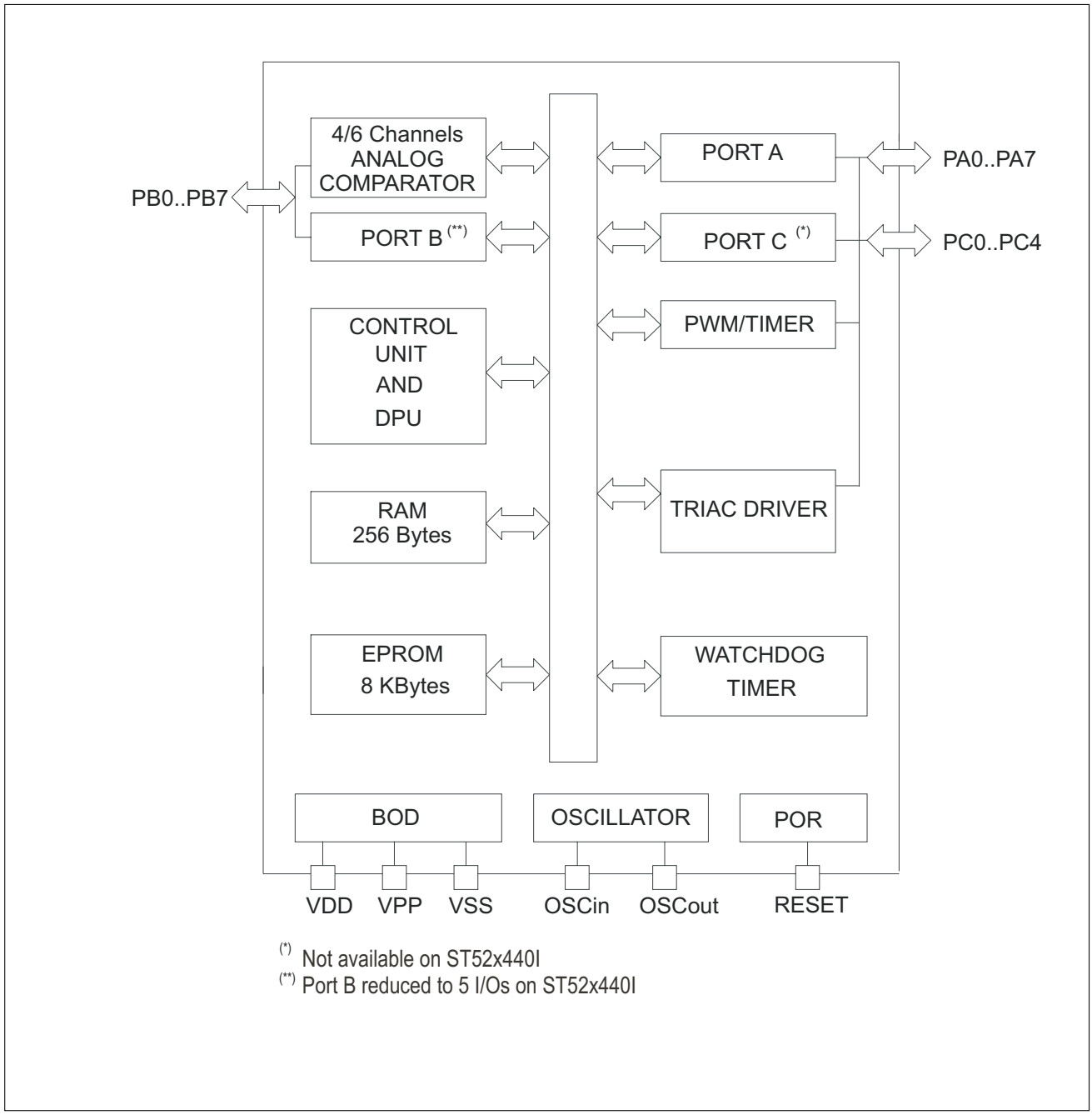


Figure 2.4 ST52x440 Peripherals Block Diagram



(*) Not available on ST52x440I
(**) Port B reduced to 5 I/Os on ST52x440I

from 6 (fuzzy instructions) to 26 (DIVISION) clock pulses to be performed.

The DPU receives, stores and sends the instructions coming from the EPROM, the RAM or from the peripherals in order to execute them.

2.1.1 Program Counter

The Program Counter (PC) is a 13-bit register that contains the address of the next memory location to be processed by the core. This memory location may be an opcode, an operand or an address of an operand.

The 13-bit length allows the direct addressing mode of 8192 bytes in the program space.

After having read the current instruction address, the PC value is incremented. The result of this operation is shifted back into the PC.

The PC can be changed in the following ways:

- JP (Jump) instruction PC = Jump Address
- Interrupt PC = Interrupt Vector
- RETI instruction PC = Pop (stack)
- Reset PC = Reset Vector
- Normal Instruction PC = PC + 1

2.1.2 Flags

The ST52x440 core includes two pair of different flags that correspond to 2 different modes: normal mode and interrupt or call mode. Each pair consist of a CARRY flag (C), a ZERO flag (Z) and SIGN flag (S). One pair (CN, ZN, SN) is used during normal operation and one is used during the interrupt mode (CI, ZI, SI). Formally the user has to

manage only a set of flag: C, Z and S.

The ST52x440 core uses the pair of flags that correspond to the actual mode: as soon as an interrupt or a call is generated, the ST52x440 core uses the interrupt flags instead of the normal flags. When the RETI instruction is executed, the normal flags are restored if the MCU was in the normal mode before the interrupt. It should be observed that each flag set can only be addressed in its own routine.

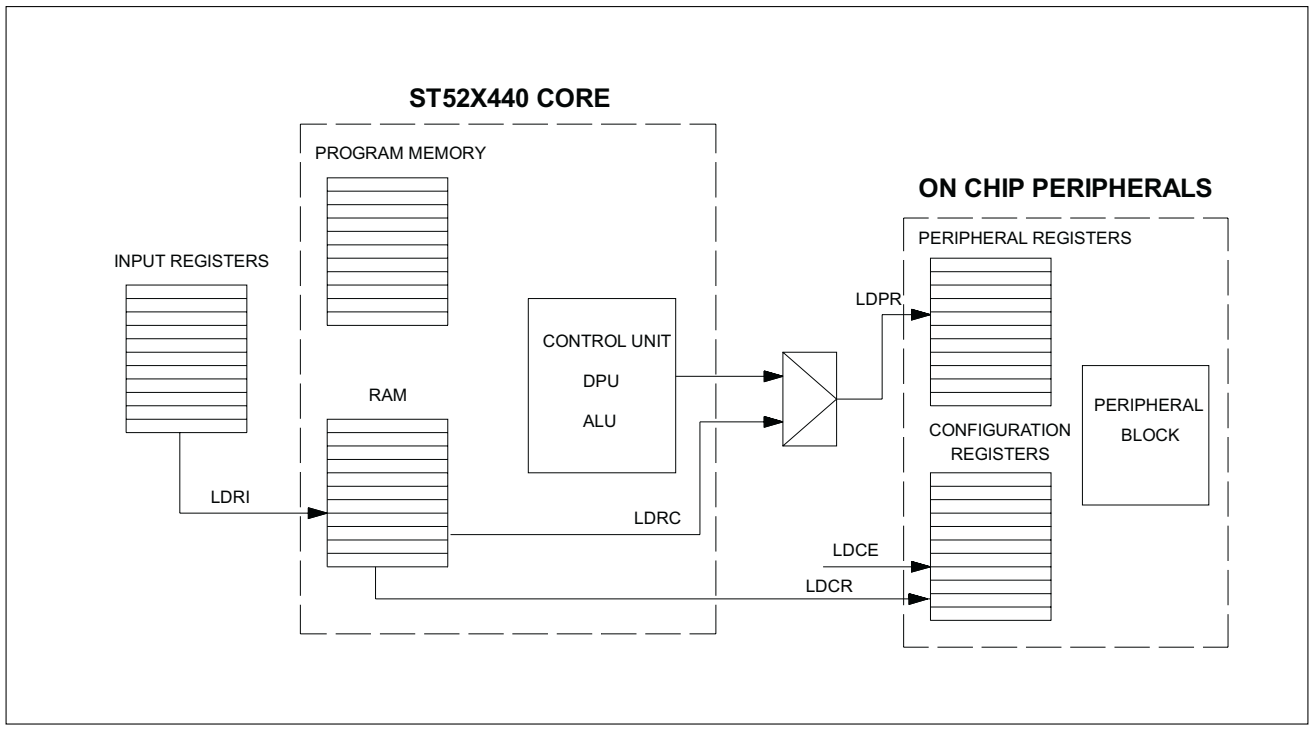
The flags are not cleared during the context switching and remain in the state they were at the exit of the last routine switching.

The Carry flag is set when an overflow occurs during arithmetic operations, otherwise it is cleared.

The Sign flag is set when an underflow occurs during arithmetic operations, otherwise it is cleared.

The switching between the two sets of flags is automatically performed when an interrupt, a CALL, a RET or a RETI instruction occurs.

Figure 2.5 Address Spaces Description



2.2 ADDRESS SPACES

ST52x440 has four separate address spaces:

- RAM: 128 or 256 Bytes
- Input Registers: 20 8-bit registers
- Output Registers 10 8-bit registers
- Configuration Registers: 20 8-bit registers
- Program memory: up to 8K Bytes

The Program memory will be described in further details in the MEMORY section

2.2.1 RAM and STACK

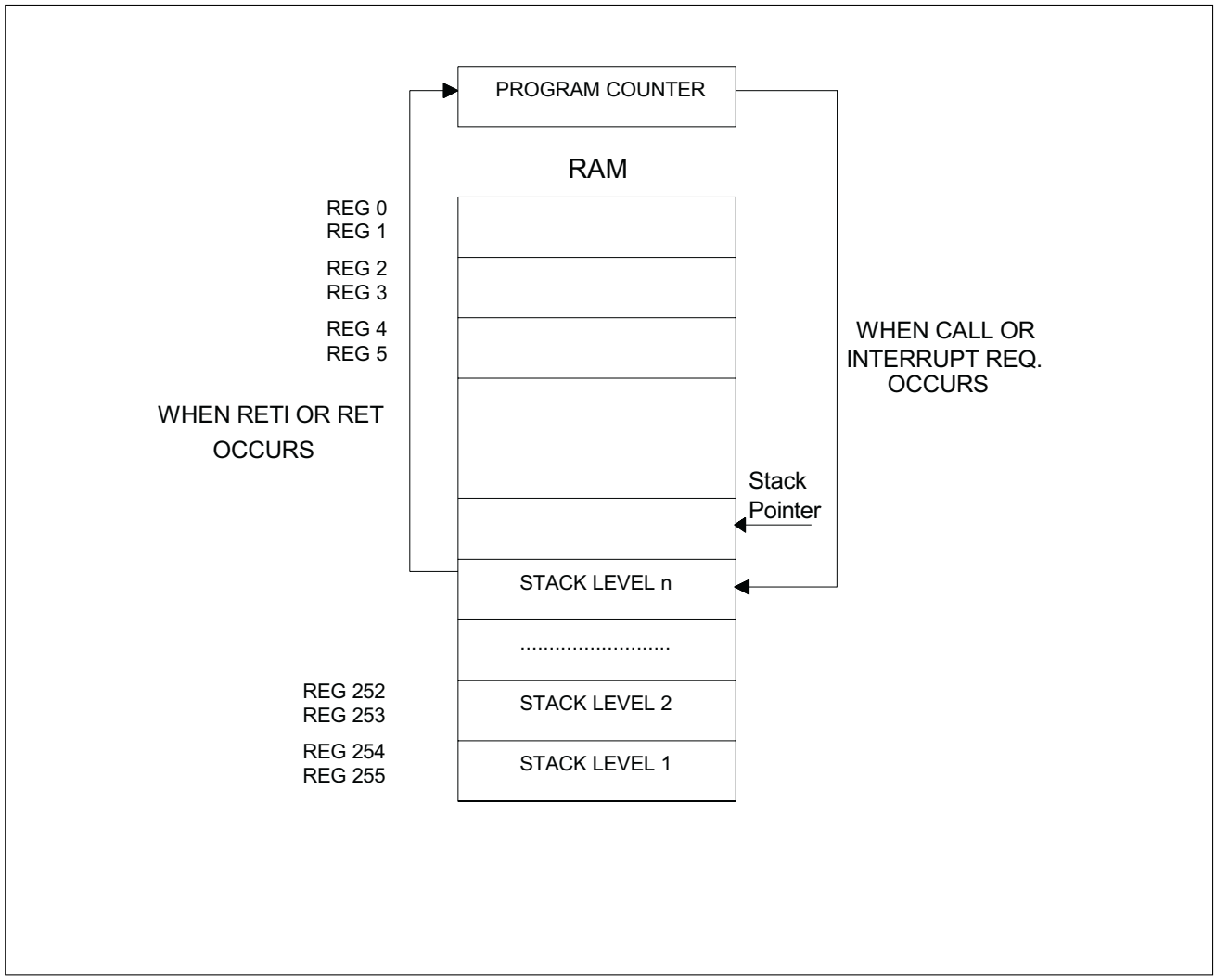
The RAM consists of 128 (for ST52x440G0/G1/I0/I1) or 256 (for ST52x440G2/G3/I2/I3) general purpose 8-bit registers.

All the registers in the RAM can be specified by using a decimal address, e.g. 0 identify the first register of the RAM.

To read or write the RAM registers the LOAD instructions must be used (see table 2.6)

When the instructions like Interrupt request or CALL are executed, a STACK is used to push the PC. The STACK is push directly in the RAM. For each level of stack 2 byte of the RAM are used. The values of this stack are stored from the last RAM register (address 255). The maximum level of stack must be less than 128. When a subroutine

Figure 2.6. Stack Operation



call or interrupt request occurs, the contents of each level is shifted into the next level while the content of the PC is shifted into the first level. When a subroutine or interrupt return occurs (RET or RETI instructions), the first level register is shifted back into the PC and the value of each level is popped back into the previous level. These operating modes are described in the Figure 2.6.

2.2.2 Input Registers Bench

The Input Registers (IR) bench consists of 20 8-bit registers containing data or status of the peripherals.

All the registers can be specified by using a decimal address, e.g. 0 identifies the first register of the IR.

The assembler instruction:

LDRI RAM_Reg IR_i

loads the value of the i-th IR in the RAM location identified by the address RAM_Reg.

The first input register is dedicated to store the value of the stack pointer. The next 12 registers of the IR are dedicated to the 6 (for ST52X440G) or the 4 converted values (for ST52X440I) in case of converted values coming from the Analog Comparator. Each of these values are stored on two bytes because of the resolution of the A/D conversion process. The last 7 registers contain data from the I/O ports and PWM/Timers. The following table summaries the IR address and the relative peripheral. For simplicity reasons a mnemonic name is assigned to the registers. The

Table 2.1 Input Registers

IR MNEMONIC NAME	PERIPHERAL REGISTER	ADDRESS
STACK_POINTER	STACK POINTER	0
AC_CHAN 0H	Analog Comparator CHANNEL 0 High Byte	1
AC_CHAN 0L	Analog Comparator CHANNEL 0 Low Byte	2
AC_CHAN 1H	Analog Comparator CHANNEL 1 High Byte	3
AC_CHAN 1L	Analog Comparator CHANNEL 1 Low Byte	4
AC_CHAN 2H	Analog Comparator CHANNEL 2 High Byte	5
AC_CHAN 2L	Analog Comparator CHANNEL 2 Low Byte	6
AC_CHAN 3H	Analog Comparator CHANNEL 3 High Byte	7
AC_CHAN 3L	Analog Comparator CHANNEL 3 Low Byte	8
AC_CHAN 4H (*)	Analog Comparator CHANNEL 4 High Byte	9
AC_CHAN 4L (*)	Analog C omparator CHANNEL 4 Low Byte	10
AC_CHAN 5H (*)	Analog Comparator CHANNEL 5 HighByte	11
AC_CHAN 5L (*)	Analog Comparator CHANNEL 5 Low Byte	12
AC_STATUS	Analog Comparator Status Register	13
PORT_A	PORT A INPUT REGISTER	14
PORT_B	PORT B INPUT REGISTER	15
PORT_C	PORT C INPUT REGISTER	16
TRIAC_COUNT	TRIAC Driver COUNTER Value	17
PWM_COUNT	PWM/TIMER COUNTERValue	18
PWM_STATUS	TIMER STATUS REGISTER	19

(*) Not used on ST52x440I versions

same name is used in FUZZYSTUDIO™4.x development tools.

2.2.3 Configuration Registers

The ST52x440 Configuration Registers allow to configure all the blocks of the fuzzy microcontroller. Table 2.3 describes the functions and the related peripherals of each Configuration Register. By using the load instructions it is possible to set the Configuration Registers by using values stored in the Program Memory (EPROM) or in the RAM.

Use and meaning of each register will be described in further details in the corresponding section.

2.2.4 Output Registers

The Output Registers (OR) consist of 10 registers containing data for the microcontroller peripherals including I/O Ports.

All registers can be specified by using a decimal address, e.g. 1 identifies the second OR.

Table 2.2. Configuration Registers description

CONFIGURATION REGISTER	PERIPHERAL	DESCRIPTION
REG_CONF 0	INTERRUPT MASK	Interrupts mask setting, Polarity, Brown Out
REG_CONF 1	ANALOG COMPARATOR	AC Configuration Register 1
REG_CONF 2	WATCHDOG TIMER	Watchdog Timer Configuration
REG_CONF 3	ANALOG COMPARATOR	AC Configuration Register 2
REG_CONF 4	PORT A	PORT A digital pin I/O direction
REG_CONF 5	PWM/TIMER	PWM/TIMER Working mode Configuration
REG_CONF 6	PWM/TIMER	PWM/TIMER Prescaler configuration and output waveform selection.
REG_CONF 7	PWM/TIMER	PWM/TIMER Prescaler settings
REG_CONF 8	PWM/TRIAC	PWM/TRIAC Prescaler settings
REG_CONF 9	PWM/TRIAC	PWM/TRIAC Prescaler configuration and output waveform selection.
REG_CONF 10	PWM/TRIAC	PWM/TRIAC Working mode Configuration
REG_CONF 11	PORT C	PORT C digital pin I/O direction and Bandgap Reference options
REG_CONF 12	PORT A	PORT A Alternate function settings
REG_CONF 13	PORT B	PORT B digital pin I/O direction
REG_CONF 14	PORT B	PORT B settings for digital or analog pin
REG_CONF 15	ANALOG COMPARATOR	Analog Comparator Prescaler settings
REG_CONF 16	ANALOG COMPARATOR	Analog Comparator in A/D working mode
REG_CONF 17	INTERRUPT	Interrupt priorities
REG_CONF 18	INTERRUPT	Interrupt priorities
REG_CONF 19	TRIAC	TRIAC Pulses Width Configuration
REG_CONF 20	TRIAC	TRIAC Pulses Width Configuration

By using the LOAD instruction it is possible to set the Output Registers (OR) with values stored in the Program Memory (LDPE) or in the RAM (LDPR)

The assembler instruction:

LDPR OR_i RAM_Reg.

loads the value of the RAM location identified by the address RAM_Reg in the i-th Output Registers. Table 2.4 describes the OR: for simplicity reasons a mnemonic name is assigned to each Output Registers. The same name is used in FUZZYSTUDIO™4.x development tool.

Use and meaning of each register will be described in further details in the corresponding section.

Table 2.3 Output Registers

OR MNEMONIC NAME	PERIPHERAL REGISTER	ADDRESS
PORT_A	PORT A OR	0
PORT_B	PORT B OR	1
PORT_C	PORT C OR	2
TRIAC_COUNT	TRIAC Driver COUNTERValue	3
PWM_RELOAD (*)	PWM/TIMER RELOAD Value	4
	not used	5
	not used	6
	not used	7
	not used	8
PWM_COUNT	TIMER/PWM COUNTERValue	9

(*) If Peripheral has been programmed in PWM Mode

2.4 FUZZY CAPABILITIES

ST52x440 Fuzzy main features are:

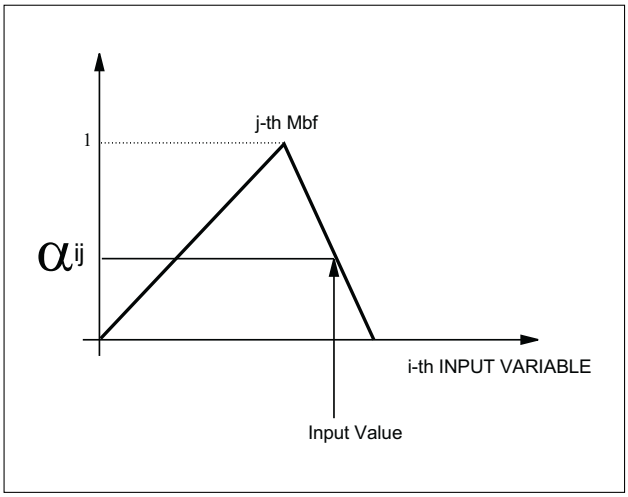
- Up to 8 Inputs with 8-bit resolution;
- 1 Kbyte of Program Memory (EPROM) available to store more than 300 to Membership Functions (Mbfs) for each Input;
- Up to 256 Outputs with 8-bit resolution;
- Possibility to process fuzzy rules with an UNLIMITED number of antecedents
- UNLIMITED number of Rules and Fuzzy Blocks.

The limits on the number of Fuzzy Rules and fuzzy Blocks are only related to the program memory size.

2.4.1 Fuzzy Inference

The block diagram shown in figure 2.8 describes the different steps performed during a fuzzy algorithm. ST52x440 core allows to implement a MAMDANI type fuzzy inference with crisp consequents. The input for the fuzzy inference are stored in 8 dedicated Fuzzy input registers. The instruction LDFR is used to set the input fuzzy registers with the values stored in the RAM. The result of a fuzzy inference is directly stored in a location of the RAM.

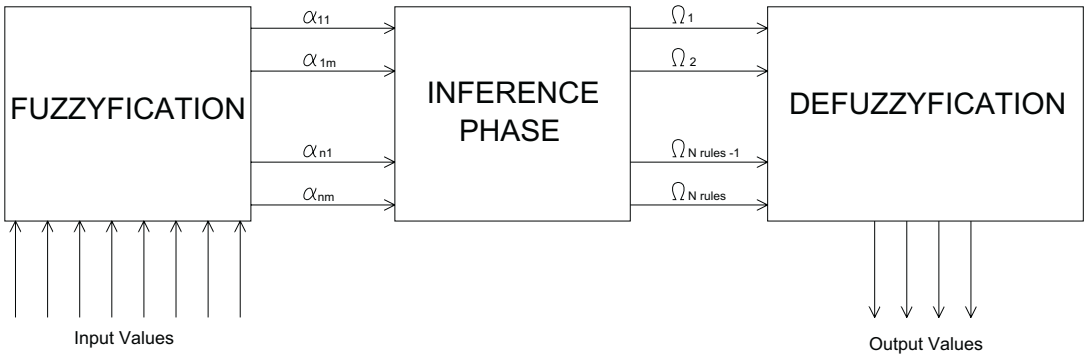
Figure 2.7. Alpha Weigth calculation



2.4.2 Fuzzyfication Phase

In this phase the intersection between the input values and the related Mbfs is performed to calculate the ***alpha weight*** (Fig. 2.7).

Figure 2.8. Fuzzy Inference



8 Fuzzy input registers are available for the fuzzy inferences.

After loading the input values by using the LDFR assembler instruction, the user can start the fuzzy inference by using the assembler instruction FUZZY.

During the **fuzzyfication**: the input data are transformed in activation level (alpha weight) of the Mbfs.

2.4.3 Inference Phase

It manages the alpha weights obtained during the fuzzyfication phase to compute the truth value ω of each rule.

Each rule antecedent part can link together several conditions by linguistic connectives AND/OR, NOT operator and brackets.

Therefore the calculus of the activation value of each rule is performed as the maximum (in case of OR operators) and/or the minimum (in case of AND operators) of the alpha values of each variable present in the antecedent part of the rule, according to the logical connectives of the fuzzy rules is performed on .

The rule truth value ω and the related output singleton of all rules are then used in the Defuzzyfication phase to complete the inference calculation.

Figure 2.9. Fuzzyfication

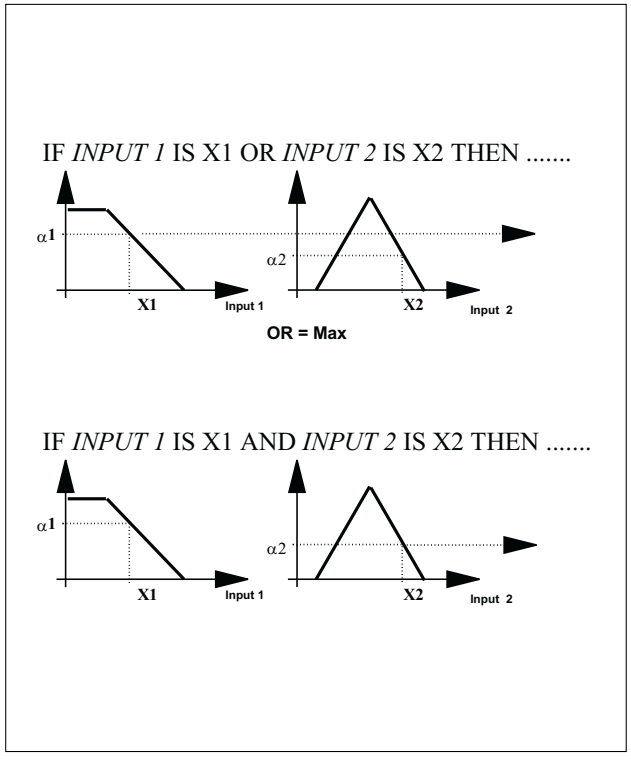
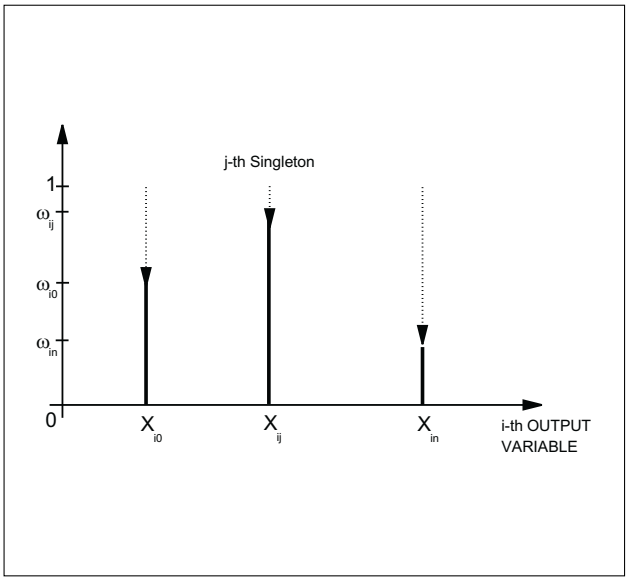


Figure 2.10 Output Membership Functions.



2.4.4 Defuzzyfication

In this phase the output crisp values are determined implementing the consequent part of the rules.

Each consequent Singleton X_i is multiplied by its weight values ω_i, calculated by the Fuzzy Inference Unit in order to compute the upper part of the defuzzification.

Each output value is deduced from the consequent crisp values (X_i) by using the defuzzification formula:

$$Y_i = \frac{\sum_{j}^{N} X_{ij}\ \omega_{ij}}{\sum_{j}^{N} \omega_{ij}}$$

where:

i = 0,1 identifies the current output variable

N = number of the active rules on the current output

ω_{ij} =weight of the j-th singleton

Xij = abscissa of the j-th singleton

The fuzzy outputs are stored in the RAM location i-th specified in the assembler instruction OUT i.

2.4.5 Input Membership Function

ST52x440 allows to manage triangular Mbfs. In order to define a Mbf it is necessary to store three different data on the program memory:

- the vertex of the bf: **V**;
- the length of the left semi-base: **LVD**;
- the length of the riMght semi-base: **RVD**;

In order to reduce the size of the memory area and the computational effort the vertical dimension of the vertex is fixed to 15 (4 bits)

By using the previous memorization method it is possible to store different kinds of triangular Membership Functions. The figure 2.12 shows a typical example of Mbfs that can be defined in ST52x440.

Each Mbf is then defined storing 3 bytes in the first 1 Kbyte of the memory program.

The Mbf is memorized by using the following instruction:

MBF n_mbf lvd v rvd

where n_mbf identifies the Mbf, lvd, v, rvd are the parameters describing the Mbf's shape.

2.4.6 Output Singleton

ST52x440 uses for the output variables a particular kind of membership function called Singleton. A Singleton has not a shape, like a traditional Mbf, and it is characterized by a single point identified by the couple (X, ω), where the ω is calculated by the Inference Unit as described before.

Often, a Singleton is simply identified with its Crisp Value X.

2.4.7 Fuzzy Rules

The rules can have the following structures:

if A op B op C.....then Z

if (A op B) op (C op D op E...) .....then Z

where op is one of the possible linguistic operators (AND/OR)

In the first case the rule operators are managed sequentially; in the second one, the priority of the operator is fixed by the brackets.

Each rule is codified by using an instruction set, the inference time for a rule with 4 antecedents and 1 consequent is about 3 microseconds.

The assembler Instruction Set allowing to manage the fuzzy instructions is reported in the following table

Example 1:

IF $Input_1$ IS NOT Mbf_1 AND $Input_4$ is Mbf_{12} OR $Input_3$ IS Mbf_8 THEN $Crisp_1$

is codified by the following instructions:

LDN 1 1	calculates the NOT α value of $Input_1$ with Mbf_1 and stores the result in internal registers
LDP 4 12	fixes the α value of $Input_4$ with M_{12} and stores the result in internal registers
FZAND	adds the NOT α and α values obtained with the operations LDN1 1 and LDP 4 12
LDK	stores the result of the operation FZAND in internal registers

Figure 2.11. Mbfs Parameters

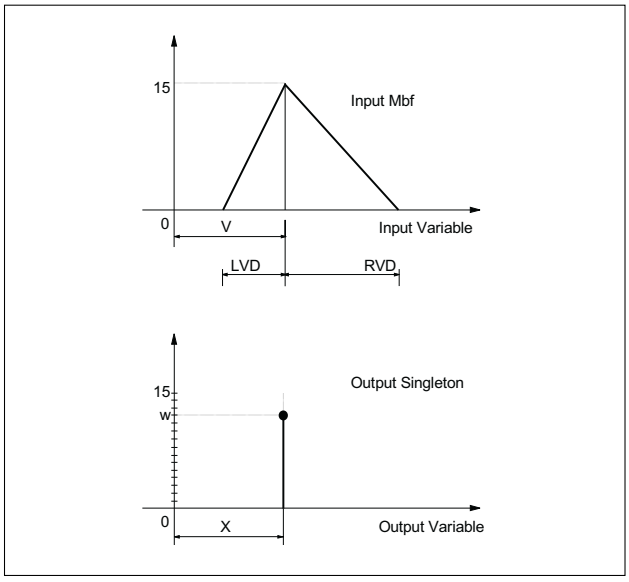
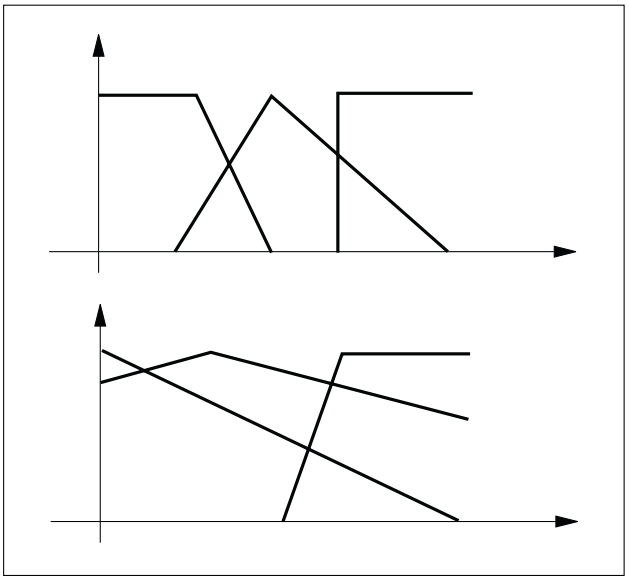


Figure 2.12. Example of valid Mbfs



LDP 3 8 fixes the α value of $Input_3$ with Mbf_8 and stores the result in internal registers

FZOR implements the operation OR between the results obtained with the operations LDK and LDP

CON crisp$_1$ multiplies the result of the last Ω operation with the crisp value $Crisp_1$

Example 2, the priority of the operator is fixed by the brackets:

IF ($Input_3$ IS Mbf_1 AND $Input_4$ IS NOT Mbf_{15}) OR ($Input_1$ IS Mbf_6 OR $Input_6$IS NOT Mbf_{14}) THEN $Crisp_2$

LDP 3 1

Table 2.4. Fuzzy Instructions Set

Instruction	Description
MBFn_mbf lvd v rvd	Stores the Mbf n_mbf with the shape identified by the parameters lvd, v and rvd.
LDP n m	Fixes the alpha value of the input n with the Mbf m and stores it in internal registers.
LDN n m	Calculates the negated alpha value of the input n with the Mbf m and store the result in internal registers.
FZAND	Implements the fuzzy operation AND between the last two values stored in internal registers.
FZOR	Implements the fuzzy operation OR between the last two values stored in internal registers.
LDK	Stores the result of the last fuzzy operation executed in internal registers.
SKM	Stores the result of the last fuzzy operation executed in internal registers.
LDM	Copies the value of the register M in the data stack.
CON crisp	Multiplies the crisp value with the last ω weight.
OUT n_out	Performs the defuzzification and store the fuzzy output in the RAM n_out location.
FUZZY	Starts the fuzzy algorithm.

	fixes the α value of $Input_3$ with Mbf_1 and stores the result in internal registers
LDN 4 15	calculates the NOT α value of $Input_4$ with Mbf_{15} and stores the result in internal registers
FZAND	adds NOT α and α values obtained with the operations LDP 3 1 and LDN 4 15
SKM	stores the result of the operation FZAND in internal registers
LDP 1 6	fixes the α value of $Input_1$ with Mbf_6 and stores the result in internal registers
LDN 2 14	calculates the NOT α value of $Input_6$ with Mbf_{14} and stores the result in internal registers
FZOR	implements the operation OR between the α and NOT α values obtained with the two previous operations (LDP 1 6 and LDN 2 14)
LDK	stores the result of the operation OR in internal registers
LDM	copies the value of the memory register M in internal registers
FZOR	implements the operation OR between the last two values stored in internal registers (LDK and LDM)
CON crisp$_2$	multiplies the result of the last Ω operation with the crisp value $Crip_2$

At the end of the fuzzy rules by using the instruction OUT RAM_reg a byte is set then the control of the algorithm goes back to the CU.

2.5 ARITHMETIC LOGIC UNIT

The 8-bit Arithmetic Logic Unit (ALU) allows to perform arithmetic calculations and logic instructions which can be divided into 5 groups: Load, Arithmetic, Jump, Interrupts and Program Control instructions (refer to the ST52x440 Assembler Set for further details).

The computational time required for each instruction consists of one clock pulse for each Cycle plus 3 clock pulses for the decoding phase.

The ALU of the ST52x440 is able to perform multiplication (MULT) and division (DIV). The multiplication is performed by using 8 bit operands storing the result in 2 registers (16 bit values), see Figure 2.13.

The division is performed between a 16 bit dividend and an 8 bit divider, the result is stored in an 8 bit register (See Fig. 2.14)

Table 2.5. Arithmetic & Logic Instructions Set

Load Instructions						
Mnemonic	**Instruction**	**Bytes**	**Cycles**	**Z**	**S**	**C**
LDCE	LDCF conf, reg	3	17	-	-	-
LDCR	LDRC conf, reg	3	14	-	-	-
LDFR	LDFR FUZZY_i, reg	3	14	-	-	-
LDPE	LDPE per, reg	3	17	-	-	-
LDPR	LDPR reg, reg	3	14	-	-	-
LDRC	LDRC reg, const	3	14	-	-	-
LDRE	LDRE regi, regi	3	16	-	-	-
LDRE	LDRE (regi), (regj)	3	18	-	-	-
LDRI	LDRI regi, inp_reg	3	15	-	-	-
LDRR	LDRR regI, regj	3	16	-	-	-
PGSET	PGSET const	2	9	-	-	-

Arithmetic Instructions						
Mnemonic	**Instruction**	**Bytes**	**Cycles**	**Z**	**S**	**C**
ADD	ADD regi, regj	3	17	I	-	I
ADDO	ADDO regi, regj	3	20	I	I	I
AND	AND regi, regj	3	17	I	-	-
ASL	ASL regi	2	15	I	-	I
ASR	ASR regi	2	15	I	I	-
DEC	DEC regi	2	15	I	I	-
DIV	DIV regi, regj	3	26	I	I	I
INC	INC regi	2	15	I	-	I
MULT	MULT regi, regj	3	19	I	-	-
NOT	NOT regi	2	15	I	-	-
OR	OR regi, regj	3	17	I	-	-
SUB	SUB regi, regj	3	17	I	I	-
SUBO	SUBO regi, regj	3	20	I	I	I
MIRROR	MIRROR regi	2	15	I	-	-

Table 2.6. Arithmetic & Logic Instructions Set (Continue)

Jump Instructions						
Mnemonic	**Instruction**	**Bytes**	**Cycles**	**Z**	**S**	**C**
CALL	CALL addr	3	18	-	-	-
JP	JP addr	3	12	-	-	-
JPC	JPC addr	3	10/12	-	-	-
JPNC	JPNC addr	3	10/12	-	-	-
JPNS	JPNS addr	3	10/12	-	-	-
JPNZ	JPNZ addr	3	10/12	-	-	-
JPS	JPS addr	3	10/12	-	-	-
JPZ	JPZ addr	3	10/12	-	-	-
RET	RET	1	13	-	-	-

Interrupt Instructions Set						
Mnemonic	**Instruction**	**Bytes**	**Cycles**	**Z**	**S**	**C**
HALT	HALT	1	7/15	-	-	-
MEGI	MEGI	1	7/15	-	-	-
MDGI	MDGI	1	6	-	-	-
RETI	RETI	1	12	-	-	-
RINT	RINT INT	2	8	-	-	-
UDGI	UDGI	1	6	-	-	-
UEGI	UEGI	1	7/15	-	-	-
WAITI	WAITI	1	7/14	-	-	-

Control Instructions set						
Mnemonic	**Instruction**	**Bytes**	**Cycles**	**Z**	**S**	**C**
FUZZY	FUZZY	1	5	-	-	-
NOP	NOP	1	6	-	-	-
WDTRFR	WDTRFR	1	7	-	-	-
WDTSLP	WDTSLP	1	6	-	-	-

Notes:

I affected

- not affected

Figure 2.13 Multiplication

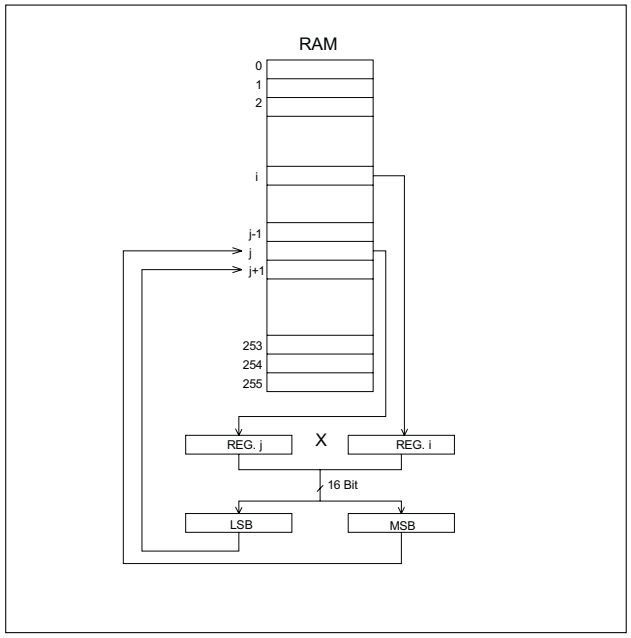
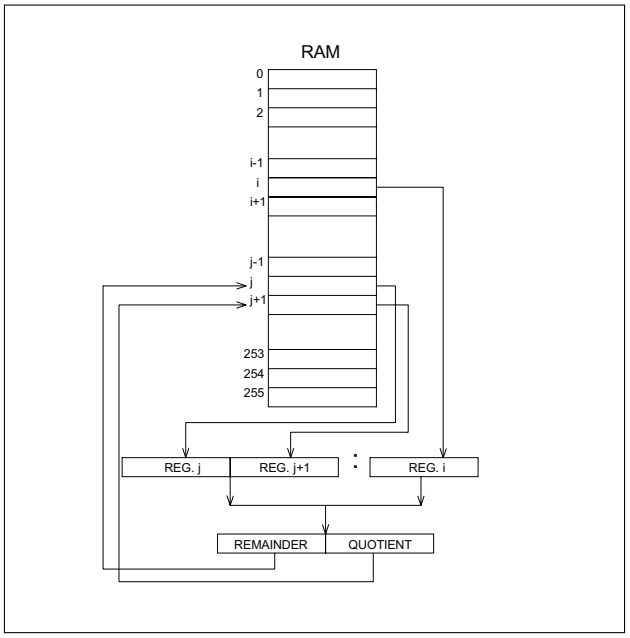


Figure 2.14 Division



3 EPROM

The EPROM memory provides an on-chip user-programmable non-volatile memory, that allows fast and reliable storage of user data.

The EPROM memory can be locked by user. In fact a memory location, called Lock Cell, is devoted to lock the EPROM and to avoid external operations. It is possible to write a software identification code, called ID CODE, to distinguish which software version is stored in the memory.

Table 3.1. EPROM Configuration Register

OPERATION	REGISTER VALUE
Stand By	0
Memory Reading / Verify	1
Memory Unlock and Lock Status Reading	2
Memory Writing	3
Memory Lock	4
ID CODE Writing	5
Memory Lock Status Reading / Verify	9
ID CODE Reading / Verify	10

There are 64 kbits of memory space with an 8-bit internal parallelism (8 kbytes) addressed by a 13-bit bus. The data bus is of 8 bits.

The memory has a double supply: V_{PP} is equal to 12V±5% in Programming Phase or to V_{SS} during Working Phase. V_{DD} is equal to 5V±10%.

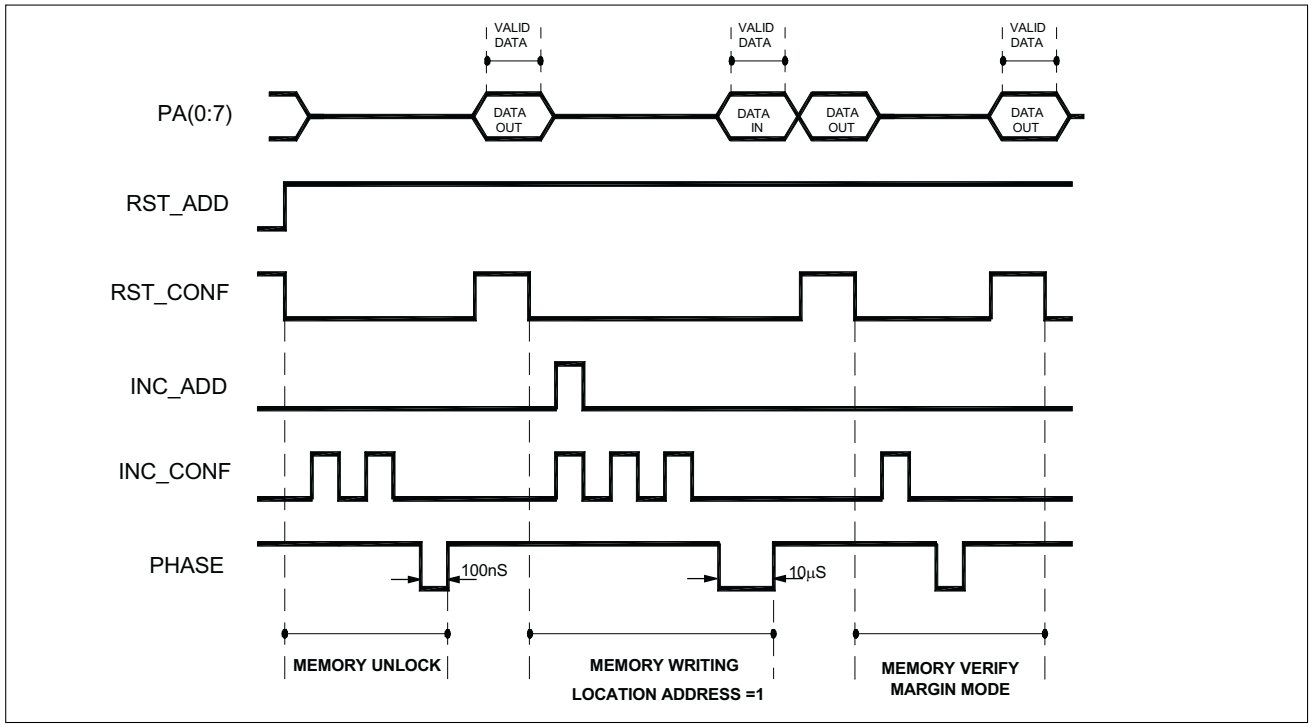
The ST52x440 EPROM memory is divided into three main blocks (see Figure 3.1):

- *Interrupt Vectors memory block* (3 through 14) contains the addresses for the interrupt routines. Each address is composed of three bytes.
- *Mbfs Setting memory block* (15 through 1024) contains the coordinates of the vertexes of every Mbf defined in the program.**If this part of the memory is not used to store the Mbfs setting, it can be use to store the instruction set on the user program**
- *The Program Instruction Set memory block* (1024 through 8191) contains the instruction set of the user program.

The locations 0, 1 and 2 contain the address of the first microcode instruction.

The operations that can be performed on the EPROM during the Programming Phase are: Stand By, Memory Writing, Reading and

Figure 3.2. EPROM Programming Timing



Verify/Margin Mode, Memory Lock, IDCode Writing and Verify.

Above operations are managed by using an internal 4-bit configuration register and an EPROM Configuration Register. The reading phase is executed with V_{PP}= 5V±5%, while the verify/Margin Mode phase needs V_{PP}= 12V±5%. The Blank Check must be a reading operation with V_{PP}= 5V±5%.

Table 3.1 shows the EPROM Configuration Register codes used to identify the running operation.

3.1 EPROM Programming Phase Procedure

The Programming mode is selected by applying 12V±5% voltage or 5V±5% voltage to the V_{PP} pin and setting the control signal as following:

RESET (pin 1) =Vss

TEST (pin 4) =Vss

If the V_{PP} voltage is 5V±5% it is possible only to read.

RST_ADD(pin 9), INC_ADD (pin 10), RST_CONF (pin 11), INC_CONF (pin 12) and PHASE (pin 5) are the control signals used during the Programming Mode.

PHASE, RST_CONF and RST_ADD signals are active on level, the others are active on rising edge.

PHASE and RST_ADD signals are active low, RST_CONF signal is active high.

Data in/out digital port is port B (from pin 18 up to pin 25).

It is possible to lock the memory by means of the Memory Lock Status, that is a flag used to enable the EPROM operations.

If Memory Lock Status is 1 all EPROM operations are enabled, otherwise, it is only possible to read (and verify) the OTP code and the Memory Lock Status.

Only If the EPROM is not locked by mean of Lock Cell (see paragraph 3.1.2), it is possible to enable the EPROM operations, changing the Memory Lock Status from 0 to 1 .

RST_ADD signal resets the memory address register and the Memory Lock Status. For this reason, when the RST_ADD becomes high, it is necessary to unlock the memory to read or write.

INC_ADD signal increments the memory address.

RST_CONF signal resets the EPROM Configuration Register. **When RST_CONF is high, the DATA I/O Port B is in output, otherwise it is always in input.**

INC_CONF signal increments the EPROM Configuration Register value.

PHASE signal validates the operation selected by means of EPROM Configuration Register value.

3.1.1 EPROM Operation

To execute one EPROM operation (See Table 3.1), the corresponding identification value must be loaded in the EPROM Configuration Register. The signal timing is the following: RST_ADD= high and PHASE= high, RST_CONF changes from low to high level, to reset the EPROM Configuration Register, and INC_CONF signal generates a number of positive pulses equal to the value to be loaded. After this sequence, a negative pulse of the PHASE signal will validate the selected operation. The minimum PHASE signal pulse width must be 10 µs for the EPROM Writing Operation and 100 ns for the others.

When RST_CONF is high, the DATA I/O Port B is enabled in output and the reading / verify operation results are available.

After a writing operation, when RST_CONF is high, the Port B is in output with no valid data.

3.1.2 EPROM Locking

The Memory Lock operation, that is identified with the number 4 in the EPROM Configuration Register, writes “0" in the Memory Lock Cell.

At the beginning of an External Operation, when RST_ADD signal changes from low level to high level, the Memory Lock Status is “0", therefore it is necessary to unlock it before to proceed.

To unlock the Memory Lock Status the operation, that is identified with the number 2 in the EPROM Configuration Register must be executed (see Figure 3.2).

The Memory Lock Status can be changed only if Memory Lock Cell is “1", therefore, for this reason, after a Memory Lock operation it is not possible to execute external operations except to read (or verify) the OTP Code and the Memory Lock Status.

3.1.3 EPROM Writing

When the memory is blank, all the bits are at logic level “1". The data are introduced by programming only the zeros in the desired memory location; however all input data must contain both ”1" and “0".

The only way to change “0" into ”1" is to erase the whole memory (by exposure to Ultra Violet light) and reprogram it.

The memory is in Writing mode when the EPROM Configuration Register value is 3.

The V_{PP} voltage must be 12V±5%, with stable data on the data bus PB(0:7).

The signals timing is the following (see Figure 3.2):

1) RST_ADD and RST_CONF change from low to high level,

2) two pulses on INC_CONF signal load the Memory Unlock operation code,

3) a negative pulse (100 ns) on the PHASE signal validates the Memory Unlock operation,

4) a negative pulse on RST_CONF signal resets the EPROM Configuration Register,

5) three positive pulses on INC_CONF load the Memory Writing operation code,

6) a train of positive pulses on INC_ADD signal increments the memory location address up to the requested value (generally this is a sequencial operation and only one pulse is used),

7) a negative pulse (10 μs) on the PHASE signal validates the Memory Writing operation,

3.1.4 EPROM Reading / Verify Margin Mode

The reading phase is executed with V_{PP}= 5V±5%, instead of verify phase that needs V_{PP}= 12V±5%.

The Memory Verify operation is available in order to verify the correctness of the data written. It is possible to execute a Memory Verify Margin Mode operation immediately after the writing of each byte and in this case (see Figure 3.2):

1) a positive pulse on RST_CONF signal resets the EPROM Configuration Register, if it was not already reset

2) one positive pulse on INC_CONF load the Memory Reading/Verify operation code,

3) a negative pulse (100 ns) on the PHASE signal validates the Memory Reading / Verify operation,

4) a negative pulse on RST_CONF signal puts in the PB(0:7) port the value stored in the actual memory address and resets the EPROM Configuration Register.

Then, if any error in writing occurred, the user has to repeat the EPROM writing.

3.1.5 Stand by Mode

The EPROM has a standby mode which reduces the active current from 10mA (Programming mode) to less than 100 μA. The Memory is placed in standby mode by setting PHASE signal at high level or when the EPROM Configuration register value is 0 and PHASE signal is low.

3.1.5 ID code

It is possible to write a software identification code, called ID code, to distinguish which software version is stored in the memory.

64 Bytes are dedicated to store this code by using the address values from 0 to 63.

It is possible to read or verify the ID Code also if the Memory Lock Status is “0".

The signals timing is the same of a normal operation.

3.2 Eprom Erasure

Thanks to the transparent window available in the CDIP28W package, its memory contents may be erased by exposure to UV light.

Erasure begins when the device is exposed to light with a wavelength shorter than 4000Å. It should be noted that sunlight, as well as some types of artificial light, includes wavelengths in the 3000-4000Å range which, on prolonged exposure, can cause erasure of memory contents. It is thus recommended that EPROM devices be fitted with an opaque label over the window area in order to prevent unintentional erasure.

The recommended erasure procedure for EPROM devices consists of exposure to short wave UV light having a wavelength of 2537Å. The minimum recommended integrated dose (intensity x expo-sure time) for complete erasure is 15Wsec/cm 2.

This is equivalent to an erasure time of 15-20 minutes using a UV source having an intensity of 12mW/cm 2 at a distance of 25mm (1 inch) from the device window.

4 INTERRUPTS

The Control Unit (CU) responds to peripheral events and external events through its interrupt channels.

When such an event occurs, if the related interrupt is not masked and according to a priority order, the current program execution can be suspended to allow the CU to execute a specific response routine.

Each interrupt is associated with an interrupt vector that contains the memory address of the related interrupt service routine. Each vector is located in the Program Space (EPROM Memory) at a fixed address (see Interrupt Vectors table fig. 4.2).

4.1 Interrupt Functionment

If, at the end of an arithmetic or logic instruction, there are pending interrupts, the one with the highest priority is passed. To pass an interrupt means to store the arithmetic flags and the current PC in the stack and execute the associated Interrupt routine, whose address is located in two bytes of the EPROM memory location between address 3 and 20.

The Interrupt routine is performed as a normal code checking, at the end of each instruction, if an higher priority interrupt has to be passed. An Interrupt request with the higher priority stops the lower priority Interrupt. The Program Counter and the arithmetic flags are stored in the stack.

With the instruction RETI (Return from Interrupt) the arithmetic flags and Program Counter (PC) are restored from the top of the stack. This stack was already described in the section 2.2.1.

An Interrupt request cannot stop the processing of the fuzzy rules, but this is passed only after the end of a fuzzy rule or at the end of a logic, or arithmetic, instruction.

REMARK: A fuzzy routine can be interrupted only in the Main program. An interrupt request cannot stop a Fuzzy function that is running inside another interrupt routine. For this reason, to use a Fuzzy function inside an interrupt routines, the user MUST include the Fuzzy function between an UDGI (MDGI) instruction and and UEGI (MEGI) instruction (see the following paragraphs), in order to disable the interrupt request during the execution of the fuzzy function.

Figure 4.1. Interrupt Flow

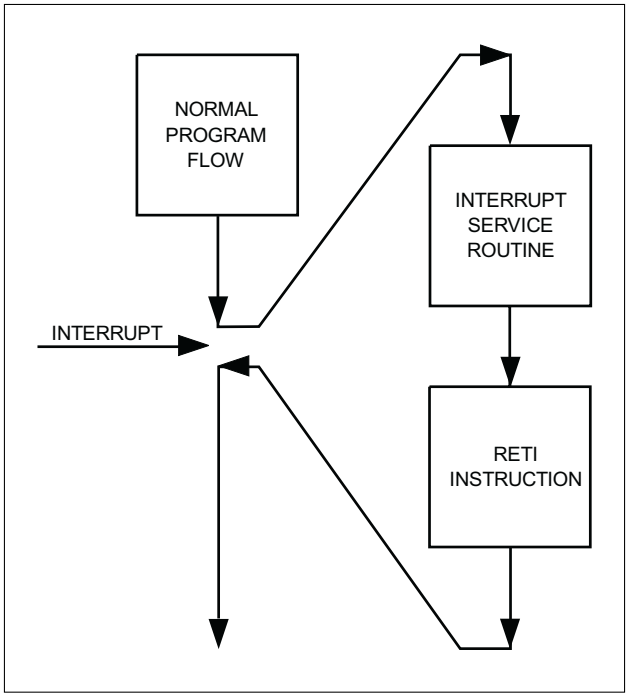


Figure 4.2. Interrupt Vectors Mapping

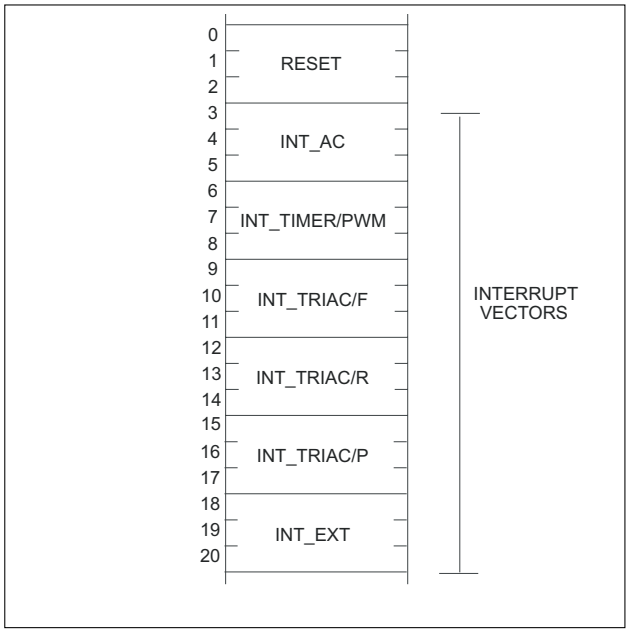
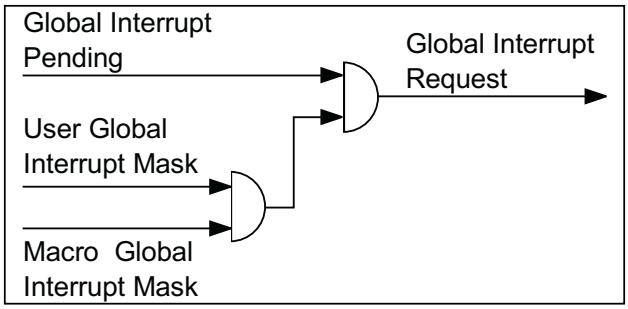


Figure 4.3 Global Interrupt Request generation



4.2 Global Interrupt Request Enabling

When an Interrupt occurs, it generates a Global Interrupt Pending (GIP), that can be hanged up by software. After a GIP a Global Interrupt Request (GIR) will be generated and Interrupt Service Routine associated to the interrupt with higher priority will start.

In order to avoid possible conflicts between interrupt masking set in the main program, or inside macros, the GIP is hanged up through the User Global Interrupt Mask or the Macro Global Interrupt Mask (see fig.4.3).

UEGI/UDGI instruction switches on/off the User Global Interrupt Mask enabling/disabling the GIR for the main program.

MEGI/MDGI instructions switches on/off the Macro Global Interrupt Mask in order to ensure that the macro will not be broken.

4.3 Interrupt Sources

ST52x440 manages interrupt signals generated by the internal peripherals (TIMER/PWM, TRIAC/PWM and Analog Comparator) or coming from the External INTERRUPT on pin PA7. The External Interrupt can be programmed to be active on the rising or falling edge of INT/PA7 signal by setting the PEXTINT bit of Configuration Register 0.

Each peripheral can be programmed in order to generate the associated interrupt; further details are described in the related chapter.

4.4 Interrupt Maskability

The interrupts can be masked by configuring the REG_CONF 0 by means of an LDCR or an LDCE instruction. The interrupt is enabled when the bit associated to the mask interrupt is “1". Viceversa, when the bit is ”0", the interrupt is masked and it is kept pendent.

For example:

```
LDRC 10,6 // loads the constant 6 in
the RAM Register 10
LDCR 0, 10 // sets CONF_REG 0 with the
value stored in RAM Register 10
```

the result is CONF_REG0=00000110 thus enabling the interrupts coming from the Analog Comparator (INT_AC) and from the PWM/TIMER (INT_PWM/TIMER).

Table 4.1. Configuration Register 0 Description

Bit	Name	Value	Description
0	MSKE	0	External Interrupt Masked
		1	External Interrupt Not Masked
1	MSKAC	0	Analog Comparator Interrupt Masked
		1	Analog Comparator Interrupt Not Masked
2	MSKTM	0	PWM/TIMER Interrupt Masked
		1	PWM/TIMER Interrupt Not Masked
3	MSKTRR	0	TRIAC Rising Edge Interrupt Masked
		1	TRIAC Rising Edge Interrupt Not Masked
4	MSKTRF	0	TRIAC Falling Edge Interrupt Masked
		1	TRIAC FallingEdge Interrupt Not Masked
5	MSKTRP	0	TRIAC Pulse Interrupt Masked
		1	TRIAC Pulse Interrupt Not Masked
6	PEXTINT	0	External Interrupt active on Rising Edge
		1	External Interrupt active on Falling Edge
7	MSKBR	0	Brown-Out Disabled
		1	Brown-Out Enabled

Reset Configuration ‘00000000’

Table 4.2. Interrupts Description

Name	Description		Priority	Peripheral Code	Maskable	Vector Addresses
INT_AC	Analog Comparator	Int	Programmable	000	yes	3-5
INT_PWM/TIMER	PWM/TIMER	Int	Programmable	001	yes	6-8
INT_TRIAC/F	TRIAC Falling Edge	Int	Programmable	010	yes	9-11
INT_TRIAC/R	TRIAC rising edge	Int	Programmable	011	yes	12-14
INT_TRIAC/P	TRIAC Pulse	Int	Programmable	100	yes	15-17
INT_EXT	External Interrupt (INT)	Ext	Highest	-	yes	18-20

Figure 4.4. Interrupt Configuration Register 0

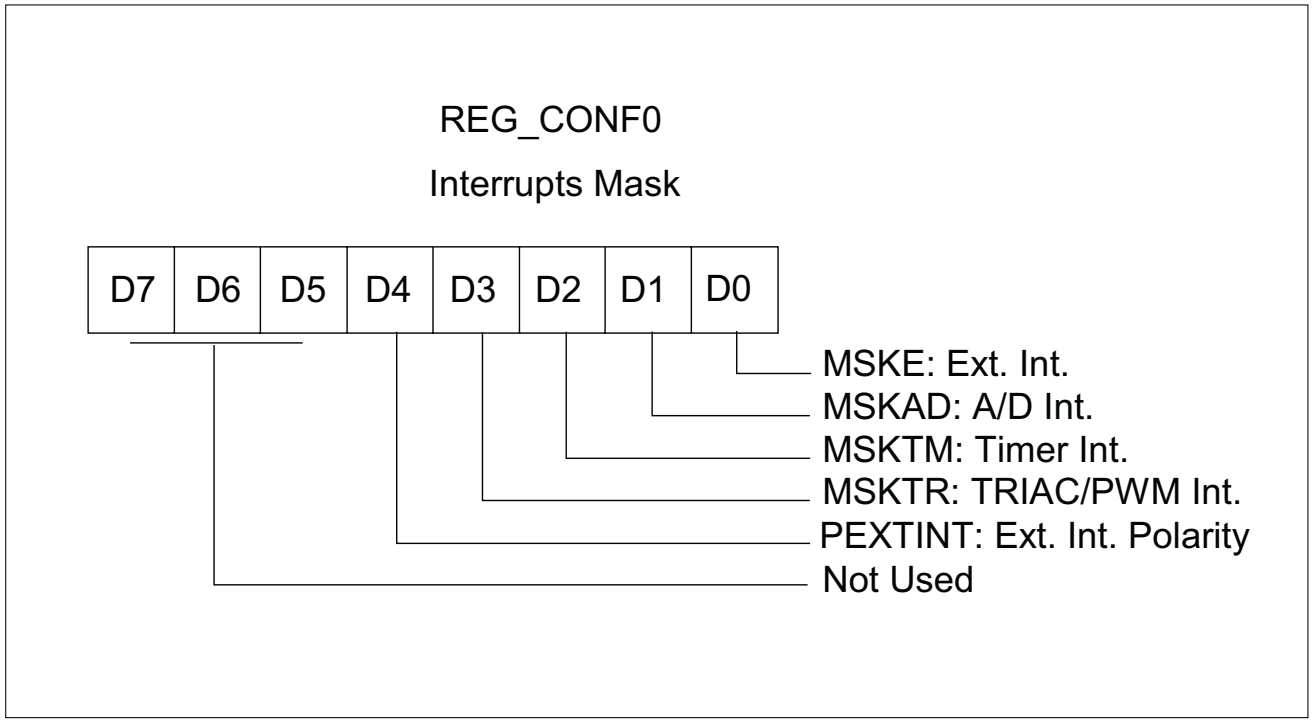
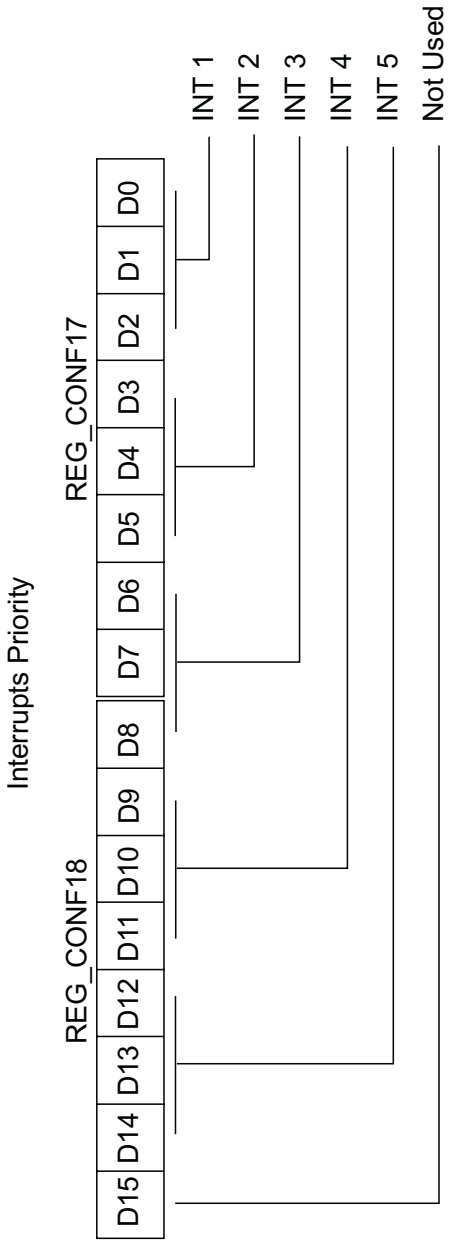


Figure 4.5. Interrupt Configuration Registers 17 and 18



4.5 Interrupt Priority

Seven priority levels are available: level 6 has the lowest priority, level 0 has the highest priority.

Level 6 is associated to the Main Program, levels 5 to 1 are programmable by means of the priority registers called REG_CONF 17 and REG_CONF 18; whereas the higher level is related to the external interrupt (INT_EXT).

PWM/Timer, TRIAC/PWM and Analog Block are identified by a three-bits Peripheral Code (see Table 4.2); in order to set the *i*-th priority level the user must write the peripheral label *i* in the related INT*i* priority level.

For instance:

```
LDRC 10, 193 //(load the value 193='11000001' in the RAM Register 10)
```

LDRC 11, 168 //(load the value 168='10101000' in the RAM Register 11)

```
LDCR 17, 10 // set the REG_CONF17= '11000001'
LDCR 18, 11 // set the REG_CONF18= '10101000'
```

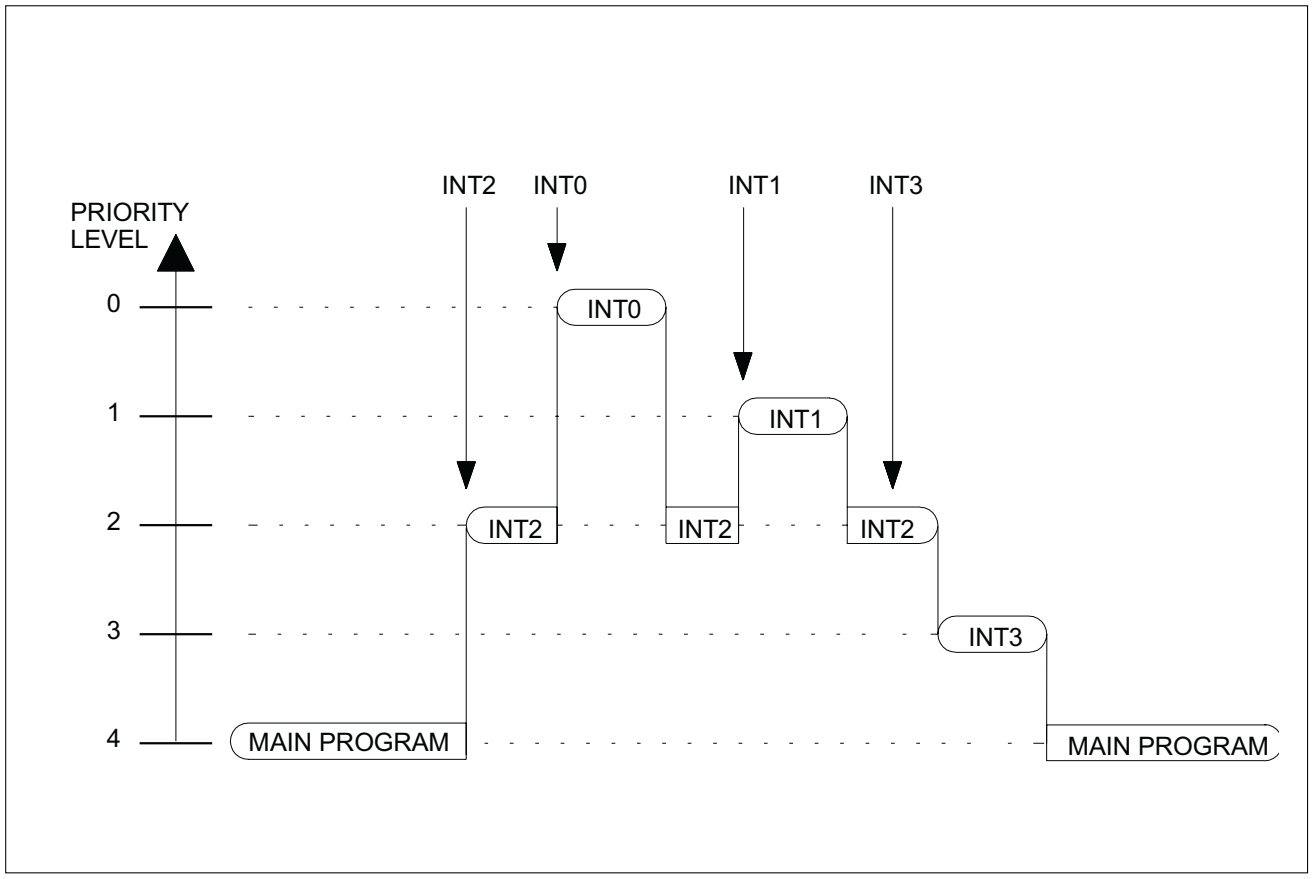
Table 4.3. Conf. Registers 17-18 Description

Bit	Name	Value	Level
0, 1, 2	INT1	Peripheral Code	High
3, 4, 5	INT2	Peripheral Code	MediumHigh
6, 7, 8	INT4	Peripheral Code	MediumLow
9, 10, 11	INT5	Peripheral Code	Low
12, 13, 14	INT6	Peripheral Code	Very Low

thus defining the following priority levels:

- Level 1: INT_PWM/TIMER (PWM/TIMER Code: 001)
- Level 2: INT_ADC (ADC Code: 000)
- Level 3: INT_TRIAC/PWM_R (TRIAC/PWM Code: 011)
- Level 4: INT_TRIAC/Ph (TRIAC/Ph Code: 100)

Figure 4.6. Example of a Sequence of Interrupt Requests



- Level 5: INT_TRIAC/PWM_F (TRIAC/PWM_F Code: 010)

REMARK: The Interrupt priority must be fixed ad the beginning of the main program, because at the RESET REG_CONF1='00000000', and this condition could generate wrong operations. During the program execution it is possible to modify the interrupt priority only with the following procedure:

STEP 1:

Mask the interrupts by means of a UDGI (or MDGI) instruction

STEP 2:

Change the REG_CONF 17&18 values to modify the interrupt priority

STEP 3:

Reset by means of RINT instructions all the pending interrupt routines

STEP 4:

Unmask the interrupts by means of a UEGI (or MEGI) instruction

When a source provides an Interrupt request, and the request processing is also enabled, the CU changes the normal sequential flow of a program by transferring program control to a selected service routine.

When an interrupt occurs the CU executes a JUMP instruction to the address loaded in the related location of the Interrupt Vector.

When the execution returns to the original program, it begins immediately following the interrupted instruction.

4.6 Interrupts and Low power mode

All interrupts allow the processor to leave the WAIT low power mode. Only the external Interrupt, when enabled, allows the processor to leave the HALT low power mode *when it is not masked*. ***Verify***???

4.7 Interrupt RESET

An eventually pending interrupt can be reset with the instruction `RINT j`, which resets the interrupt *j*-th where *j* identify the peripherals as described in the following table. (see table 4.4).

The assembler instruction:

```
RINT 2
```

Resets the TRIAC/PWM_F interrupt.

Table 4.4. RINT instruction code

Peripheral Name	Value
Analog Comparator	0
PWM/TIMER	1
TRIAC/F	2
INT_TRIAC/R	3
INT_TRIAC/P	4
External Interrupt	5

5 CLOCK, RESET & POWER SAVING MODE

5.1 Clock System

The ST52x440 Clock Generator module generates the internal clock for the internal Control Unit, ALU and on-chip peripherals and it is designed to require a minimum of external components.

The ST52x440 oscillator circuit generates an internal clock signal with the same period and phase as at the OSCin input pin. The maximum frequency allowed is **20 MHz**.

The system clock may be generated by using either a quartz crystal, or a ceramic resonator (CERALOC); or, at least, by means of an external clock.

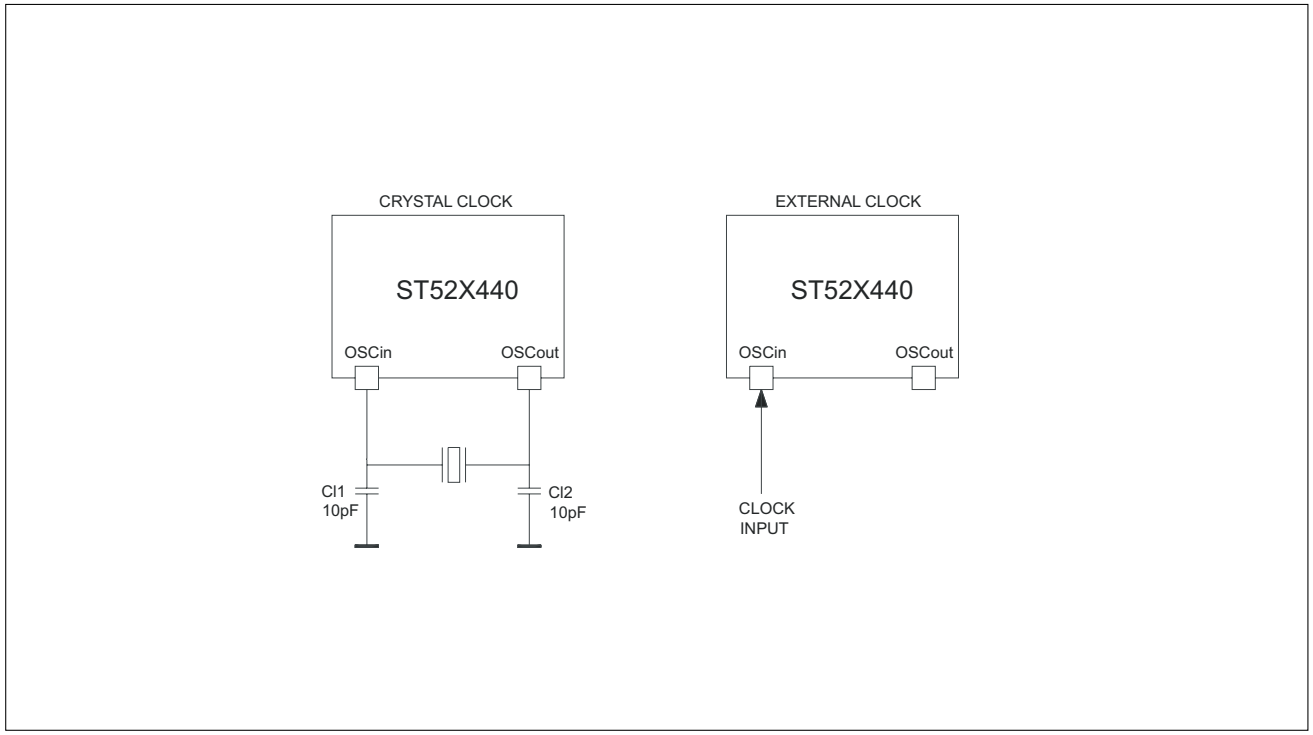
The different clock generator options connection methods are shown in Figure 5.1.

When an external clock is used, it must be connected on the pin OSCin while OSCout can be floating.

The crystal oscillator start-up time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitance (CL), IC parameters, ambient temperature, supply voltage.

It must be observed that the crystal or ceramic leads and circuit connections must be as short as possible. Typical values for CL1, CL2 are 10pF for a 20 MHz crystal.

Figure 5.1 Oscillator Connections



5.2 RESET

There are four sources of Reset:

- RESET pin (external source)
- WATCHDOG (internal source)
- POWER ON Reset (Internal source)
- BROWN OUT Reset (Internal source)

When a Reset event happens, the user program restarts from the beginning.

5.2.1 External Reset

The Reset pin is an input. An internal reset does not affect this pin.

A Reset signal originated by external sources is instantaneously recognized. The RESET pin may be used to ensure Vdd has risen to a point where the MCU can operate correctly before the user program is run. In working mode the Reset must be set to '1' (see Table 1.2)

Figure 5.2 Reset Block Diagram (extended)

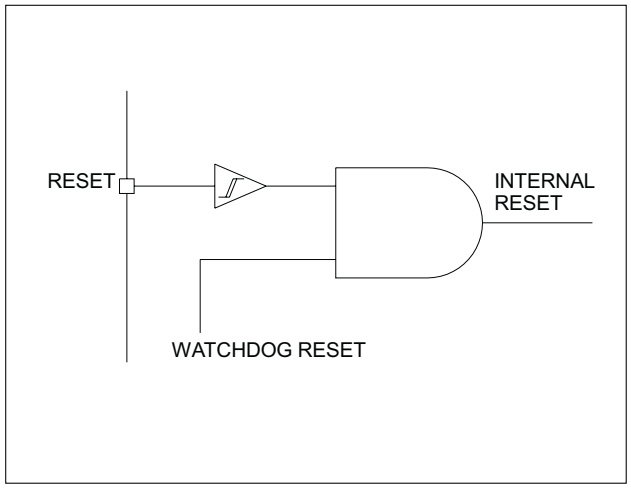
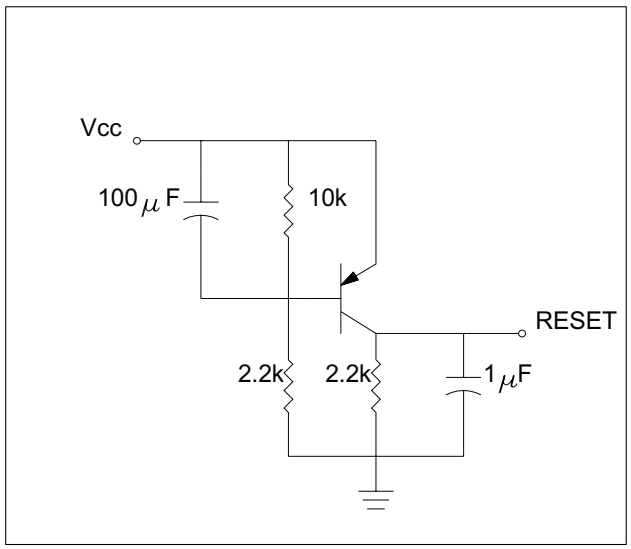


Figure 5.3 Simple Reset Circuit



5.2.1 Reset Operation

The duration of a RESET condition is fixed at 1.000.000 internal CPU clock cycles (or 4096 in case of BOR).

Following a Power-On Reset event, or after exiting Halt Mode, a 1.000.000 CPU clock cycle delay period is initiated in order to allow the oscillator to stabilise and to ensure that recovery has taken place from the Reset state.

A Pull up resistor of 100 KΩ guarantees that RESET pin be at level "1" when no HALT nor Power-On events occurred.

If an external resistor is connected to the RESET pin a minimum value of 10KΩ must be used.

After a RESETprocedure is completed, the core reads the instruction stored in the first 3 bytes of the EPROM, that contains a JUMP instruction to the EPROM address containing the first intruction of the user program. The Assembler program automatically generates this Jump instruction with the first instruction address.

5.2.2 Power-on Reset (POR)

A Power-On Reset is generated by an on-chip detection circuit. This circuit ensures that the device is not started until the Vdd has reached the nominal level of 2.3V and allows the clock oscillator to stabilize.

Once reached 2.3 V, the Power-On circuit generates an internal RST signal that release the internal reset to the CPU and invokes a delay counter of 1.000.000 CPU clock cycles, during which the device is kept in RESET after Vdd rise.

A correct working of Power-on detector is guaranteed if the slew rate of Vdd is 0.05mV/s.

The power supply must fall below 1V for the internal POR circuit to detect the next rise of Vdd. This event however is not possible if the microcontroller is in HALT mode condition: in this case Vdd must fall at 0 V for the POR be re-initialized.

At power on the POR is enabled by default.

POR is designed exclusively to cope with power-up conditions and should not be used to detect a drop in the power supply voltage, for which the Brown-out Detector can be used instead.

5.2.3 Brown-Out Detector (B0R)

The on-chip Brown-Out Detector circuit allows to prevent the processor to fall into an unpredictable manner if the power supply drops below a certain level.

When Vdd drops below the Brown-out detection level, the Brown-out causes an internal processor reset RST that remains active as long as Vdd remains below the Brown-Out Trigger Level.

The Brown-Out resets all the device unless the Power-on Detector and the Brown-out itself.

Enabling/disabling of Brown-out detector can be performed through software setting of the control bit BOR of REG_CONF0 (Table 4.1)

When Vdd increases above the Trigger Level, the Brown-Out reset is deactivated after a delay of 4096 CPU clock cycles, that ensures stabilization of the oscillator.

The Brown-Out falling voltage level typical value is 3.65V and the corresponding rising voltage activation level 3.96V.

A minimum hysteresis of 100mV *for the trigger is guaranteed for spike free brown-out* detection.

Brown-Out circuit will only detect a drop if Vdd voltage stays below the safe threshold for longer than 5 μs before activation/deactivation of the Brown-Out is guaranteed to filter voltage spikes.

Brown-Out function is not enabled by default and is not active when in HALT mode.

5.3 POWER SAVING MODES

There are two Power Saving modes: WAIT and HALT mode. These conditions may be entered using the WAIT or HALT instructions.

5.3.1 Wait Mode

Wait mode places the MCU in a low power consumption by stopping the CPU. All peripherals and the watchdog remain active. During the WAIT mode, the Interrupts are enabled. The MCU will remain in Wait mode until an Interrupt or a RESET occurs, whereupon the Program Counter jumps to the interrupt service routine or, if a RESET occurs, at the beginning of the user program.

5.3.2 Halt Mode

The Halt mode is the MCU lowest power consumption mode. The Halt mode is entered by executing the HALT instruction. The internal

Figure 5.4 WAIT Flow Chart

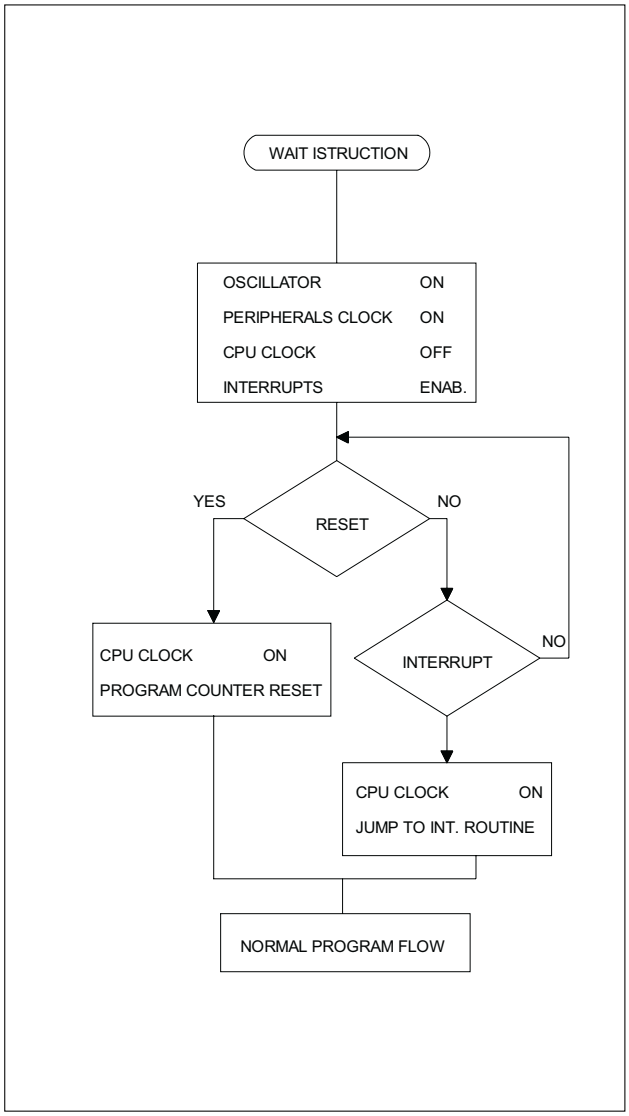
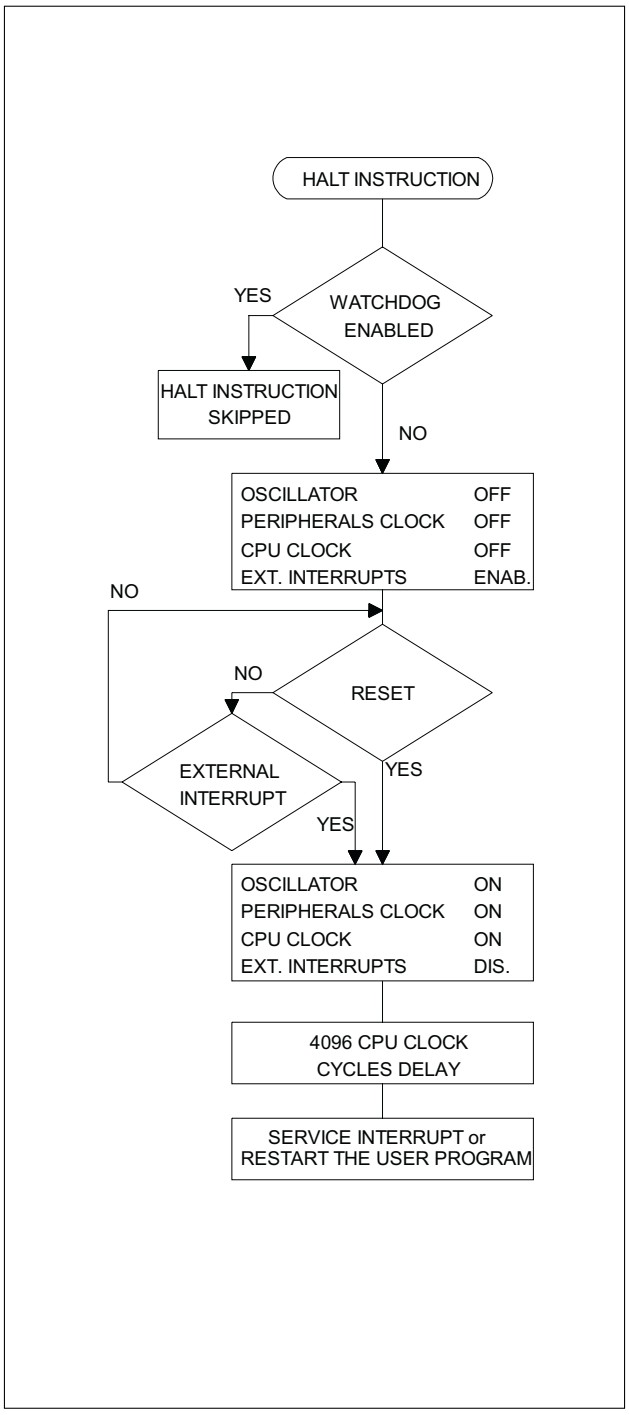


Figure 5.5 HALT Flow Chart



oscillator is turned off, causing all internal processing to be stopped, including the operations of the on-chip peripherals. **The Halt mode cannot be used when the watchdog is enabled.** If the HALT instruction is executed while the watchdog system is enabled, it will be skipped without modifying the normal CPU operations.

In Halt mode the external interrupt is enabled. If an interrupt occurs, the CPU becomes active.

The MCU can exit the Halt mode upon reception of an external interrupt if it is not masked or a reset. The oscillator is then turned on and a stabilization time is provided before restarting the CPU operations. The stabilization time is 4096 CPU clock cycles.

After the start up delay, the CPU restarts the operations by serving the external interrupt routine or from the beginning of the user program.

6. I/O PORTS

6.1 Introduction

ST52x440 device offers flexible individually programmable multi-functional input/output lines. Refer to the following figure for specific pin allocations.

20 I/O lines , grouped in 3 different ports, are available for **ST52x440G** devices:

PORT A = 8-bit ports (PA0 - PA7 pins)

PORT B = 8-bit ports (PB0 - PB7 pins)

PORT C = 5-bit port (PC0 - PC4 pins)

14 I/O lines , grouped in 2 different ports, are available for **ST52x440I** devices:

PORT A = 8-bit ports (PA0 - PA7 pins)

PORT B = 8-bit ports (PB0 - PB7 pins)

These I/O lines can be programmed to provide Digital Input/Output and Analog Input, or to connect input/output signals to the on chip peripherals as Alternate Pin Functions.

The input buffers are TTL compatible with Schmitt Trigger in ports A and C while port B is CMOS compatible without Schmitt trigger.

The output buffer is able to supply up to 8 mA.

The port cannot be configured to be at the same time as an input and as an output.

Each port is configured by using Configuration Registers indicated in Table 6.1. The first is used to define if a pin is an Input or an Output, the second defines the Alternate functions.

Table 6.1 I/O Port Configuration Registers

PORT A	PORT B	PORT C (*)
Reg_Conf 4	Reg_Conf 13	Reg_Conf 11
Reg_Conf 12	Reg_Conf 14	
	Reg_Conf 11(**)	

(*) Not available in ST52x440I

(**) Only in ST52x440I

6.2 Input Mode

The input configuration is selected setting to “1" the corresponding configuration register bit in REG_CONF4, REG_CONF13 and, where applicable, REG_CONF11 (see Paragraph 6.5) .

The digital input data are automatically stored in the Input Registers, but it is not possible to read

Figure 6.1 Ports A and C Functional Blocks

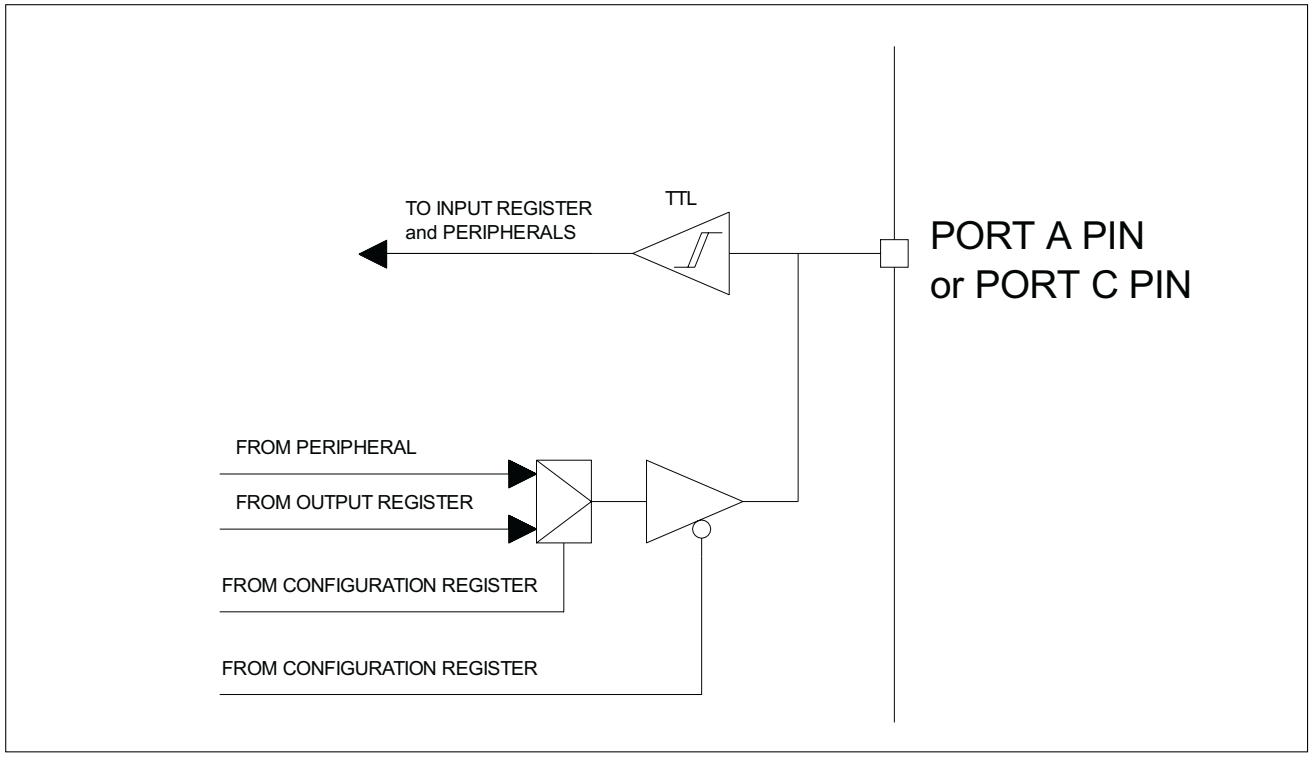
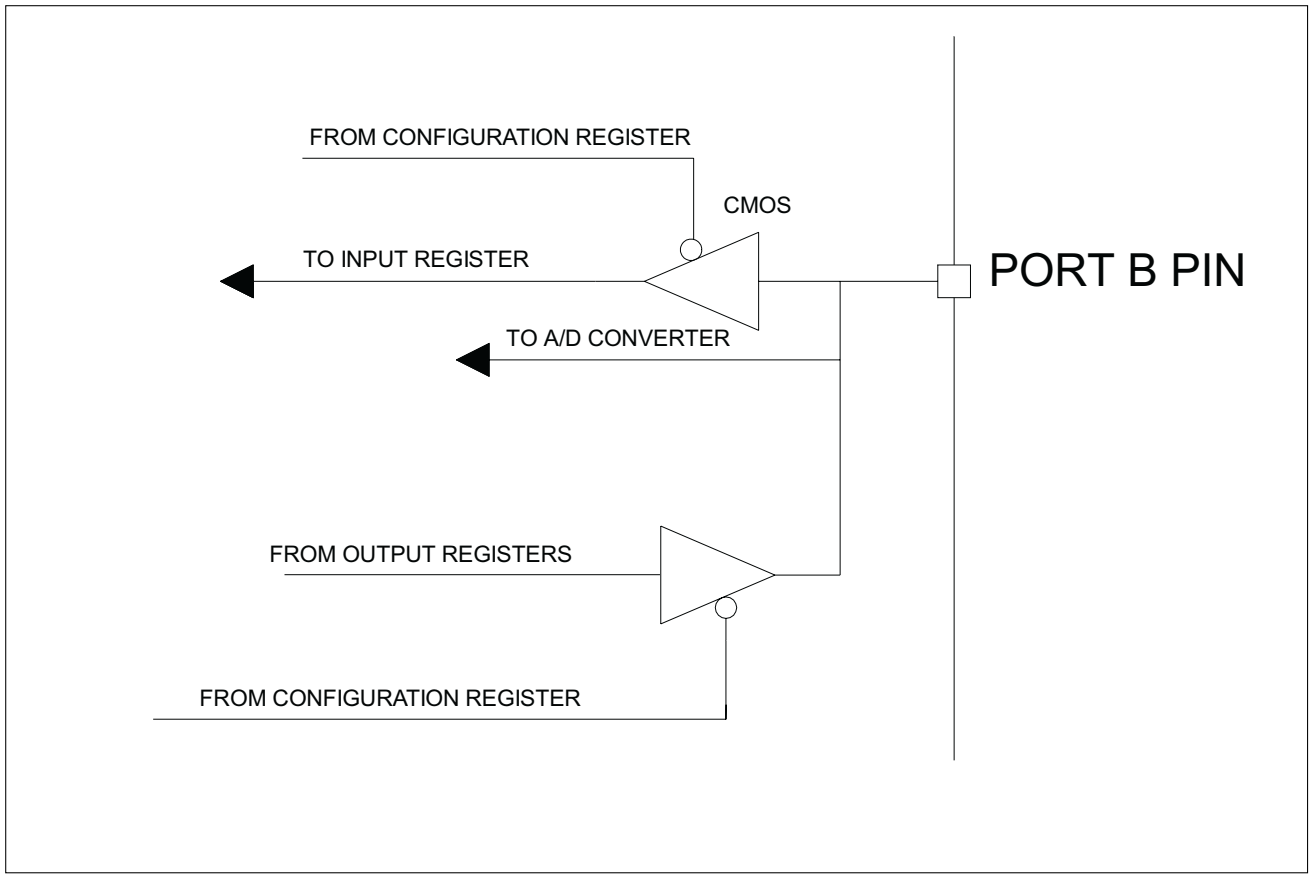


Figure 6.2 Port B Functional Blocks



directly the single bit of the Input Register (IR) and it is necessary to copy the value in a RAM location. The digital data are stored in a RAM location by using the assembler instruction:

```
LDRI RAM_Reg Input_i
```

Table 6.2 Input Register and I/O Ports

PORT A	PORT B	PORT C (*)
IR 14	IR 15	IR 16

(*) Not used for ST52x440I

Table 6.3 Output Register and I/O Ports

PORT A	PORT B	PORT C (*)
OR0	OR 1	OR 2

(*) Not used for ST52x440I

6.3 Output Mode

The output pin configuration is selected setting to '0' the corresponding configuration register bit (REG_CONF4, REG_CONF13 and, where applicable, REG_CONF11) (see paragraph 6.5).

The digital data are transferred to the related I/O Port by means of the Output register, by using the assembler instructions LDPE or LDPR.

6.4 Alternate Functions.

Several ST52x440 pins are configurable to be used with different functions (Alternate Functions) (Table 1.1).

When an on chip peripheral is configured to use a pin, it is mandatory to select the correct I/O mode of the related pin.

For example: if pin 20 PA5/TCLK (on ST52x440G) has to be used as external PWM/Timer Clock, REG_CONF4[(5)] bit must be set to '1'.

When the signal is an input of an on-chip peripheral, the related I/O pin has to be configured in Input Mode.

When a pin of Port B is used as an Analog Input, the related I/O pin is automatically set in threestate. The analog multiplexer (controlled by the Analog Comparator Configuration Register) switches the analog voltage present on the selected pin to the common analog rail which is connected to the ADC input.

It is recommended not to change the voltage level or loading on any port pin while conversion is running. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

6.5 I/O Port Configuration Registers

The I/O mode for each bit of the three ports are selected by using Configuration Registers 4, and 12, 13 and 11 (Table 6.1) The structure of these registers is shown in the following tables.

Each bit of the configuration registers sets the I/O mode of the related port pin.

Analog Comparator Inputs

Pins PB0-PB5 for ST52x440G and PB0-PB3 in case of ST52x440I can be configured to be Analog Inputs setting to “1” the related bit in REG_CONF 14 (Table 6.7) and to “1” the related bit in REG_CONF13 (Table 6.6). These analog inputs are connected to the on chip 8-bit Analog to Digital Converter.

If the BandGap Reference (BG) is needed as an Ouput, in ST52x440I REG_CONF13[3] must be set to “0”, REG_CONF11[5] to “0” and REG_CONF14[3] to “1”, while for ST52x440G REG_CONF13[6] must be set to “0” and REG_CONF14[6] to “1”.

Timer/PWM Alternate Functions

Pins of Port A can be configured to be I/Os of the on-chip TIMER/PWM of ST52x440. The configuration of these pins is performed by using Configuration Registers REG_CONF4 and REG_CONF12 (Tables 6.4 and 6.5).

If a pin has to be a TIMER Input (TSTRT, TCLK, TRES) the related bit of REG_CONF4 must be set to “1” and of REG-CONF12 must be set to “1”.

If instead it must be a TIMER Output (TOUT, TOUTN), REG_CONF12 related bit must be set to “0” and the related bit of REG_CONF4 be set to “0”.

TRIAC Alternate Function

When using the on-chip TRIAC, to have the TRIAC Output on pin PA0,bit REG_CONF12[0] must be set to “0” and REG_CONF4[0] to “0”.

When a synchronization with the Mains voltage is necessary, in case either the Phase Angle Partialization or the Burts Modes is chosen, to have MAIN1 and MAIN2 as Inputs on PortA, it is necessary to set bits 1 and 2 of REG_CONF4 to “1”.

Table 6.4 - Port A - REG_CONF 4

Bit	Name	Value	Pin Description
0	D0	X	PA0
1	D1	X	PA1/MAIN1
2	D2	X	PA2/MAIN2
3	D3	X	PA3/ACSTRT
4	D4	X	PA4/TSTRT
5	D5	X	PA5/TCLK
6	D6	X	PA6/TRES
7	D7	X	PA7/INT
X = 0 Pin set as Digital Output X = 1 Pin set as Alternate Function Input			
Reset Configuration ‘1111’			

Table 6.5 - Port A - REG_CONF12

Bit	Name	Value	Pin Description
0	D0	X	PA0/TROUT
1	D1	X	PA6/TOUT
2	D2	X	PA2/TOUTN
3	D3	X	PA7/ACSYNC
4	D4		not used
5	D5		not used
6	D6		not used
7	D7		not used
X = 0 Pin set as Alternate Function Output X = 1 Pin set as Digital I/O			
Reset Configuration ‘1111’			

Table 6.6 - Port B - REG_CONF 13

Bit	Name	Value	Related Pin
0	D0	X	PB0
1	D1	X	PB1
2	D2	X	PB2
3	D3	X	PB3
4	D4	X	PB4(*)
5	D5	X	PB5(*)
6	D6	X	PB6(*)
7	D7	X	PB7
X = 0 Pin set as Output X = 1 Pin set as Input			
Reset Configuration '11111111'			

(*) **Not available in ST52x440I**

Table 6.7 Analog Inputs REG_CONF11

Bit	Name	Value	Pin Description
0	D0	X	PB0/AC0
1	D1	X	PB1/AC1
2	D2	X	PB2/AC2
3	D3	X	PB3/AC3
4	D4	X	PB4/AC4(*)
5	D5	X	PB5/AC5(*)
6	D6		not used
7	D7		not used
X = 0 Pin set as Digital I/O X = 1 Pin set as Analog Input			
Reset Configuration '00000000'			

(*) **Not available in ST52x440I**

Table 6.8 - Port C and B - REG_CONF 11

Bit	Name	Value	Related Pin
0	D0	X	PC0(*)
1	D1	X	PC1(*)
2	D2	X	PC2(*)
3	D3	X	PC3(*)
4	D4	X	PC4(*)
5	D5	Y	PB3(**)
6	D6		not used
7	D7		not used
X = 0 Pin set as Digital Output X = 1 Pin set as Digital Input Y = 0 Pin setas BG reference Y = 1 Pin set as Analog Input (AC3)			
Reset Configuration '11111111'			

(*) **Not used in ST52x440I**

(**) **Must be used to set BG as an Ouput only in ST52x440I**

7. ANALOG COMPARATOR

Analog Module Overview

The ST52x440 includes among its peripherals an Analog Comparator (AC).

This peripheral is endowed with analog and digital elements in order to allow the user to use it also as single slope Analog to Digital converter. In particular ST52X440 is endowed with:

- Analog Comparator
- 7 channels analog mux (6 external lines, 1 internal voltage reference)
- a current source providing 7 values
- 16 bit Timer with a Capture Register and 12 bit Prescaler

The selection of the working mode, as either Analog Comparator or single slope A/D, can be done through REG_CONF3[(0)] bit (Table 7.2).

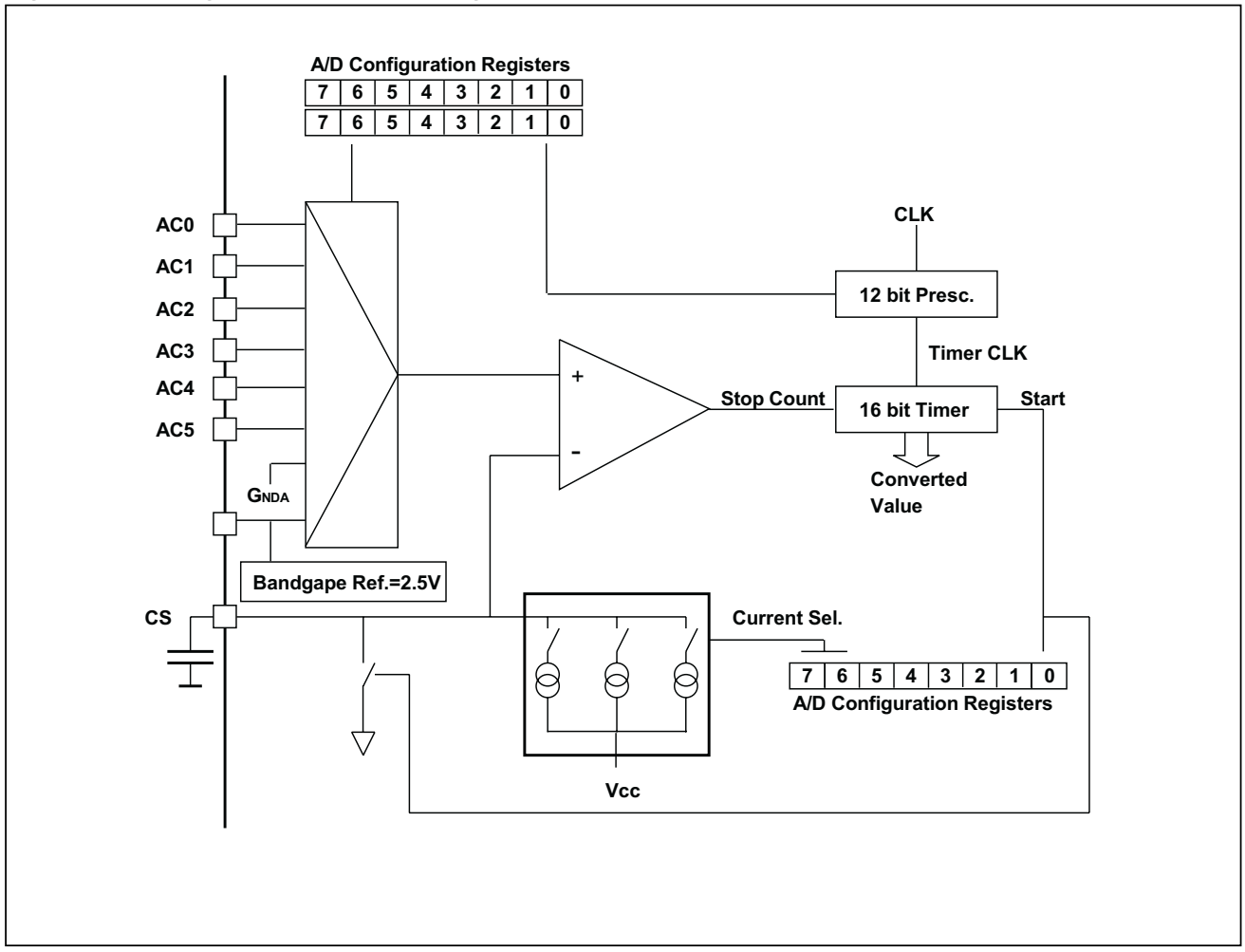
In Analog Comparator mode, REG_CONF3[(0)] is set to “0” . The AC inputs are available on the external pins. The reference signal must be connected pin CS, the signal to be compared goes to one of the 6 analog inputs AC0-AC5 in the case of ST52x440G or 4 analog inputs AC0-AC3 for ST52x440I. When the input become greater than the reference, the Analog Comparator output changes to value “1”; this output can be read on the less significative bit of Input Register 1 AC_CHAN0H (Table 2.1).

To use the Analog Comparator for A/D Conversion, REG_CONF3[(0)] bit must be set to “1”.

Table 7.1 12 Bit A/D Conversion Time

A/D Timer Clock	Conversion Time
20 MHz	204 µS
10 MHz	409.6 µS
5 MHz	819.2 µS

Figure 7.1 Analog Peripheral Block Diagram



In A/D Mode it is possible to choose either the insertion of an external capacitor to pin CS or the connection to an external signal to generate the reference ramp.

In the first case REG_CONF16[(0)] bit must be set to “0”. The internal current generator, used to charge the capacitor, provides 7 possible current values, that can be selected through REG_CONF3[(7:5)].

The 16 bit Timer, directly triggered by the output of the analog comparator, allows to measure the conversion time.

The Timer clock is divided by an internal 12 bit Prescaler. The maximum conversion time for a 12 bit resolution is resumed in the following table when in the case of a master clock of 20 Mhz.

The Timer clock is obtained by prescaling the master clock.

In the second case REG_CONF16[(0)] bit must be set to “1”.

When an appropriate value of the capacitor is selected, the conversion should be complete before the full count is reached.

A timer overflow flag is set once the Timer reaches its maximum count value.

Generally the conversion time of the A.D converter depend on the bit resolution and on the Timer clock period. The maximum conversion time can be calculated using the following formula:

$$ConversionTime = \frac{1}{TimerClk} \times 2^N$$

Where TimerClk is the Timer clock and N is the number of bit of the desired resolution.

In order to avoid the errors introduced by the A/D components drift, a periodic conversion on the internal $\mathbf{V_{ref}}$ should be performed. Two different internal voltage references are available:

1) Bandgap voltage = 2.5 V, this reference voltage can be used also externally for analog signal conditioning.

2) GNDA.

3 bit allow to set the 8 different current values of the internal current source that is used to charge the external capacitor generating the ramp voltage to the Analog Comparator. The current values are in the range 0 to 70μA with step of 10μA . The current source is internally connected to the CS pin and is used to charge the external capacitor. This capacitor should have a low voltage coefficient for optimum results. In order to achieve the wished resolution, conversion time and full scale input voltage it is necessary to select an appropriate capacitor value. The following formula can be used to fix the best value of the capacitor:

$$C = \frac{ConversionTime \times Current}{FullScale}$$

The Conversion Time is in seconds, the current value in μA and the Full Scale value is 3 Volts.

To ensure secure and stable measurements it is recommended to do several measurements on the same channel and mediating the obtained results.

The optimum linearity in conversion can be obtained if the voltage level on the selected input channel does not exceed a maximum of Vcc-1.5V

For more information see related Application Note.

Table 7.2 A/D Conversion Times

Timer Clock	A/D Resolution	Max Conversion Time	Full Scale	Current Source Value	Capacitor Value (Nearest Standard)
20 MHz	14 bits	0.8192 mS	3 V	30 μA	8.2 nF
20 Mhz	12 bits	0.2048 mS	3 V	30 μA	2.2 nF
20 Mhz	10 bits	51.2 μS	3 V	30 μA	0.47 nF
20 MHz	8 bit	12.8 μS	3 V	30 μA	0.1 nF

8 WATCHDOG TIMER

8.1 Functional Description

The Watchdog Timer (WDT) is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The WDT circuit generates an MCU reset on expiry of a programmed time period (Timeout), unless the program refreshes the WDT before the end of the programmed time period itself.

16 different time delays can be selected by using the WDT configuration register CONF_REG2 as in Table 8.2.

WDT is activated by the assembler instruction WDTRFR.

At the end of the programmed time delay, WDT starts a reset cycle pulling low the reset pin.

During normal operation, when WDT is active, the application program has to refresh this peripheral at regular intervals to prevent an MCU reset. WDT refresh is done by WDTRFR assembler instruction.

To stop WDT during the user program executions instruction WDTSLP has to be used.

WDT working frequency PRES CLK is equal to Master Clock frequency. It is divided by a fixed Prescaler with a division factor of 500, to obtain WDT CLK signal that is used to fix the WDT Timeout period (Figure 8.1).

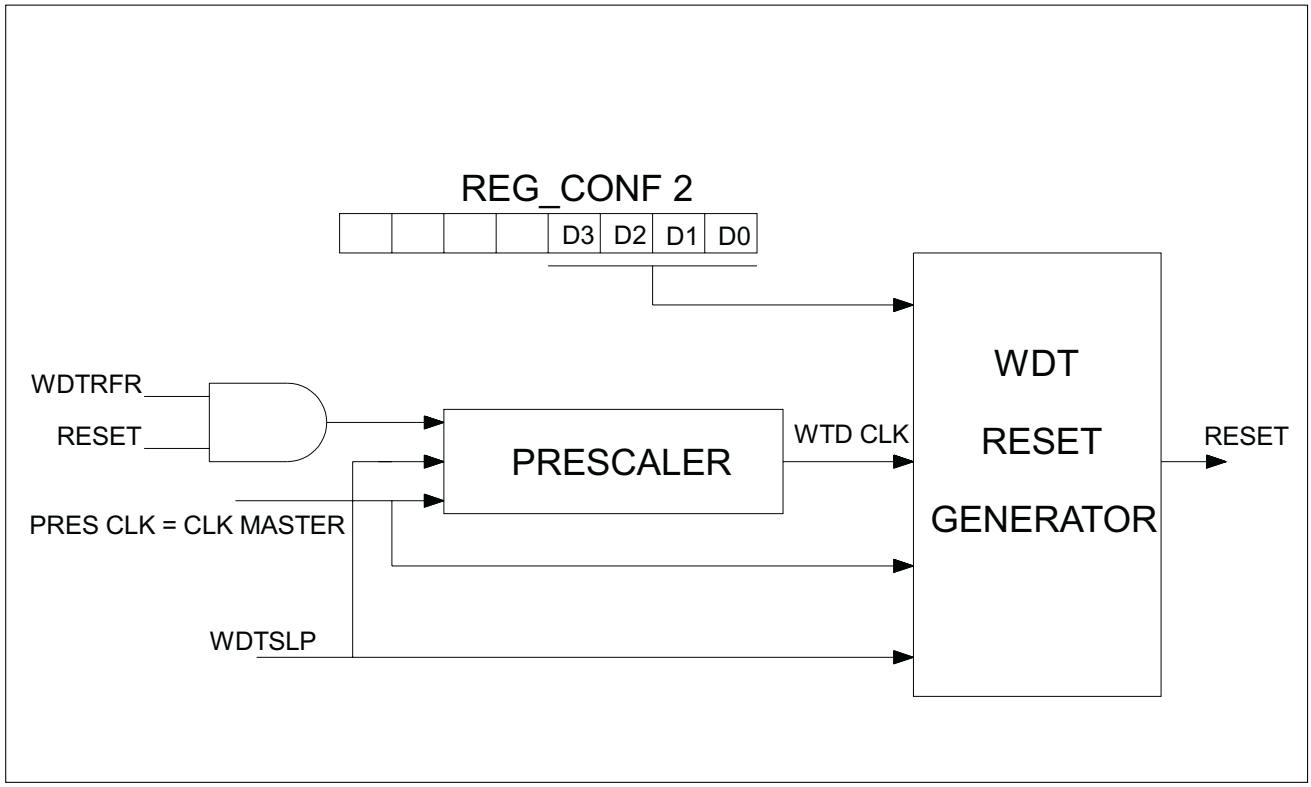
Table 8.1 Watchdog Timing range (CLK=20MHz)

	WDT Timeout period (ms)
min	0.025
max	234.375

With a Master Clock of 20MHz, for instance, it is possible to define a WDT Timeout period between 0.025ms and 234.375ms, depending on WDT CONF_REG2 values.

Timeout delay values at different Master Clock frequencies can be calculated as the product of WDT clock pulses number (Table 8.2) by WDT CLK period (Table 8.4).

Figure 8.1 Watchdog Block Diagram



8.2 Register Description

WDT Timeout period can be fixed setting the first 4 bits of REG_CONF2: this allows to define 16 different values of WDT Clock pulse number.

The WDT CLK is derived from the Master Clock divided by 500. The Timeout is then obtained by multiplying the WDT CLK period for the number of pulses defined in configuration register REG_CONF2. The Table 8.4 shows the pulses length for typical values of Master Clock.

The Table 8.3 shows the timeout WDT values when Master Clock is 5 MHz.

Table 8.2 WDT REG_CONF 2

Bit	Name	Value	Timeout Values (WDT CLK pulses)
0	D(3:0)	0000	1
		0001	625
		0010	1250
		0011	1875
1		0100	2500
		0101	3125
		0110	3750
		0111	4375
2		1000	5000
		1001	5625
		1010	6250
		1011	6875
3		1100	7500
		1101	8125
		1110	8750
		1111	9375
4-7	NC	x	Not Used
Reset Configuration ‘0000’			

Table 8.3 Timeout Values with CLK=5 MHz

Bit	Name	Value	Timeout Values (ms)
0	D(3:0)	0000	0.1
		0001	62.5
		0010	125
		0011	187.5
1		0100	250
		0101	312.5
		0110	375
		0111	437.5
2		1000	500
		1001	562.5
		1010	625
		1011	687.5
3		1100	750
		1101	812.5
		1110	875
		1111	937.5
4-7	NC	x	Not Used
Reset Configuration ‘0000’			

Table 8.4 Typical WDT CLK PERIOD

MASTER CLK (MHz)	WDT CLK (KHz)	WDT CLK PERIOD (ms)
4	0.8	0.125
5	1	0.1
8	1.6	0.0625
10	2	0.05
20	4	0.025

9 PWM/TIMER

ST52x440 on-chip PWM/TIMER peripherals consists of an 8-bit counter with a 16-bit programmable prescaler that provide a maximum count of 2^{24} (Figure 9.1).

The TIMER has two different working modes:

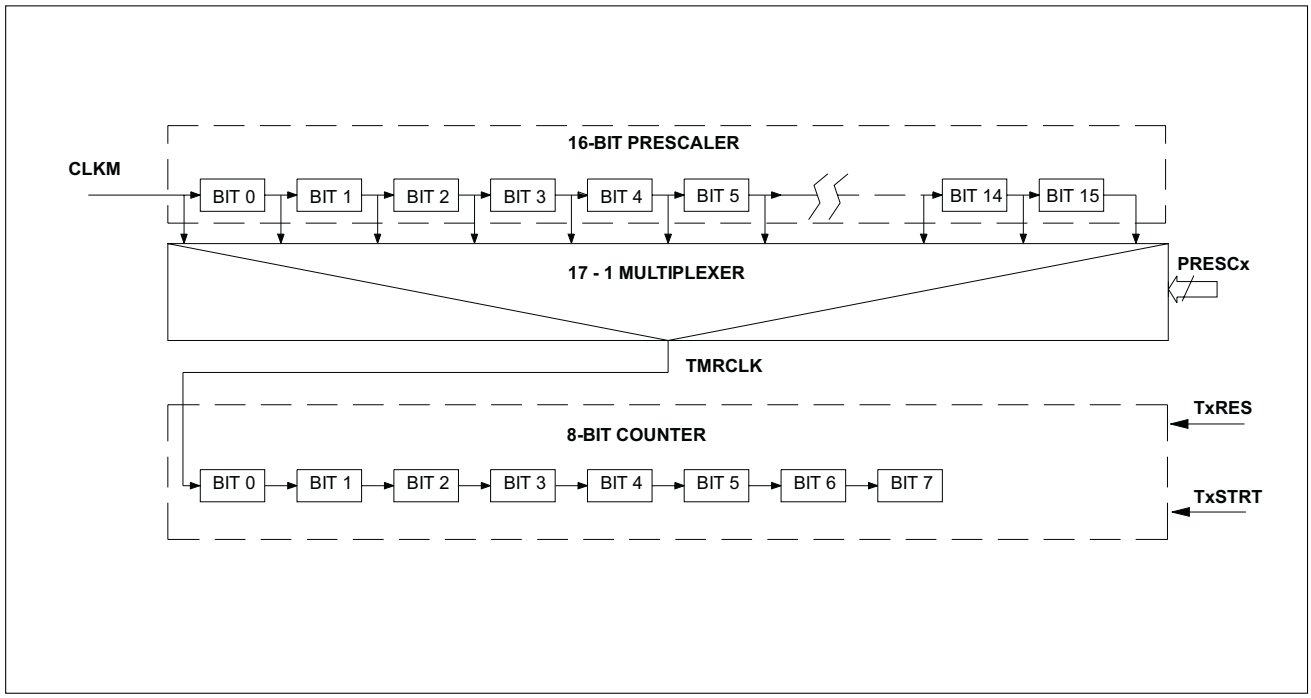
Timer Mode

PWM (Pulse Width Modulation) Mode

TMRCLKx is the Prescaler x output, that increments, on the rising edge, the Counter x value. TMRCLKx is obtained from the internal clock signal (CLKM) or, only for TIMER0, from the external signal provided on the PA5/T0CLK pin.

NOTE: The external clck signal, applied on T0CLK pin, must have a frequency at least two times smaller than the internal master clock.

Figure 9.1 Timer Peripheral Block Diagram



, that can be selected by setting register REG_CONF5[7] bit TxMODE.

The Timer has Autoreload Function in PWM Mode. Its output TOUT is available, with its negated signal TOUTN on external pins by setting PA6 and PA2 bits of REG_CONF4 and REG_CONF12 (see tables 6.4 and 6.5).

The TIMER can also use an external START/STOP signal (Input capture and Output compare), an external RESET and an external CLOCK signals: PA4/TSTRT, PA6/TRES and PA5/TCLK pins.

REMARKS: To use TSTRT, TRES, TCLK external signals the related pins PA4, PA6 and PA5 must be configured in Input Mode setting registers REG_CONF4 and REG_CONF12 (see table 6.4 and 6.5)

The content of the 8-bit counter of the TIMER is incremented on the Rising Edge of the 16-bit prescaler output (PRESCOUT) and it can be read at any instant of the counting phase then saved in a location of the RAM memory. The PWM/Timer Counter value can be read from the Input Register PWM_COUNT (Input Registers 18, see Table 2.1)

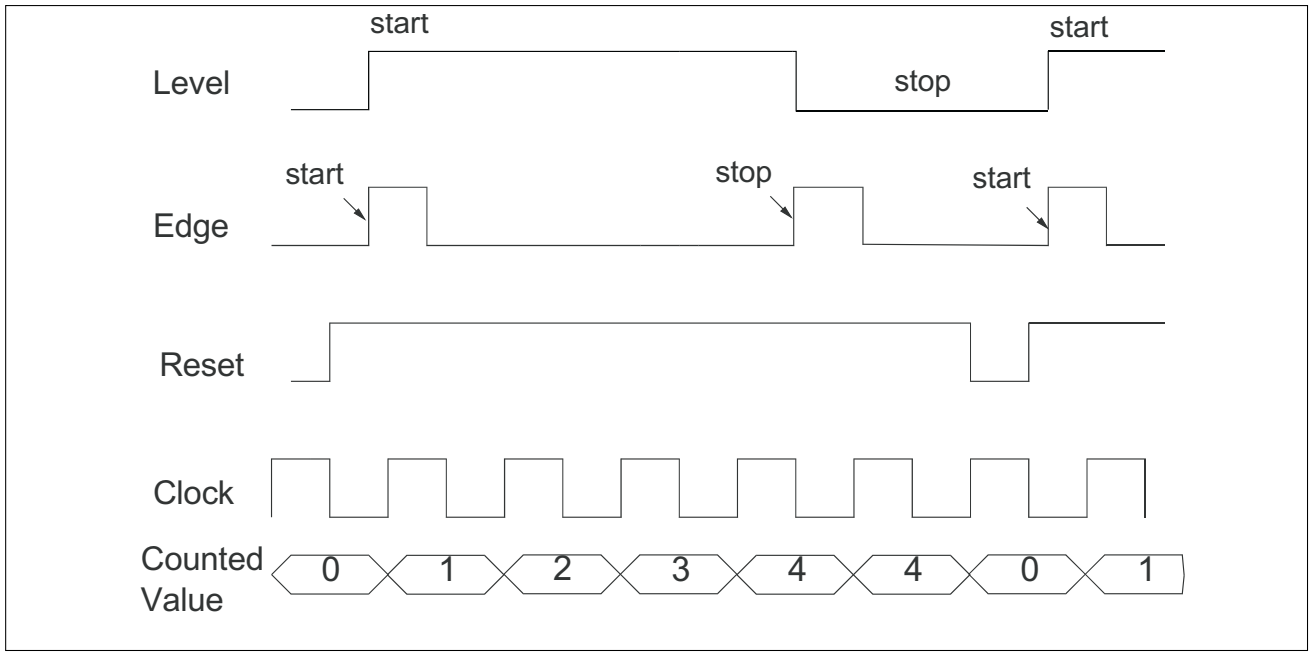
The PWM/Timer Status can be read from the Input Register PWM_STATUS (Input Registers 19. See Table 2.1).

9.1 Timer Mode

Timer Mode is selected fixing TxMODE bit of REG_CONF5[7].

The TIMER can receive as Inputs three signals: Timer Clock (TCLK), Timer Reset (TRES) and Timer Start (TSTRT) (Figure 9.1). Each of these signals can be generated internally or externally by setting TSTRT, TRES, TCLK bits of REG_CONF7 register as in Table 9.3.

Figure 9.2. Timer 0 External START / STOP Mode



The prescaler output can be selected by setting PRESC bit of REG_CONF6 register (Table 9.2).

TRES resets to zero the content of the TIMER 8-bit counter. It is generated by the TIRSTx and TMSK bits of REG_CONF5, REG_CONF7,

NOTE: The external clck signal, applied on T0CLK pin, must have a frequency at least two times smaller than the internal master clock.

The prescaler output can be selected by setting PRESC bit of REG_CONF6 (Table 9.2).

TRES resets to zero the content of the TIMER 8-bit Counter. It is generated by TIRST bit in REG_CONF5 register.

TSTRT signal starts and stops the Timer counting only if the peripheral is configured in Timer mode

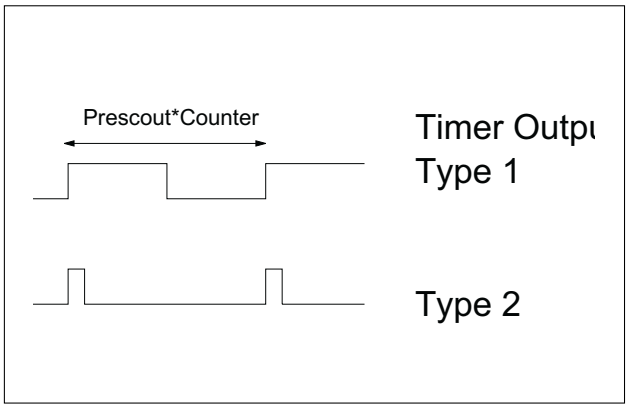
REG_CONF8 and REG_CONF10 registers (see tables 9.1, 9.3, 9.4 and 9.6).

TxSTRT signal start/stop the Timer x counting only if the peripherals are configured in Timer mode. This signal is forced by setting the correspondent TISTR bit of REG_CONF5.

TIMER START/STOP can be given from the external on TSTRT pin. In this case, TSTRT signal allows to work in two different modes, by setting the TESTR configuration bit of REG_CONF5 register (**Input capture**):

LEVEL (Time Counter): When TSTRT signal is high the Timer starts the count. When the TSTRT is low the counting is stopped and the current value is stored in the PWMCOUNT Input Register.

Figure 9.3. TIMEROUT Signal Type



EDGE(Period Counter): After the reset, on the first TSTRT rising edge, the TIMER starts the counting and, at next rising edge, it is stopped. In this way it is possible to measure the period of an external signal.

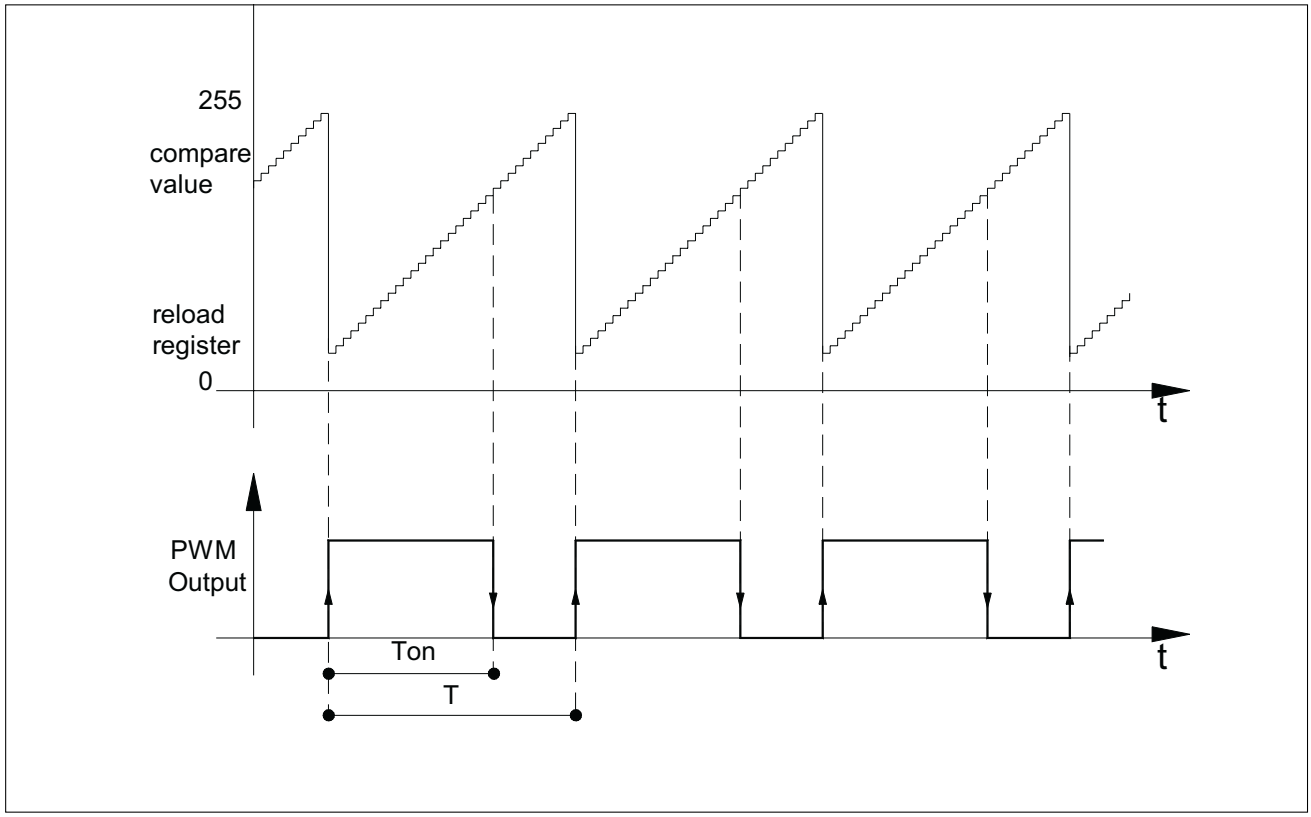
The Timer output signal, TIMEROUT, is a signal with a frequency equal to the 16 bit-Prescaler output signal, TMRCLK, divided by the Output Register PWM_COUNT value (8 bit) (Output Registers 9, Table 2.3), that is the value to count.

TIMEROUT waveform can be of two types:

type 1: TOUT waveform equal to a square wave with a 50% duty-cycle

type 2: TOUT waveform equal to a pulse signal with the pulse duration equal to the Prescaler output signal.

Fig. 9.4 PWM Mode with Auto Reload



The Timer output signal waveform type can be selected by setting the correspondent TMRW bit of REG_CONF6.

9.2 PWM Mode

The PWM working mode is obtained setting to "1" the correspondent TMODE bit of REG_CONF5.

TIMEROUT, in PWM Mode, consists of a signal, with a fixed period, whose duty cycle can be modified by the user.

TIMEROUT signal can be available on TOUT pin and TIMEROUT inverted signal can be available on TOUTN pin, setting the relative bits on PORT A, REG_CONF12[1] and REG_CONF12[2], to "0" and REG_CONF4[6] and REG_CONF4[2] to "0".

The PWM TIMEROUT period can be fixed setting the 16-bit prescaler output and an initial autoreload 8-bit counter value stored in the Output Register PWM_RELOAD, as shown in Figure 9.4.

The Output Register PWM_RELOAD value is automatically reloaded when the Counter restarts the count.

The 16-bit Prescaler divides the master clock, CLKM or the external signal TCLK. The Prescaler output can be selected setting PRESC bit of REG_CONF6.

NOTE: The external clock signal, applied on TCLK pin, must have a frequency at least two times smaller than the internal master clock.

When the Counter reaches the Peripheral Register PWM_COUNT value (**Compare Value**), the TIMEROUT signal changes from high to low level, up to the next counter start.

The period of the PWM signal is obtained by using the following relation:

$$T = (255 - PWM\_RELOAD) \times TMRCLK$$

where TMRCLK is the output of the 16-bit prescaler.

The duty cycle of the PWM signal is controlled by the Output Register PWM_COUNT:

$$Ton = (PWM\_COUNT - PWM\_RELOAD) * TMRCLK$$

If the Output Register PWM_COUNT value is 255 the TIMEROUT signal is always at high level.

If the Output Register PWM_COUNT is 0, or less than the PWM_RELOAD value, TIMEROUT signal is always at low level.

NOTE. If PWM_RELOAD value increases, the duty cycle resolution decreases.

By using a 20 MHz Master Clock it is possible to obtain a PWM frequency in the range 1.2 Hz to 78.43 Khz.

NOTE: The Timer, before using a new value of the counter or of the reload, has to complete the previous counting. If the counter/reload value is changed during a counting, the new value of the timer counter is used only at the end of the previous counting phase. This happens both in Timer and in PWM mode.

When the Timer is in Reset, or when the device is reset, the TOUT pin goes to threestate: it is recommended to put a pull-up or a pull-down resistor if this output is used to drive an external device.

9.3 Timer Interrupt

The TIMER can be programmed to generate an Interrupt request until the end of the count or when there is an external TSTRT signal. The Timer can generate programmable Interrupts into 4 different modes:

Interrupt mode 1: Interrupt on counter Stop.

Interrupt mode 2: Interrupt on Rising Edge of TIMEROUT.

Interrupt mode 3: Interrupt on Falling Edge of TIMEROUT.

Interrupt mode 4: Interrupt on both edges of TIMEROUT.

The Interrupt mode can be selected by means of INTSL and INTE bits of the REG_CONF5.

Table 9.1. Configuration Register 5 Description

Bit	Name	Value	Description
0	TIRST	0	Internal RESET
		1	Internal SET
1	TERST	0	External RESET on Level
		1	External RESET on Edge
2	TISTR	0	Internal STOP
		1	Internal START
3	TESTR	0	External START on Level
		1	External START on Edge
4	INTE	00	TIMER Interrupt on TIMEROUT Falling Edge
		01	TIMER Interrupt on TIMEROUT Rising Edge
5		10	TIMER Interrupt on Both Edges of TIMEROUT
		11	- not used
6	INTSL	0	TIMER Interrupt on Counter Stop
		1	TIMER Interrupt on TIMEROUT Edges
7	TMODE	0	TIMER MODE
		1	PWM MODE

Reset Configuration = "00000000"

Figure 9.5. Configuration Register 5

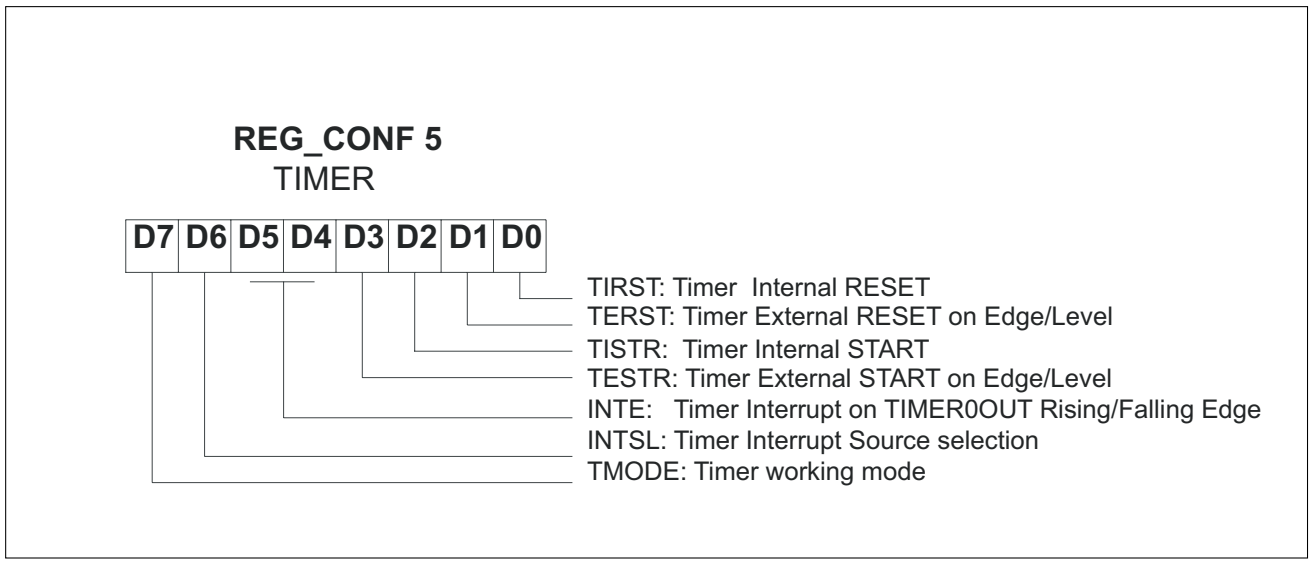


Table 9.2. Configuration Register 6 Description

Bit	Name	Value	Description
0	PRESC	00000	TIMER Clock = CLKM / 1
		00001	TIMER Clock = CLKM / 2
		00010	TIMER Clock = CLKM / 4
		00011	TIMER Clock = CLKM / 8
1		00100	TIMER Clock = CLKM / 16
		00101	TIMER Clock = CLKM / 32
		00110	TIMER Clock = CLKM / 64
2		00111	TIMER Clock = CLKM / 128
		01000	TIMER Clock = CLKM / 256
		01001	TIMER Clock = CLKM / 512
3		01010	TIMER Clock = CLKM/1024
		01011	TIMER Clock = CLKM/2048
		01100	TIMER Clock = CLKM/4096
4		01101	TIMER Clock = CLKM/8192
		01110	TIMER Clock=CLKM/16384
		01111	TIMER Clock=CLKM/32768
		10000	TIMER Clock=CLKM /65536
5	TMRW	0	TIMEROUT Pulse type waveform
		1	TIMEROUT Square type waveform
6	-	-	Not used

Reset Configuration = “00000000”

Figure 9.6. Configuration Register 6

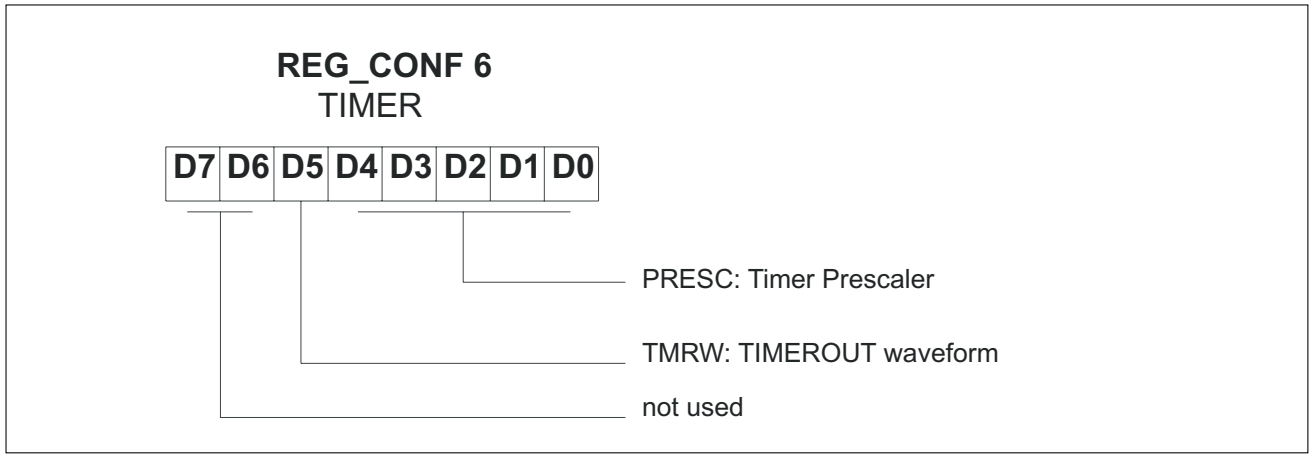
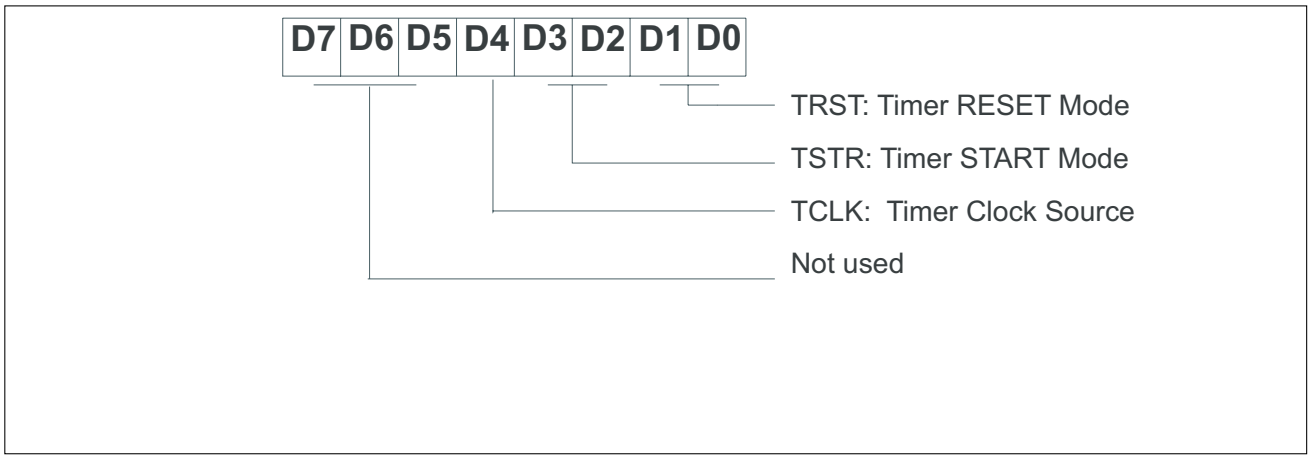


Table 9.3. Configuration Register 7 Description

Bit	Name	Value	Description
0	TRST	00	TIMER RESET Internal
		01	TIMER RESET External
1		10	TIMER RESET External or Internal
		11	Not used
2	TSTR	00	TIMER START Internal
		01	TIMER START External
3		10	TIMER START External or Internal
		11	Not used
4	TCLK	0	TIMER Clock Internal
		1	TIMER Clock External
5	NC	-	Not used
6	NC	-	Not used
7	NC	0	**Must be kept to "0"**

Reset Configuration = "00000000"

Figure 9.7. Configuration Register 7



10 TRIAC/PWM DRIVER

ST52x440 offers a peripheral able to generate a TRIACOUT signal on pin 26, to drive an external device, like a TRIAC, a IGBT or a Power Mos. Triac/PWM driver can perform 3 different working modes according to REG_CONF10(3:2) bits, MODE (see Table 10.4):

MODE = “00”: PWM

MODE = “01": Burst Mode Triac Control (Thermal Regulations)

MODE = “1x": Phase Angle Partialization Triac Control (Motor Control)

The Triac/PWM Driver can be initialized by using a value fixed by a control algorithm, that can be either the output of a fuzzy inference or the result of an arithmetic calculus stored in the Register File.

In the latter case, by using the `LDPR 9,reg-i` instruction, the value, contained in the i-th register of Register File, is stored in the Triac Driver/PWM peripheral register PERIPH_REG9.

Figure 10.1 shows the internal structure of THE Triac/PWM Driver.

PWM Mode

The PWM working mode selection can be obtained by setting REG_CONF10(3:2) bits, MODE, at “00" value.

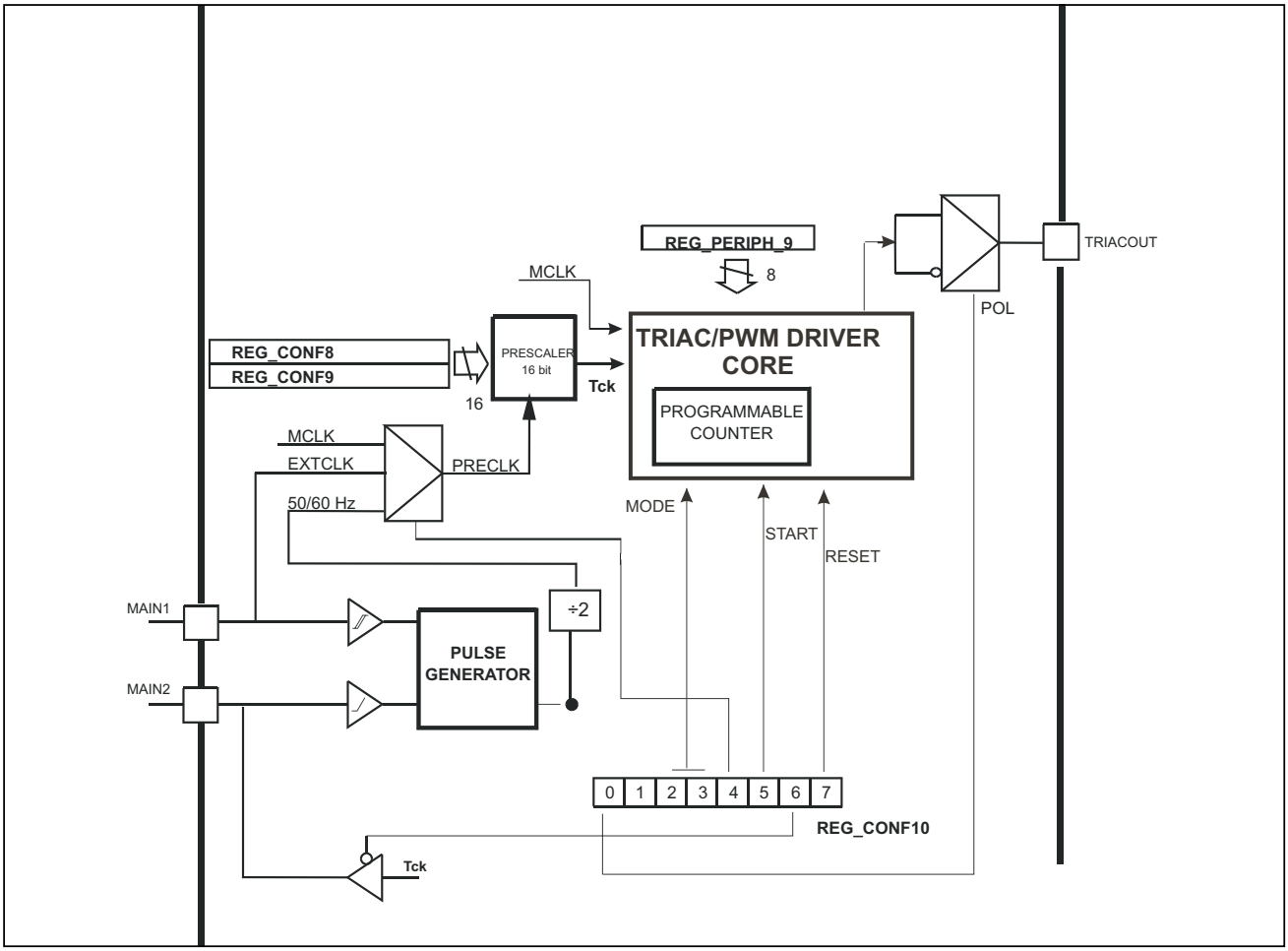
In this working mode, the peripheral provides on TRIACOUT pin a signal with a fixed period and a variable duty cycle.

The PWM period can be generated by dividing the internal master clock or by using an external clock signal.

In both cases, the clock signal is divided by a 16-bit Prescaler, managed by REG_CONF8 and REG_CONF9 (Figure 10.2).

At each period, the duty cycle is fixed by a value, that can be either the output of a fuzzy inference or the result of an arithmetic calculus. In both cases

Figure 10.1 TRIAC/PWM Driver Simplified Block Diagram



this value, that we indicate as INIT_VALUE, must be loaded directly in PERIPH_REG_9.

Burst Mode

The Burst principle is based on turning on and off the TRIAC device for a fixed integer number of mains voltage periods, in order to control the power transferred to the load.

In Burst Mode the peripheral provides a signal, with a fixed period T containing an integer number of pulses in correspondance of the main voltage zero crossings, with Duty Cycle proportional to the number of pulses that keep the TRIAC ON (Figure 10.4).

The user can define the period T by means of the internal 16-bit prescaler, setting REG_CONF8 and REG_CONF9 (Figure 10.2). T is proportional to the main voltage period and is in the range [5.10, 334233.6]s if mains frequency is 50Hz.

The duty cycle is fixed at each period by a value that can be directly the output of a fuzzy inference or the result of an arithmetic calculus.

The width and the polarity of the pulses can be programmed according to the Triac device and the circuit characteristics.

In order to work in Burst mode, it is necessary to detect the pre-post zero-crossing of main voltage, by using an external inserting circuitry.

This kind of Triac control is mainly used for thermal regulation.

Phase Angle Partialization Mode

Phase Angle Partialization method is based on turning on the TRIAC device only for a part (Phase Angle) of each main voltage period. When the phase angle is large the energy (power) supplied to the load is low, viceversa when the phase angle is small the energy supplied to the load is high.

The phase angle can be calculated by a fuzzy algorithm or by an arithmetical algorithm and its value is stored in PERIPH_REG_9.

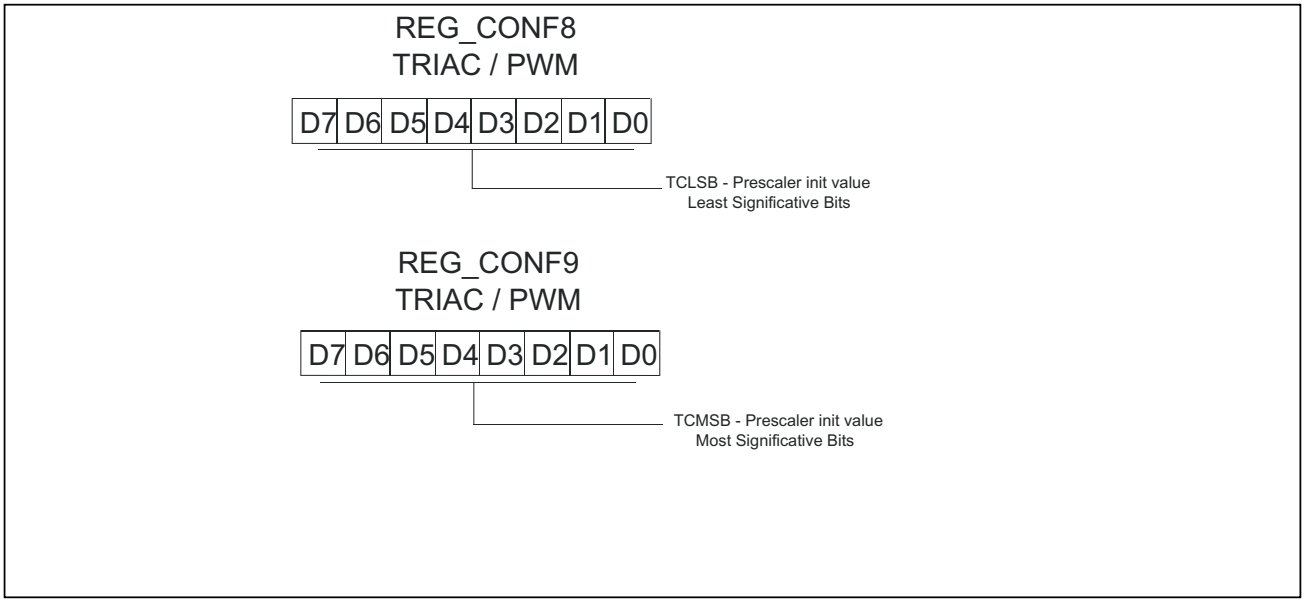
10.1. TRIAC/PWM DRIVER PROGRAMMING

In all three working mode, it is possible to SET or RESET the TRIAC/PWM Peripheral through REG_CONF10(7) bit TCRST (Table 10.4).

If TRIAC/PWM Peripheral is SET, it is possible to START or STOP the internal counter of the peripheral without resetting it, through REG_CONF10(5) bit TCST (Table 10.4).

IF TCTRS is 0 the TRIAC/PWM Peripheral output is in tristate status.

Figure 10.2 TRIAC/PWM Configuration Registers 8 and 9



10.2 PWM GENERATOR WORKING MODE

When REG_CONF10 (3:2) bits MODE are “00", the peripheral is programmed to work in PWM Mode.

By using the 16-bit prescaler, the PWM period can be generated by dividing the internal master clock, an external clock signal applied to pin MAIN1, or the mains voltage frequency, using circuit of Figure 10.6.

NOTE: The external clock signal, applied on MAIN1 pin, must have a frequency at least two time smaller than the internal master clock.

The clock source can be selected by using REG_CONF10(4) bit, CKSL (Table 10.4 and Figure 10.9).

The period T of the PWM signal **Tb** can be calculated from the following relation:

$$T = 255 * Tck$$

Table 10.2 PWM Frequencies

MCLK Frequencies	1/T	
	min	max
5 MHz	0.3 Hz	78.4 KHz
10 MHz	0.6 Hz	39.2 KHz
20 MHz	1.2 Hz	19.6 KHz

where Tck is the output of the 16-bit prescaler managed by REG_CONF8 and REG_CONF9 (Figure 10.2).

NOTE. In PWM working mode, the value N, stored in the 16-bit prescaler, must be in the range from 0 to 2^{16}-1

By using a 20 MHz clock master it is possible to obtain a PWM frequency in the range 1.2 Hz to 78.4KHz.

The value Ton is proportional to INIT_VALUE as from the relation:

$$Ton = INIT\_VALUE * Tck$$

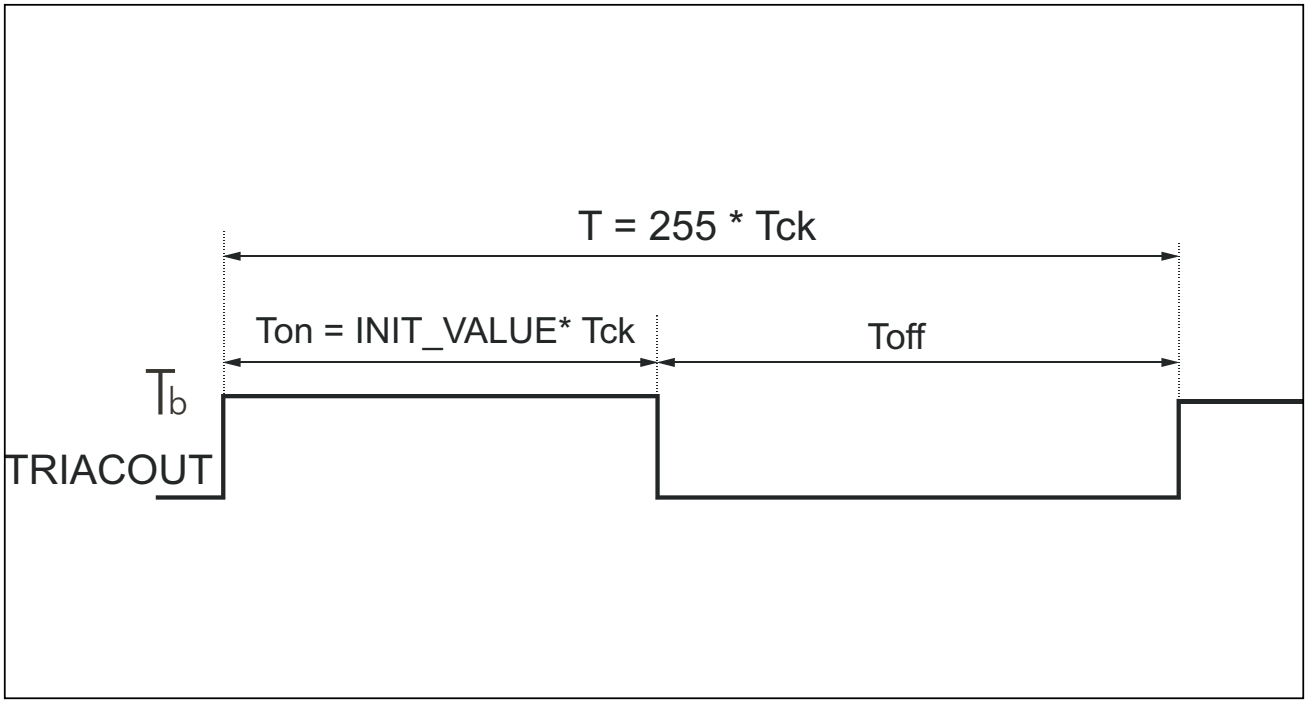
Ton can be the result of an arithmetical or a fuzzy algorithm, and can be stored in PERIPH_REG_9 from the *i-th* location of the Register File by using the instruction:

```
LDPR9, reg-i
```

INIT_VALUE can be changed on fly. If INIT_VALUE is 255 then Toff is zero and TRIACOUT is always equal to one during the period T.

IN PWM mode, it is possible to generate a programmable Interrupt in four different ways:

Figure 10.3 PWM Working Mode



1) No Interrupt;

2) Interrupt on rising edge of the signal Tb (INT_R).

3) Interrupt on falling edge of the signal Tb (INT_F)

4) Interrupt on both edges of the signal Tb.

The Interrupt is programmable through register REG_CONF0(5:3) bits, INTSL (Table 10.5).

NOTE: If the Interrupt on the rising edge (INT_R) of the pulse is not masked through REG_CONF0(4), the first Interrupt after the START_PWM occurs with a delay of a time period T.

NOTE: If INIT_VALUE is 255 or 0, the first interrupt after the START_PWM (either INT_R and INT_F) occurs at time T.

In any case for INIT_VALUE equal to 255 or 0, INT_R and INT_F coincide.

10.3 BURST MODE

When REG_CONF10(3:2) bits, MODE, are "01" the peripheral is programmed to work in BURST MODE.

Notice that when working in Burst mode, the synchronization with the mains is mandatory, therefore REG_CONF10(4) = **CKSL must be set to "1"**. (Table 10.4)

A square wave Tb is generated with a duty cycle proportional to the power the user needs to transfer to the load. A pulse is generated for each zero crossing of the mains voltage included in the Ton of the fixed period T. Figure 10.4 shows the typical Burst Control working mode. The period T of the signal Tb (Figure 10.4) is:

$$T = 255*Tck$$

The signal Tck is generated programming the 16-bit Prescaler by REG_CONF8 and REG_CONF9 (Figure 10.2). Tck is equal to the mains voltage frequency (50 or 60 Hz) divided by N+1, where N is an integer value in the range [0, 2^{16}-1].

The value Ton is proportional to a value INIT_VALUE that can be the result of a fuzzy or an arithmetical algorithm and can be stored in PERIPH_REG_9 from the i-th Register by using the instruction:

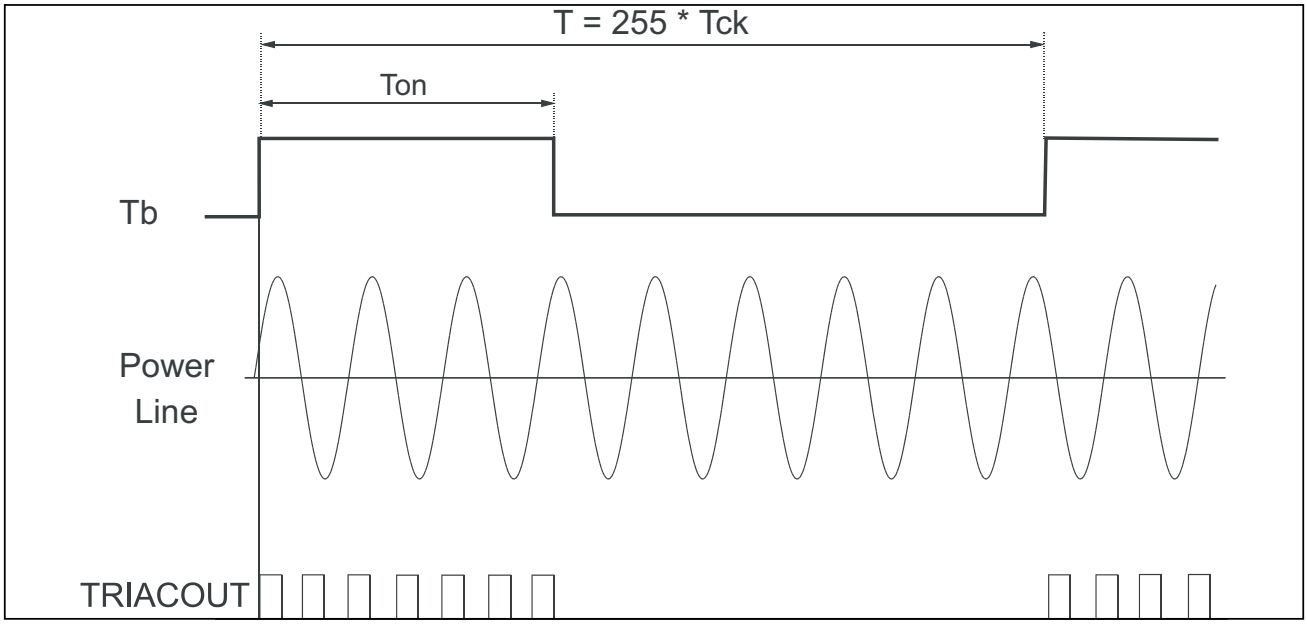
```
LDPR9, reg-i
```

The number of generated pulses N_PULSES is:

Table 10.3 TRIACOUT Signal Period

Power Line Frequency	T	
	min	max
50 Hz	5.10 s	334233.60 s
60 Hz	4.25 s	278528.05 s

Fig.10.4 - Burst Working Mode



$$N\_PULSES = 2*[(N+1)*INIT\_VALUE]$$

where N is the value stored in the 16-bit prescaler. Therefore it is:

$$Ton = INIT\_VALUE*Tck$$

The INIT_VALUE can be changed on fly and takes effect from the following period; the Prescaler value N instead is fixed since the beginning and cannot be changed on fly.

The first pulse is obtained during the first zero crossing of the main voltage and the last one is generated after clock pulses included in the time period INIT_VALUE*Tck, where Tck is the Prescaler output, generated by using the main voltage frequency applied to MAIN1 and MAIN2 pins. This guarantees synchronization with the mains voltage frequency.

Ranges of the Tb signal period depend on the power line frequency (Table 10.3).

In order to drive a Triac in Burst Mode it is required to generate a sequence of pulse, that must be centered on the zero crossing of the power line as shown in the Figure 10.7. For this reason, the pre

Figure 10.5 - Burst Mode Pulse Polarity

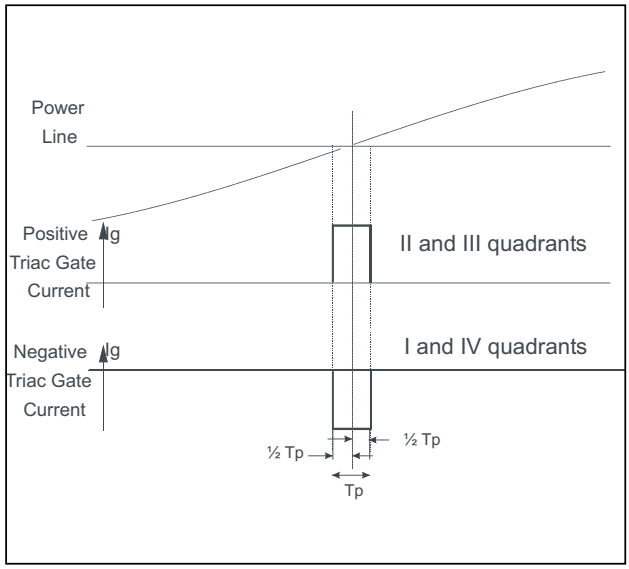


Figure 10.7 -Burts Mode Zero Crossing

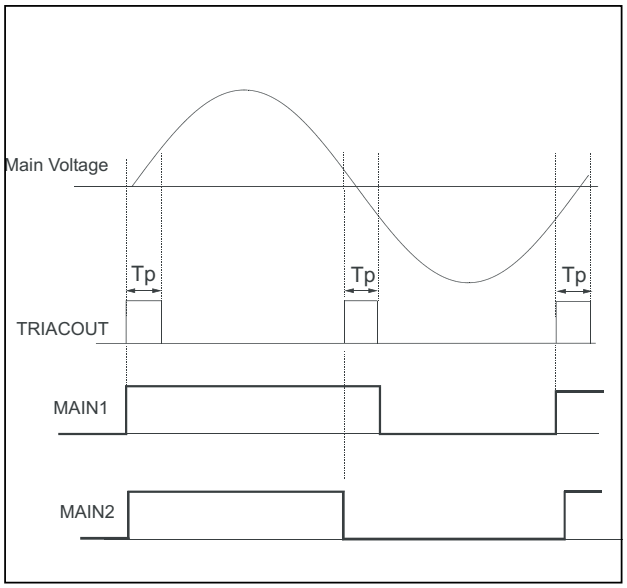
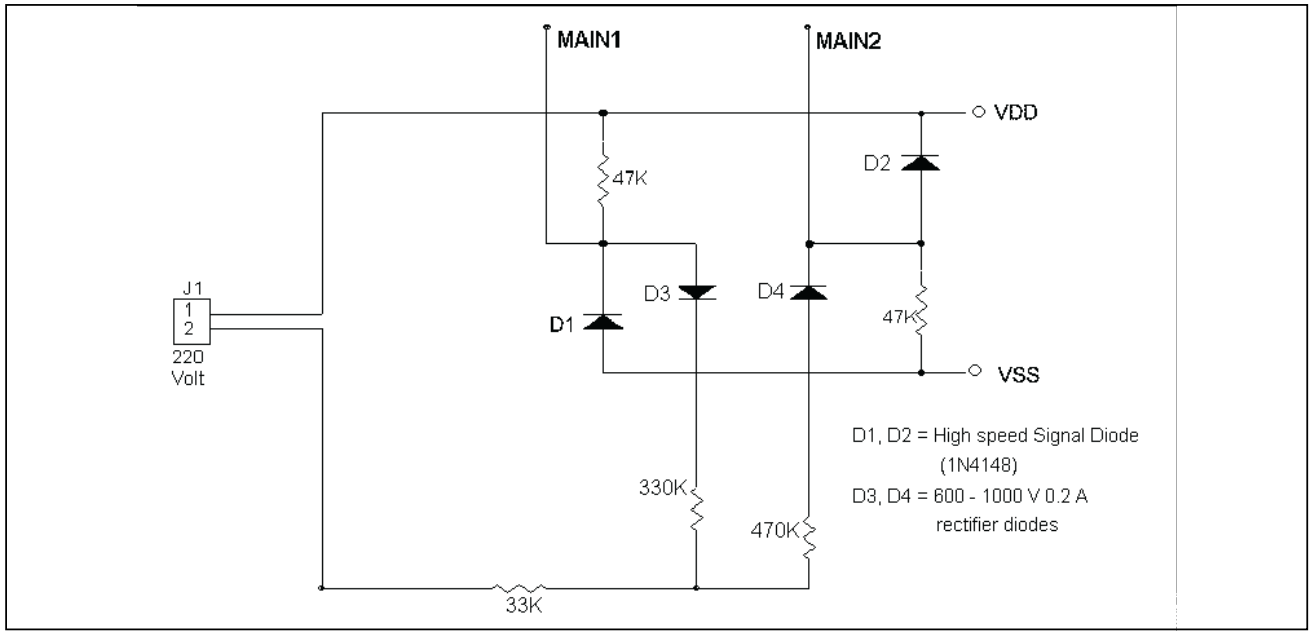


Figure 10.6 - Burst Mode Zero Crossing Circuit



zero crossing and the post zero crossing of the power line must be detected.

To detect the zero-crossing and get also the main voltage frequency, the user must generate MAIN1 and MAIN2 signals, by using the circuit shown in Figure 10.6.

MAIN1 and MAIN2 signals are used in the block called PULSE GENERATOR of the peripheral (see Figure 10.1).

In particular the pulses are generated by using the rising edge of the signal MAIN1 and the falling edge of the signal MAIN2.

Figure 10.5 shows the generation of the Triac pulses Tp .

The first firing pulse for the Triac is generated on the zero crossing of the power line, while the next pulses are centred on the zero crossing.

Generally the Triac firing pulses start 1/2 Tp before the zero crossing and the length of the pulses is Tp, see Figure 10.5.

The length Tp of the pulses is programmable by using a 16 bit value UTP, obtained with REG_CONF19 bits, UTPMSB, and REG_CONF20, UTPLSB (see figure 10.12 and table 10.6):

$$UTP(15:0) = [UTPMSB(7:0)\ UTPLSB(7:0)]$$

$$T_P = T_{MCLK} * UTP$$

The value Tp is in the range [0, 3.2] *ms* when the clock master is 20 MHz.

According to REG_CONF10(0) configuration register bit, POL, it is possible to set the firing pulses polarity; in order to obtain positive or negative gate Triac currents, allowing to work respectively in I and IV quadrants, or in the II and III quadrants (see Figures 10.5 and 10.12).

The pulses polarity can be changed on fly with immediate effect.

The pulses width and the Prescaler value N instead are fixed at the beginning and cannot be changed on fly.

Working in the II and III quadrant the peripheral implements the following procedure:

1) The firing pulse is set to “1" on the rising edge of MAIN1.

2) The firing pulse is reset to “0" after the time Tp fixed by program.

3) On the falling edge of MAIN2 the firing pulse is set to “1"

4) The firing pulse is reset to “0" after the time Tp fixed by program.

It is possible to generate a programmable Interrupt in four different ways:

1) No Interrupt;

2) Interrupt on the rising edge of the signal Tb (INT_R)

3) Interrupt on the falling edge of the signal Tb (INT_F)

4) Interrupt on both edges of the signal Tb.

5) Interrupt on risign edge of each Triac pulse (INT_P)

INT_R and INT_F interrupt signals are pulses with a duration of one clock master period. After START_PWM, INT_R and INT_F behave as described for the PWM working mode.

If POL=”1”, INT_P is generated on the falling edge of each pulse. It has a duration of 50 ns. If pulse width and INIT_VALUE are set to zero, therefore no TRIACOUT variation occurs, INT_P remains to zero.

Interrupts are programmable by using the register REG_CONF0(5:3), INTSL (see Table 10.5).

10.4 PHASE ANGLE PARTIALIZATION WORKING MODE

When REG_CONF10 (3) bit, MODE, is “1" the peripheral is programmed to work in PHASE ANGLE PARTIALIZATION mode.

In this mode Triac is controlled each semi-period of the mains voltage. The power transferred to the load is proportional to the CURRENT FLOW ANGLE γ. This kind of Triac control is suitable to drive the Triac with inductive load (i.e. universal or monophase motors). Figure10.8 shows the relation between the Phase Angle α and the Current Flow Angle γ .

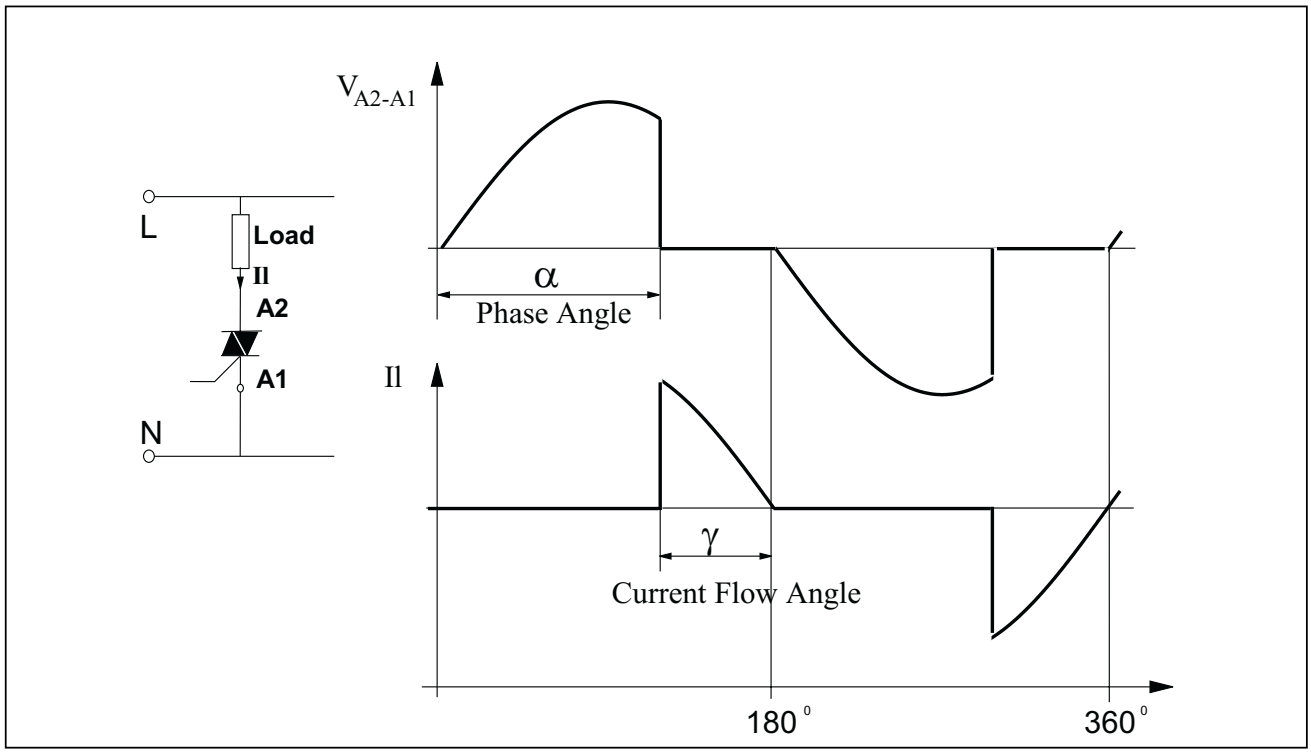
The peripheral allows to control the Phase Angle or equivalently time T1 (see Figure 10.9). It is possible to change Time T1 setting the contents of the peripheral register PERIPH_REG_9.

T1 is proportional to a value, INIT_VALUE, that can be a fuzzy algorithm output or a value coming from an arithmetical algorithm and can be stored in in PERIPH_REG_9 from the i-th Register by using the instruction:

```
LDPR9, reg-i
```

It is possible to use different circuits for the zero crossing detection, but MAIN1 signal rising edge must be synchronized with the mains voltage zero crossing and MAIN2 signal falling edge must be synchronized with the following mains voltage zero crossing, always.

Figure 10.8 - Phase angle Partialization Mode



Using the external circuit shown in Figure10.10, it is possible to use only one synchronization signal from the mains, MAIN1. In this case REG_CONF10(6) must be set to "1", MAIN2 signal coincides internally with MAIN1 and MAIN2 pin is left free for other functions.

If main voltage frequency is equal to 50 Hz, then Tr is equal to 20 ms (Figure 10.9) and T1 is:

$$T1 = INIT\_VALUE * T_{Mck} * (N+1)$$

with T_{Mck} master clock period. It is user task to verify that time T1 is not larger than a fixed time Tmax (8ms at 50 Hz) to avoid the firing of the Triac in the second half period of the mains voltage and to choose a suitable Prescaler value to avoid the shifting of the pulse sequence in the following semi-period.

In order to avoid problems for the Triac firing when the load is inductive 8 different pulses are generated by the peripheral (Figure10.12). Their WIDTH, equal the semiperiod Ti/2, is programmable by using registers REG_CONF19 and REG_CONF20 and is given by the relation:

$$Ti/2 = T_{Mck} * WIDTH$$

The choice of the pulse WIDTH must be done by the user paying attention to the fact that the duration of the 8 pulses train must be such that added to T1, it does not fall into the second half period of the mains voltage. In fact by using a clock master equal to 20 MHz and the full 16 bit value by ConfReg19 and 20, the pulse width would be in the range [0.2, 3.28] ms.

The duty cycle of Ti is always 50%.

The choice of the pulse WIDTH value must be done according to TRIAC device specifics and

Figure 10.9 - Phase Angle Partialization Mode

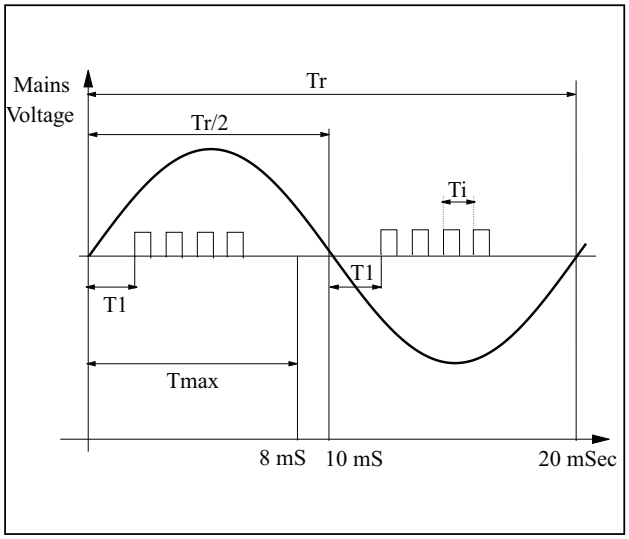
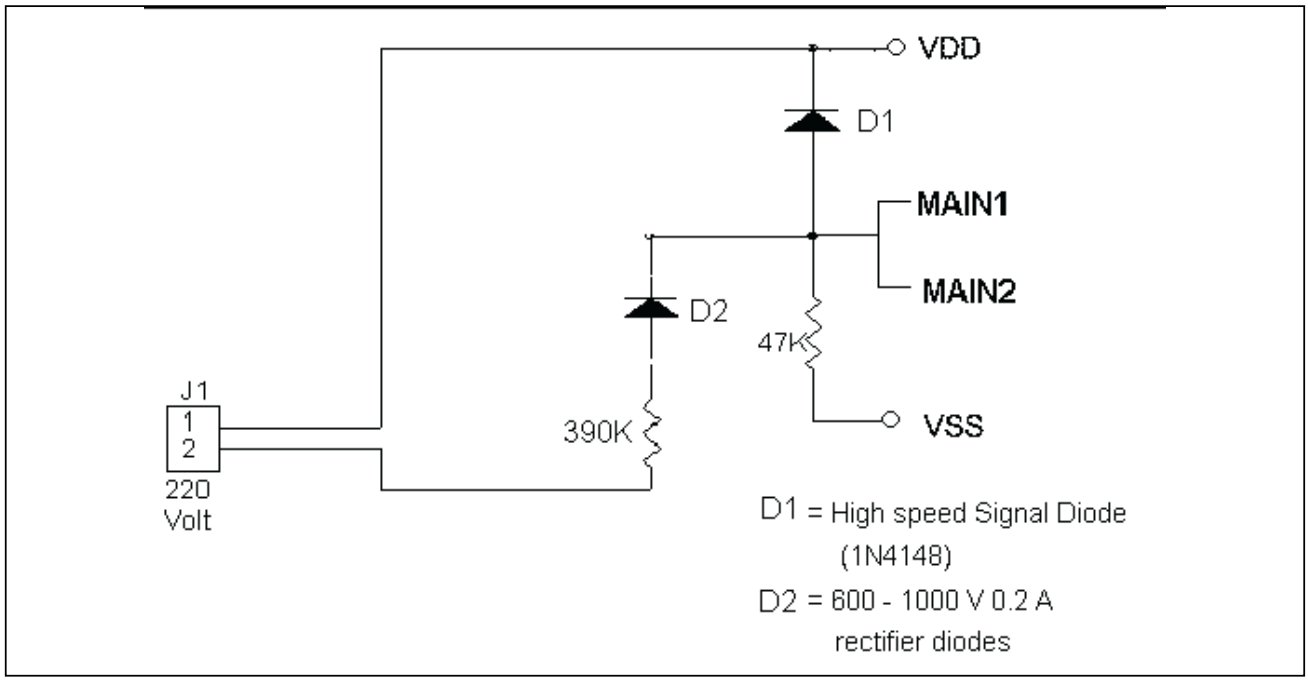


Figure 10.10 - Phase Angle Partialization Zero Crossing Circuit



must be fixed since the program beginning. To change WIDTH during program exectuion it is necessary to RESET the peripheral.

It is possible to generate a programmable interrupt in four different ways:

1) no Interrupt;

2) Interrupt on the rising edge of the signal MAIN1 (INT_R)

3)Interrupt on the falling edge of the signal MAIN2 (INT_F)

4) Interrupt on both the edges of the signal MAIN1

5) Interrupt on rising edge of first pulse after T1 (INT_P)

All Interrupt signals are pulses of duration 50 ns.

If WIDTH is 0, TRIACOUT remains at 0 (or 1, if POL=1), however after the time T1, the interrupt INT_P is generated.

The Interrupt is programmable by using the register REG_CONF0(5:3) INTSL.(Figure10.12)

Table 10.4 Configuration Register10

Bit	Name	Value	Description
0	POL	0	Set Positive Outpu t Pulse Polarity
		1	Set Negative Output Pulse Polarity
1	-	-	Not used
2	MODE	00	PWM Mode
		01	Burst Mode
3		1X	Phase Partialization Mode
4	CKSL	0	Internal Clock Master
		1	External Clock on Main1
5	TCST	0	Triac Stop
		1	Triac Start
6	IOSL	0	Set MAIN2 as Alternate Function Input on pin PA2
		1	MAIN2 coinciding MAIN1
7	TCRST	0	Triac Reset
		1	Triac Set

Reset Configuration = “00000000”

Table 10.5 Configuration Register 19

Bit	Name	Value	Description
0 - 7	UTPMSB		Output Impulse Width most significative bit

Reset Configuration = "00000000"

Table 10.6 Configuration Register 20

Bit	Name	Value	Description
0 - 7	UTPLSB		Output Impulse Width least significative bit

Reset Configuration = "00000000"

Figure 10.11 TRIAC/PWM Configuration Registers 10

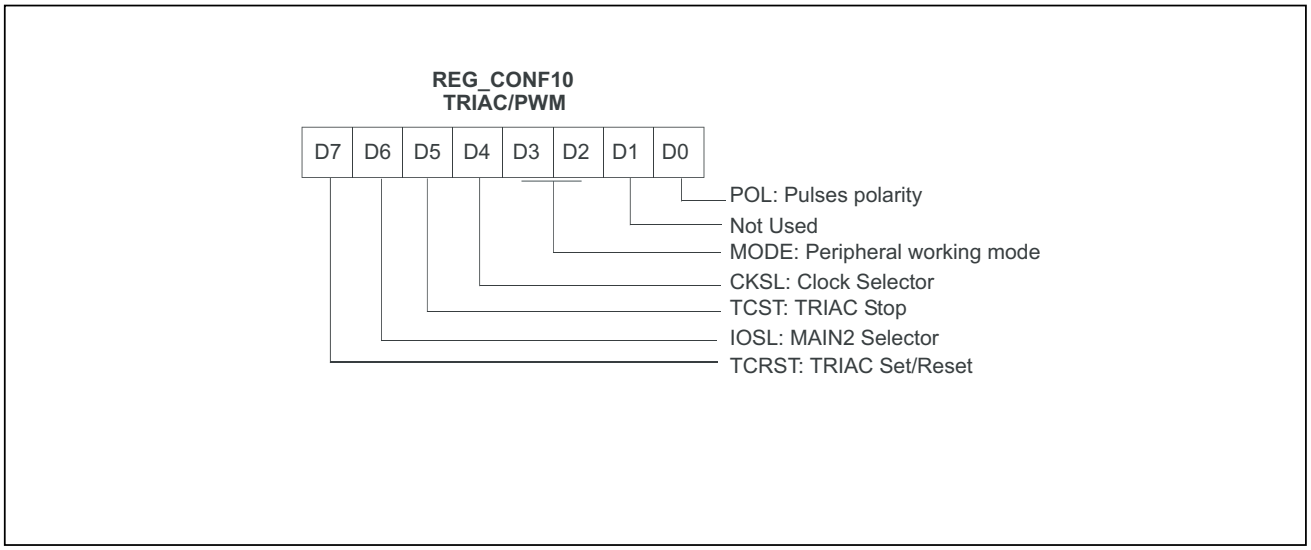
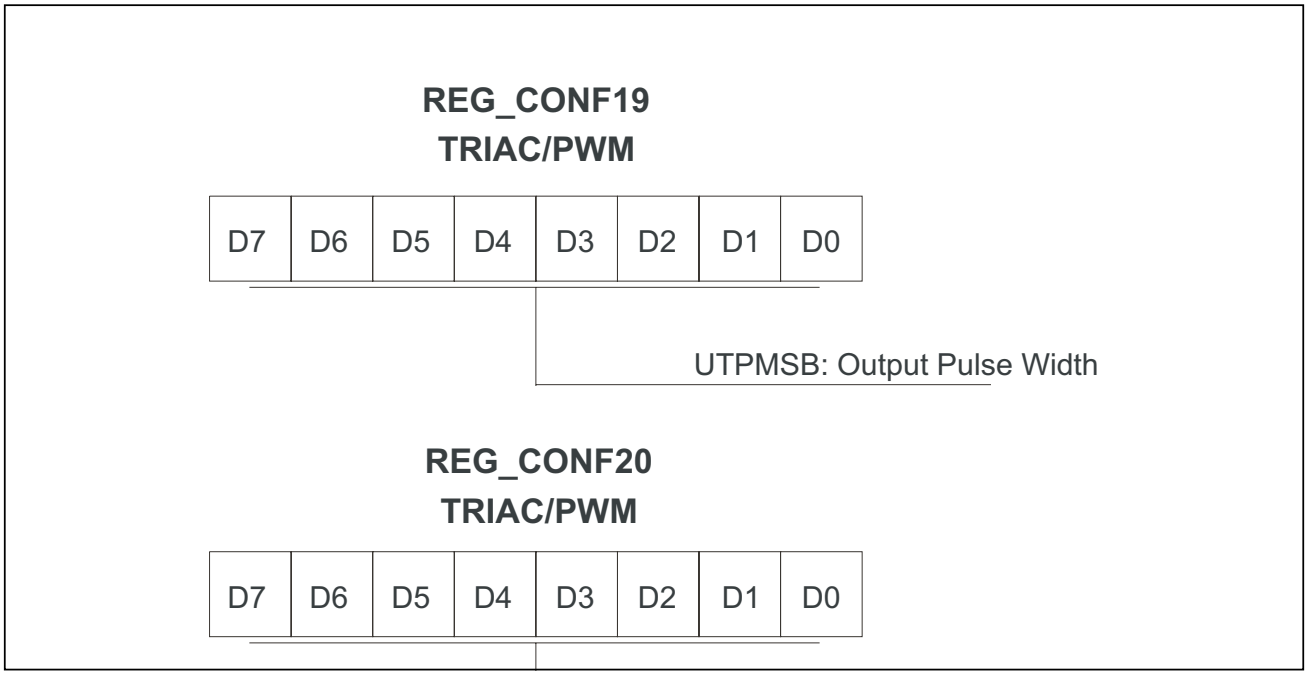


Figure 10.12 TRIAC/PWM Configuration Registers 19 and 20



ORDERING INFORMATION

Each device is available for production in user programmable version (OTP) as well as in factory programmed version (FASTROM). OTP device are shipped to customer with a default blank content FFh, while FASTROM factory programmed parts contain the code sent by customer. There is one common EPROM version for debugging and prototyping which features the maximum memory size and peripherals of the family. Care must be taken to only use resources available on the target device.

Figure 12.1 User Programmable Device Types

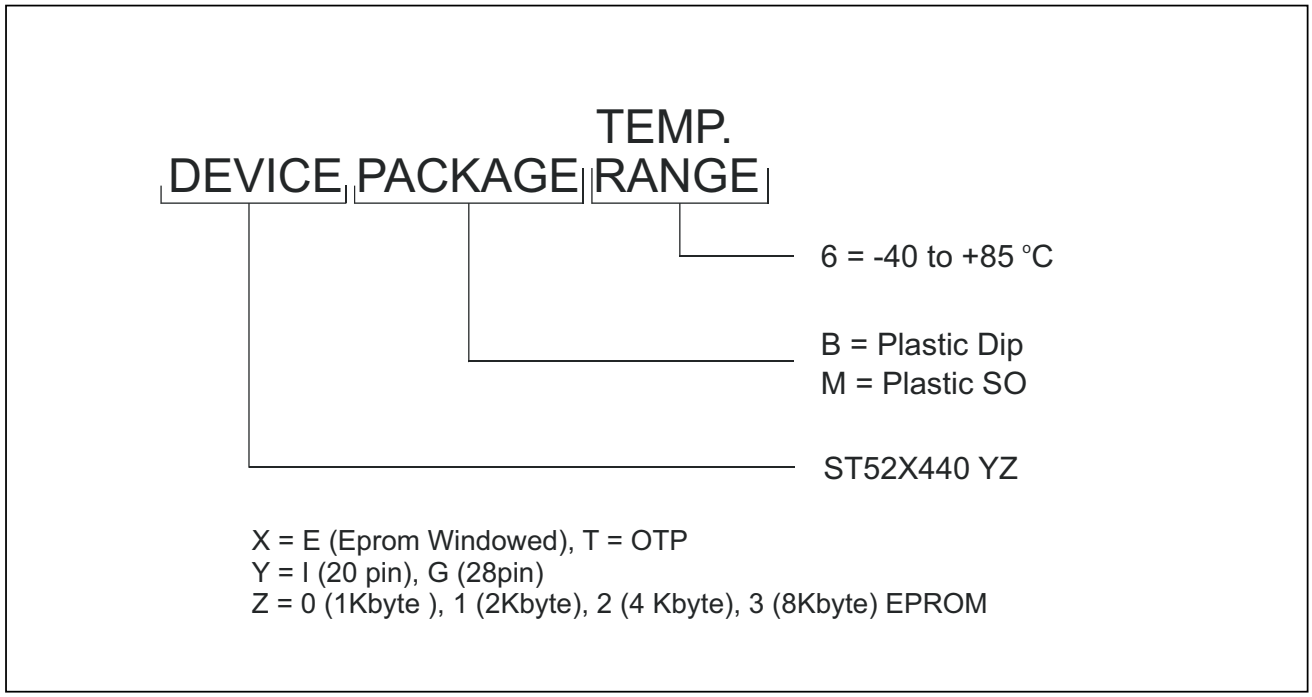


Table 12.1 ST52x440 family Part Numbers

PART NUMBER	TEMPERATURE RANGE	PACKAGE
ST52T440I0M6	-40TO +85 °C	SO20
ST52E440I1M6	-40TO +85 °C	SO20
ST52E440I2M6	-40TO +85 °C	SO20
ST52E440I3M6	-40TO +85 °C	SO20
ST52E440G0M6	-40TO +85 °C	SO28
ST52E440G1M6	-40TO +85 °C	SO28
ST52E440G2M6	-40TO +85 °C	SO28
ST52E440G3M6	-40TO +85 °C	SO28
ST52X440/KIT		DEVELOPMENT KIT