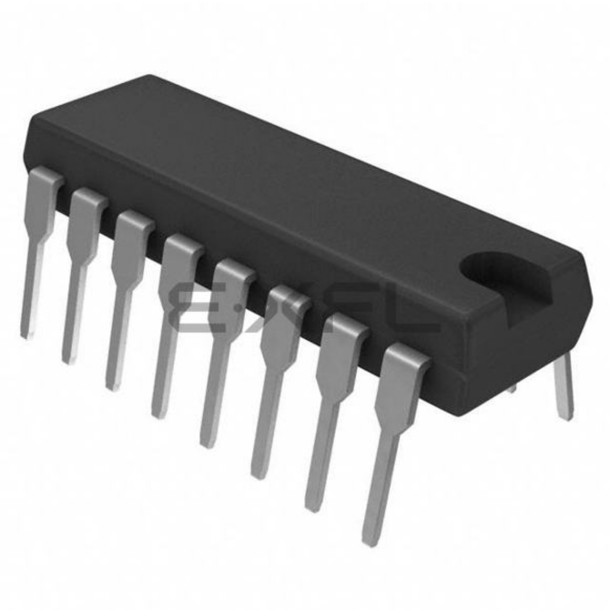


What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	13
Program Memory Size	1KB (1K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	16-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7flites2y0b6

Table of Contents

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
						Input				Output			
QFN20	SO16/DIP16			Input	Output	float	wpu	int	ana	OD	PP		
15	14	PA2/ATPWM0	I/O	C_T	HS	**X**	X			X	X	**Port A2**	Auto-Reload Timer PWM0
16	15	PA1	I/O	C_T	HS	**X**	X			X	X	**Port A1**	
17	16	PA0/LTIC	I/O	C_T	HS	**X**	ei0			X	X	**Port A0**	Lite Timer Input Capture

Note:

In the interrupt input column, "ei_x" defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is pull-up interrupt input, else the configuration is floating interrupt input.

REGISTER AND MEMORY MAP (Cont'd)

Legend: x=undefined, R/W=read/write

Table 2. Hardware Register Map

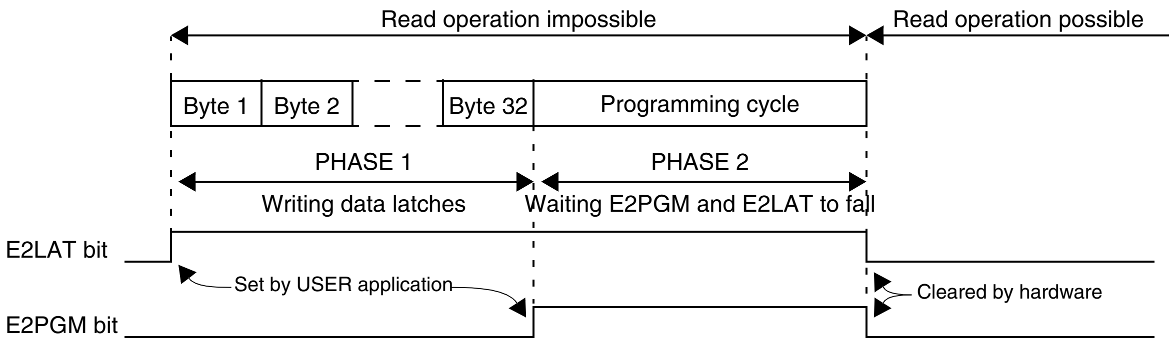
Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h 0002h	Port A	PADR PADDR PAOR	Port A Data Register Port A Data Direction Register Port A Option Register	00h 1) 00h 40h	R/W R/W R/W
0003h 0004h 0005h	Port B	PBDR PBDDR PBOR	Port B Data Register Port B Data Direction Register Port B Option Register	E0h 1) 00h 00h	R/W R/W R/W 2)
0006h to 000Ah	Reserved area (5 bytes)				
000Bh 000Ch	LITE TIMER	LTCSR LTICR	Lite Timer Control/Status Register Lite Timer Input Capture Register	xxh xxh	R/W Read Only
000Dh 000Eh 000Fh 0010h 0011h 0012h 0013h	AUTO-RELOAD TIMER	ATCSR CNTRH CNTRL ATRH ATRL PWMCR PWM0CSR	Timer Control/Status Register Counter Register High Counter Register Low Auto-Reload Register High Auto-Reload Register Low PWM Output Control Register PWM 0 Control/Status Register	00h 00h 00h 00h 00h 00h 00h	R/W Read Only Read Only R/W R/W R/W R/W
0014h to 0016h	Reserved area (3 bytes)				
0017h 0018h	AUTO-RELOAD TIMER	DCR0H DCR0L	PWM 0 Duty Cycle Register High PWM 0 Duty Cycle Register Low	00h 00h	R/W R/W
0019h to 002Eh	Reserved area (22 bytes)				
0002Fh	FLASH	FCSR	Flash Control/Status Register	00h	R/W
00030h	EEPROM	EECSR	Data EEPROM Control/Status Register	00h	R/W
0031h 0032h 0033h	SPI	SPIDR SPICR SPICSR	SPI Data I/O Register SPI Control Register SPI Control/Status Register	xxh 0xh 00h	R/W R/W R/W
0034h 0035h 0036h	ADC	ADCCSR ADCDR ADCAMP	A/D Control Status Register A/D Data Register A/D Amplifier Control Register	00h 00h 00h	R/W Read Only R/W
0037h	ITC	EICR	External Interrupt Control Register	00h	R/W
0038h 0039h	CLOCKS	MCCSR RCCR	Main Clock Control/Status Register RC oscillator Control Register	00h FFh	R/W R/W

DATA EEPROM (Cont'd)

Figure 9. Data E²PROM Write Operation

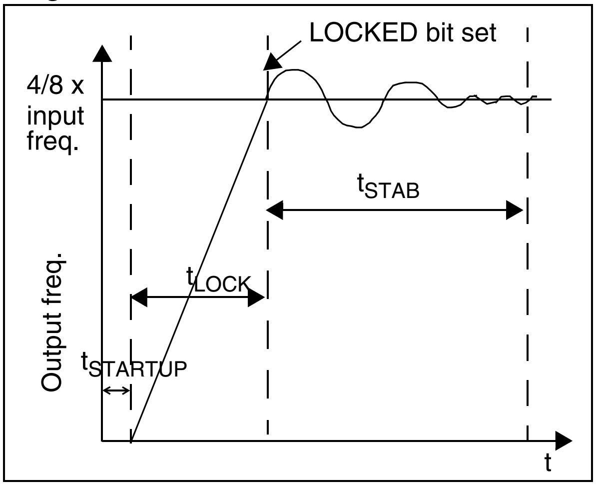
ROW DEFINITION

⇓ Row / Byte ⇒	0	1	2	3	...	30	31	Physical Address
0								00h...1Fh
1								20h...3Fh
...								
N								Nx20h...Nx20h+1Fh



Note: If a programming cycle is interrupted (by RESET action), the integrity of the data in memory will not be guaranteed.

Figure 13. PLL Output Frequency Timing Diagram



When the PLL is started, after reset or wakeup from Halt mode or AWUFH mode, it outputs the clock after a delay of $t_{STARTUP}$.

When the PLL output signal reaches the operating frequency, the LOCKED bit in the SICSCR register is set. Full PLL accuracy (ACC_{PLL}) is reached after a stabilization time of t_{STAB} (see Figure 13 and 13.3.4 Internal RC Oscillator and PLL)

Refer to section 8.4.4 on page 36 for a description of the LOCKED bit in the SICSR register.

7.3 REGISTER DESCRIPTION

MAIN CLOCK CONTROL/STATUS REGISTER (MCCSR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	MCO	SMS

Bits 7:2 = Reserved, must be kept cleared.

Bit 1 = **MCO** *Main Clock Out enable*
This bit is read/write by software and cleared by hardware after a reset. This bit allows to enable the MCO output clock.
0: MCO clock disabled, I/O port free for general purpose I/O.
1: MCO clock enabled.

Bit 0 = **SMS** *Slow Mode select*
This bit is read/write by software and cleared by hardware after a reset. This bit selects the input clock f_{OSC} or $f_{OSC}/32$.
0: Normal mode ($f_{CPU} = f_{OSC}$
1: Slow mode ($f_{CPU} = f_{OSC}/32$)

RC CONTROL REGISTER (RCCR)

Read / Write

Reset Value: 1111 1111 (FFh)

7							0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0

Bits 7:0 = **CR[7:0]** *RC Oscillator Frequency Adjustment Bits*
These bits must be written immediately after reset to adjust the RC oscillator frequency and to obtain an accuracy of 1%. The application can store the correct value for each voltage range in EEPROM and write it to this register at start-up.
00h = maximum available frequency
FFh = lowest available frequency

Note: To tune the oscillator, write a series of different values in the register until the correct frequency is reached. The fastest method is to use a dichotomy starting with 80h.

Table 5. Clock Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0038h	**MCCSR** Reset Value	0	0	0	0	0	0	MCO 0	SMS 0
0039h	**RCCR** Reset Value	CR7 1	CR6 1	CR5 1	CR4 1	CR3 1	CR2 1	CR1 1	CR0 1

7.4 RESET SEQUENCE MANAGER (RSM)

7.4.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 16:

- External $\overline{\text{RESET}}$ source pulse
- Internal LVD RESET (Low Voltage Detection)
- Internal WATCHDOG RESET

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 11.2.1 on page 53 for further details.

These sources act on the $\overline{\text{RESET}}$ pin and it is always kept low during the delay phase.

The RESET service routine vector is fixed at addresses FFFEh-FFFFh in the ST7 memory map.

The basic RESET sequence consists of 3 phases as shown in Figure 15:

- Active Phase depending on the RESET source
- 256 CPU clock cycle delay
- RESET vector fetch

The 256 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state.

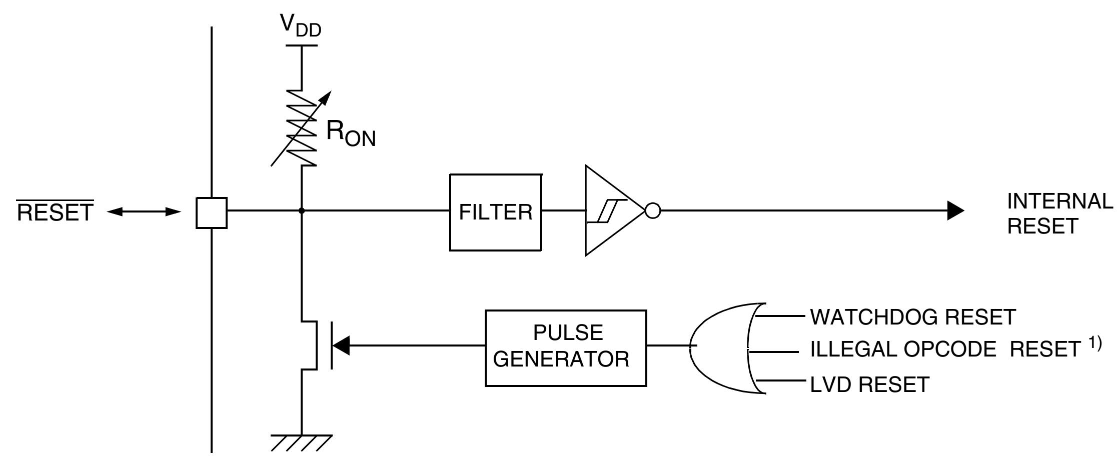
The RESET vector fetch phase duration is 2 clock cycles.

If the PLL is enabled by option byte, it outputs the clock after an additional delay of $t_{STARTUP}$ (see Figure 13).

Figure 15. RESET Sequence Phases

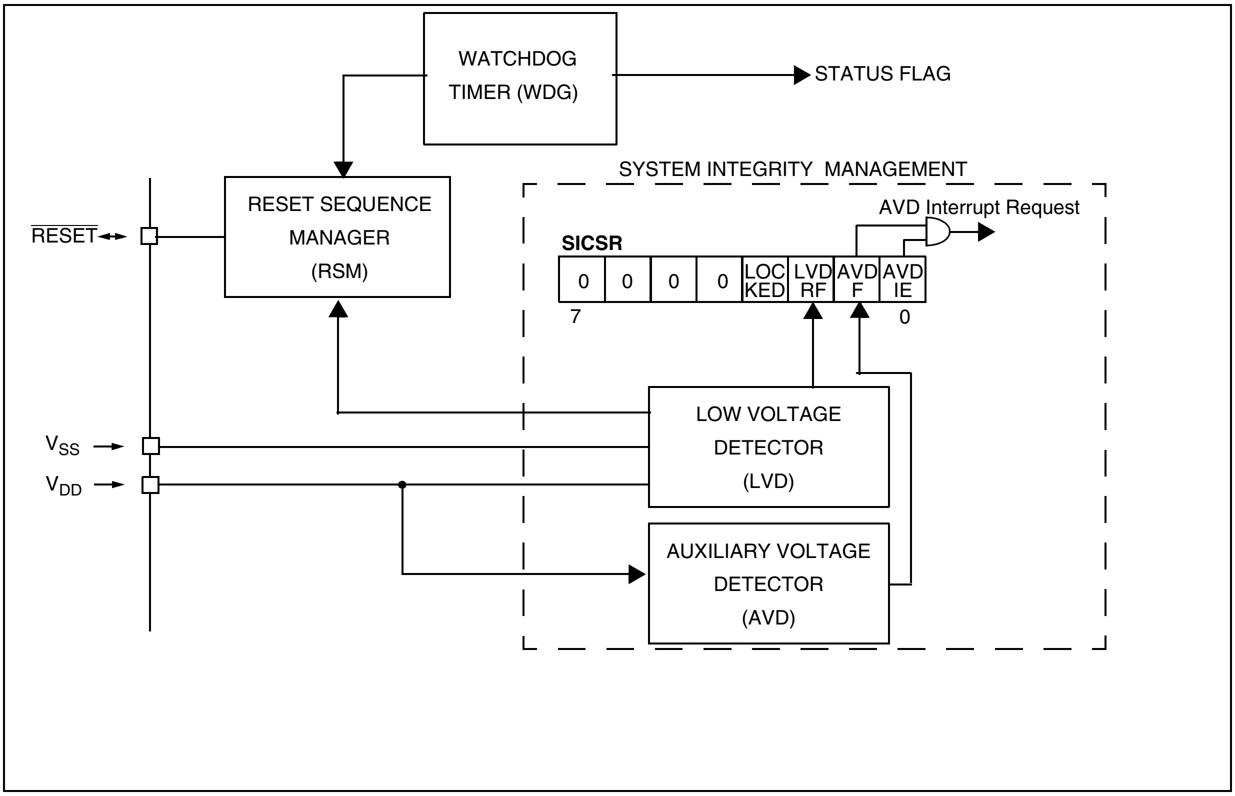
RESET		
Active Phase	INTERNAL RESET 256 CLOCK CYCLES	FETCH VECTOR

Figure 16.Reset Block Diagram



Note 1: See “Illegal Opcode Reset” on page 78. for more details on illegal opcode reset conditions.

Figure 20. Reset and Supply Management Block Diagram



I/O PORTS (Cont'd)

CAUTION: The alternate function must not be activated as long as the pin is configured as input with interrupt, in order to avoid generating spurious interrupts.

Analog alternate function

When the pin is used as an ADC input, the I/O must be configured as floating input. The analog multiplexer (controlled by the ADC registers) 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 in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

WARNING: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

10.3 UNUSED I/O PINS

Unused I/O pins must be connected to fixed voltage levels. Refer to Section 13.8.

10.4 LOW POWER MODES

Mode	Description
WAIT	No effect on I/O ports. External interrupts cause the device to exit from WAIT mode.
HALT	No effect on I/O ports. External interrupts cause the device to exit from HALT mode.

10.5 INTERRUPTS

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and the interrupt mask in the CC register is not active (RIM instruction).

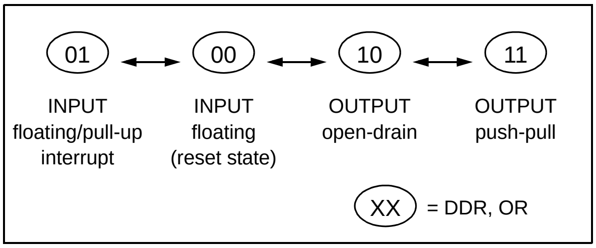
Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDRx ORx	Yes	Yes

10.6 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC Input.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 30 Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 30. Interrupt I/O Port State Transitions



The I/O port register configurations are summarised as follows.

Table 11. Port Configuration

Port	Pin name	Input (DDR=0)		Output (DDR=1)	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7	floating	pull-up interrupt	open drain	push-pull
	PA6:1	floating	pull-up	open drain	push-pull
	PA0	floating	pull-up interrupt	open drain	push-pull
Port B	PB4	floating	pull-up	open drain	push-pull
	PB3	floating	pull-up interrupt	open drain	push-pull
	PB2:1	floating	pull-up	open drain	push-pull
	PB0	floating	pull-up interrupt	open drain	push-pull

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 40)

In software management, the external $\overline{SS}$ pin is free for other application uses and the internal $\overline{SS}$ signal level is driven by writing to the SSI bit in the SPICSR register.

In Master mode:

– $\overline{SS}$ internal must be held high continuously

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 39):

If CPHA=1 (data latched on 2nd clock edge):

– $\overline{SS}$ internal must be held low during the entire transmission. This implies that in single slave applications the $\overline{SS}$ pin either can be tied to V_{SS}, or made free for standard I/O by managing the $\overline{SS}$ function by software (SSM= 1 and SSI=0 in the in the SPICSR register)

If CPHA=0 (data latched on 1st clock edge):

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 11.3.5.3).

Figure 39. Generic $\overline{SS}$ Timing Diagram

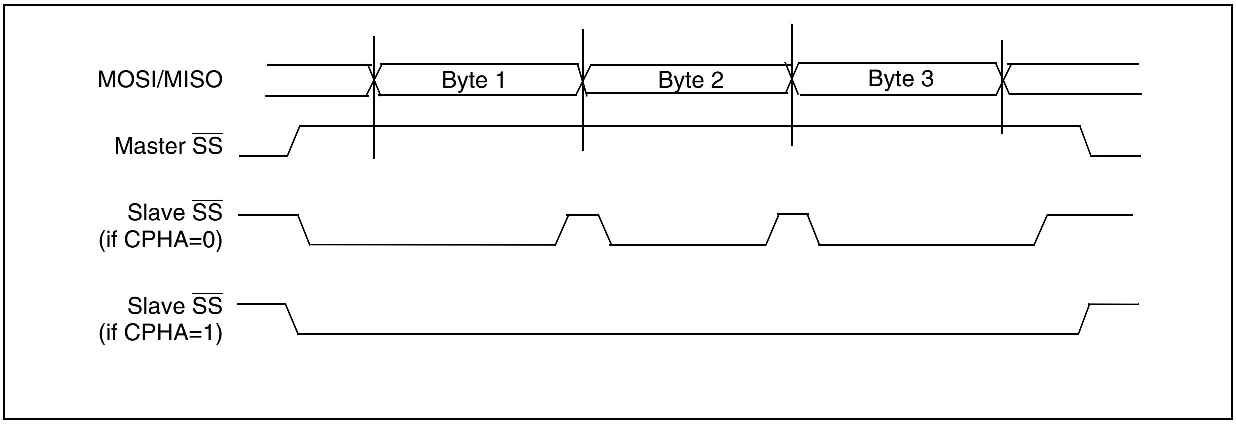


Figure 40. Hardware/Software Slave Select Management

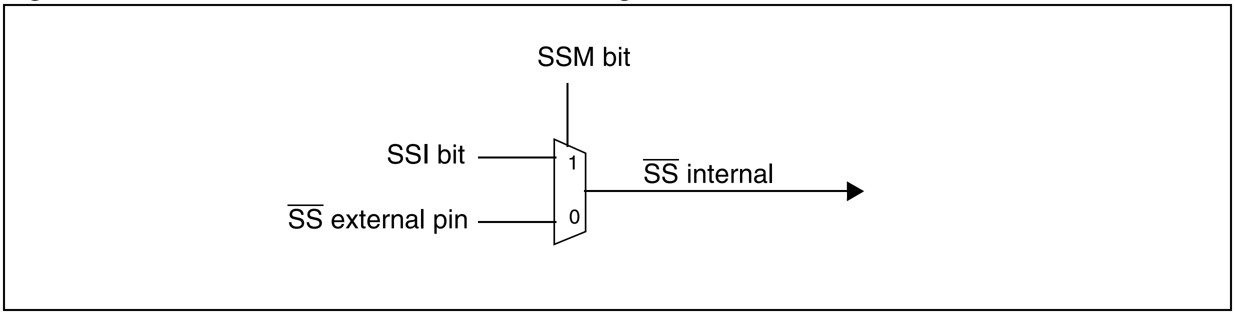
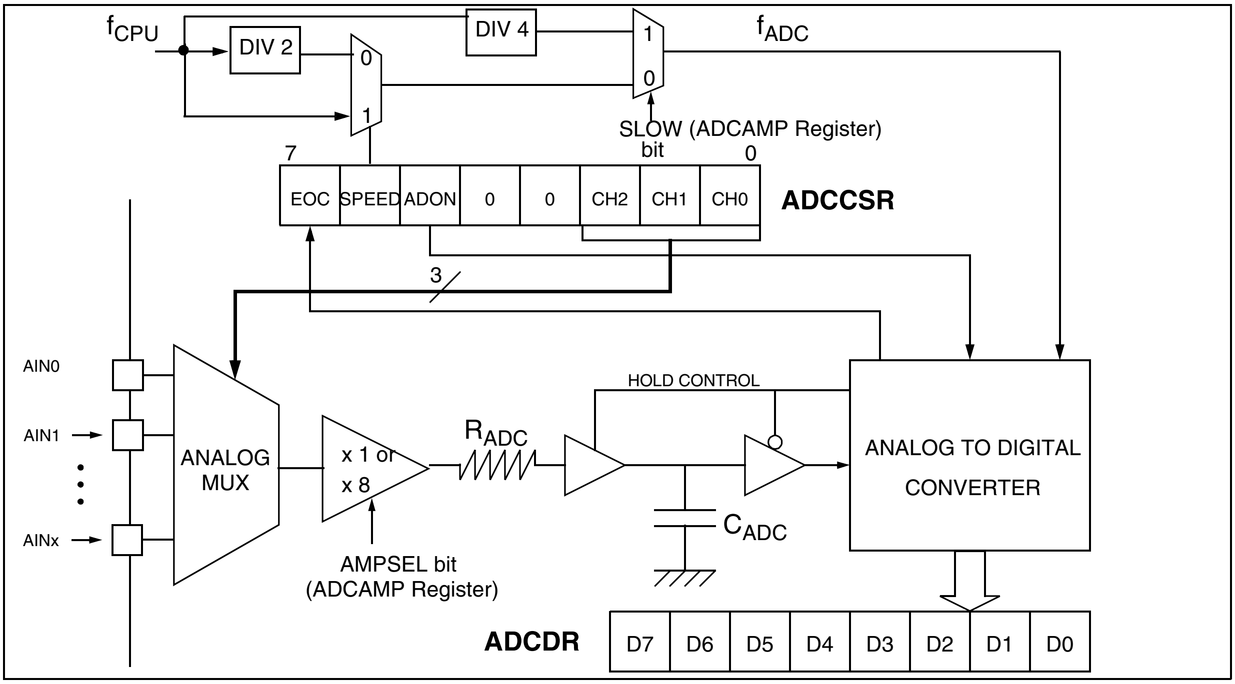


Figure 44. ADC Block Diagram



ST7 ADDRESSING MODES (cont'd)

12.1.1 Inherent

All Inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Subroutine Return
IRET	Interrupt Subroutine Return
SIM	Set Interrupt Mask
RIM	Reset Interrupt Mask
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

12.1.2 Immediate

Immediate instructions have 2 bytes, the first byte contains the opcode, the second byte contains the operand value.

Immediate Instruction	Function
LD	Load
CP	Compare
BCP	Bit Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Operations

12.1.3 Direct

In Direct instructions, the operands are referenced by their memory address.

The direct addressing mode consists of two submodes:

Direct (Short)

The address is a byte, thus requires only 1 byte after the opcode, but only allows 00 - FF addressing space.

Direct (Long)

The address is a word, thus allowing 64 Kbyte addressing space, but requires 2 bytes after the opcode.

12.1.4 Indexed (No Offset, Short, Long)

In this mode, the operand is referenced by its memory address, which is defined by the unsigned addition of an index register (X or Y) with an offset.

The indirect addressing mode consists of three submodes:

Indexed (No Offset)

There is no offset (no extra byte after the opcode), and allows 00 - FF addressing space.

Indexed (Short)

The offset is a byte, thus requires only 1 byte after the opcode and allows 00 - 1FE addressing space.

Indexed (Long)

The offset is a word, thus allowing 64 Kbyte addressing space and requires 2 bytes after the opcode.

12.1.5 Indirect (Short, Long)

The required data byte to do the operation is found by its memory address, located in memory (pointer).

The pointer address follows the opcode. The indirect addressing mode consists of two submodes:

Indirect (Short)

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - FF addressing space, and requires 1 byte after the opcode.

Indirect (Long)

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

INSTRUCTION GROUPS (cont'd)

Mnemo	Description	Function/Example	Dst	Src	H	I	N	Z	C
ADC	Add with Carry	A = A + M + C	A	M	H		N	Z	C
ADD	Addition	A = A + M	A	M	H		N	Z	C
AND	Logical And	A = A . M	A	M			N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M			N	Z	
BRES	Bit Reset	bres Byte, #3	M						
BSET	Bit Set	bset Byte, #3	M						
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M						C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M						C
CALL	Call subroutine								
CALLR	Call subroutine relative								
CLR	Clear		reg, M				0	1	
CP	Arithmetic Compare	tst(Reg - M)	reg	M			N	Z	C
CPL	One Complement	A = FFH-A	reg, M				N	Z	1
DEC	Decrement	dec Y	reg, M				N	Z	
HALT	Halt					0			
IRET	Interrupt routine return	Pop CC, A, X, PC			H	I	N	Z	C
INC	Increment	inc X	reg, M				N	Z	
JP	Absolute Jump	jp [TBL.w]							
JRA	Jump relative always								
JRT	Jump relative								
JRF	Never jump	jrf *							
JRIH	Jump if ext. interrupt = 1								
JRIL	Jump if ext. interrupt = 0								
JRH	Jump if H = 1	H = 1 ?							
JRNH	Jump if H = 0	H = 0 ?							
JRM	Jump if I = 1	I = 1 ?							
JRNM	Jump if I = 0	I = 0 ?							
JRMI	Jump if N = 1 (minus)	N = 1 ?							
JRPL	Jump if N = 0 (plus)	N = 0 ?							
JREQ	Jump if Z = 1 (equal)	Z = 1 ?							
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?							
JRC	Jump if C = 1	C = 1 ?							
JRNC	Jump if C = 0	C = 0 ?							
JRULT	Jump if C = 1	Unsigned <							
JRUGE	Jump if C = 0	Jmp if unsigned >=							
JRUGT	Jump if (C + Z = 0)	Unsigned >							

OPERATING CONDITIONS (Cont'd)

Figure 49. RC Osc Freq vs V_{DD} @ T_A=25°C (Calibrated with RCCR1: 3V @ 25°C)

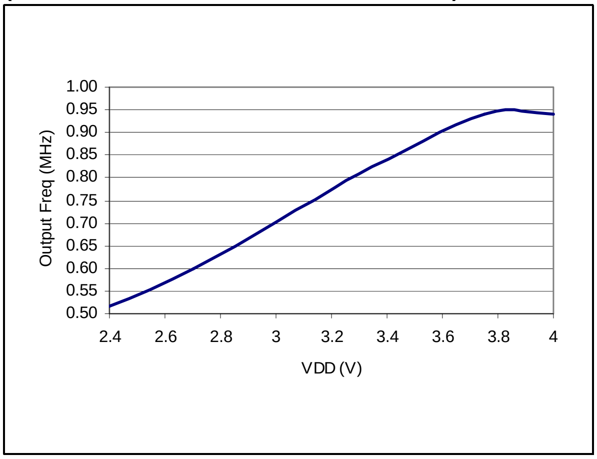


Figure 51. Typical RC oscillator Accuracy vs temperature @ V_{DD}=5V (Calibrated with RCCR0: 5V @ 25°C

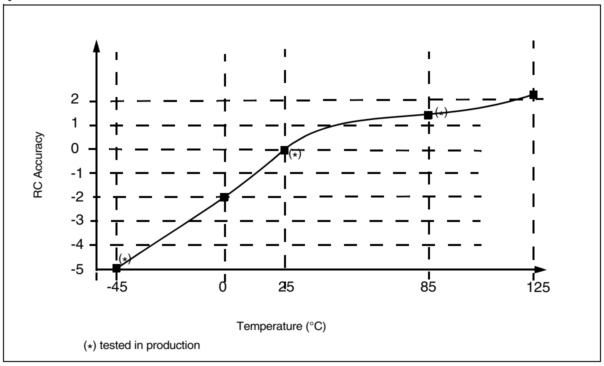


Figure 50. RC Osc Freq vs V_{DD} (Calibrated with RCCR0: 5V@ 25°C)

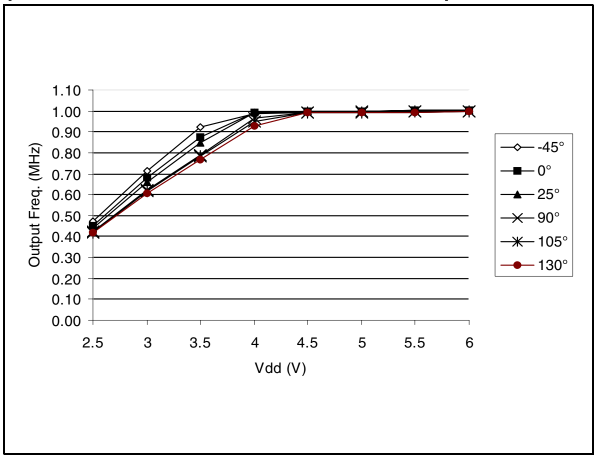


Figure 52. RC Osc Freq vs V_{DD} and RCCR Value

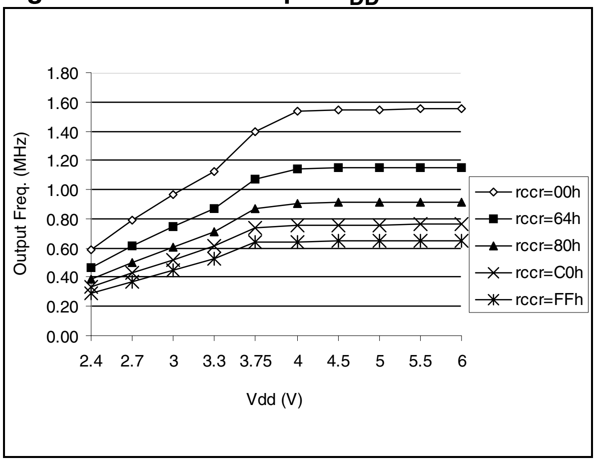
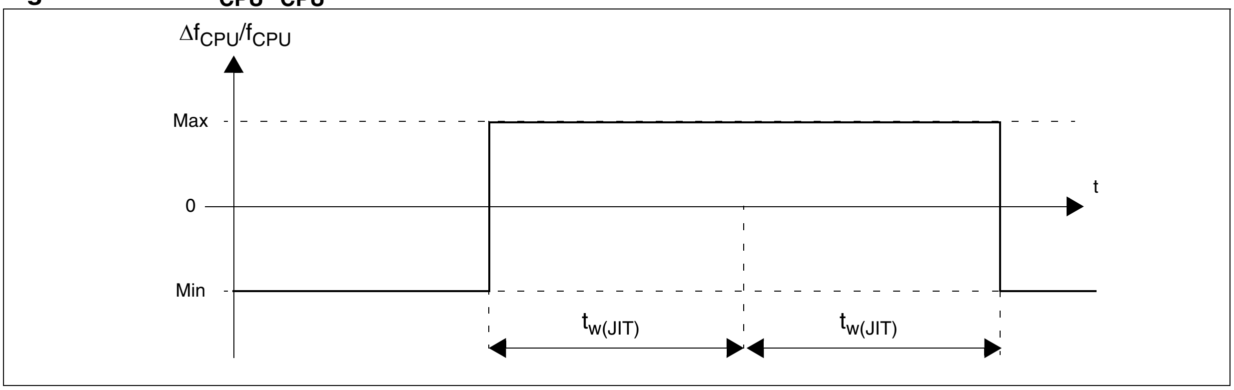


Figure 53. PLL $\Delta f_{CPU}/f_{CPU}$ versus time



13.4 SUPPLY CURRENT CHARACTERISTICS

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To get the total device consumption, the two current values must be added (except for HALT mode for which the clock is stopped).

13.4.1 Supply Current

T_A = -40 to +85°C unless otherwise specified

Symbol	Parameter	Conditions		Typ	Max	Unit
I_{DD}	Supply current in RUN mode	V_{DD}=5.5V	f_{CPU}=8MHz 1)	4.50	7.00	mA
	Supply current in WAIT mode		f_{CPU}=8MHz 2)	1.75	2.70	
	Supply current in SLOW mode		f_{CPU}=250kHz 3)	0.75	1.13	
	Supply current in SLOW WAIT mode		f_{CPU}=250kHz 4)	0.65	1	
	Supply current in HALT mode 5)		-40°C≤T_A≤+85°C	0.50	10	µA
			-40°C≤T_A≤+105°C	TBD	TBD	
			T_A= +85°C	5	100	

Notes:

1. CPU running with memory access, all I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.

2. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.

3. SLOW mode selected with f_{CPU} based on f_{OSC} divided by 32. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.

4. SLOW-WAIT mode selected with f_{CPU} based on f_{OSC} divided by 32. All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load), all peripherals in reset state; clock input (CLKIN) driven by external square wave, LVD disabled.

5. All I/O pins in output mode with a static value at V_{SS} (no load), LVD disabled. Data based on characterization results, tested in production at V_{DD} max and f_{CPU} max.

Figure 56. Typical I_{DD} in RUN vs. f_{CPU}

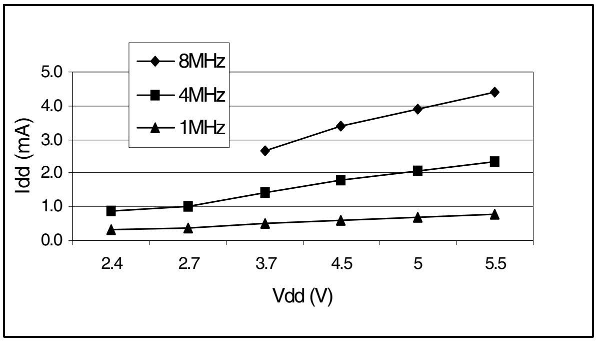


Figure 57. Typical I_{DD} in SLOW vs. f_{CPU}

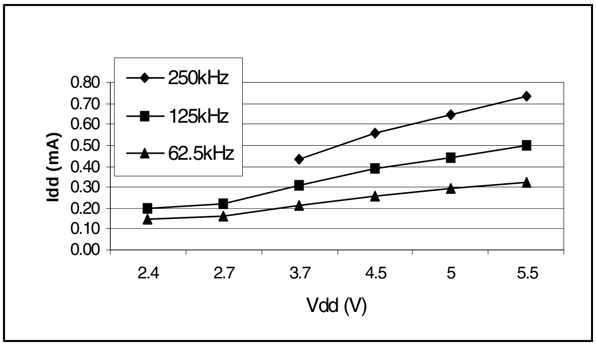


Figure 69. Typical V_{DD}-V_{OH} at V_{DD}=2.7V

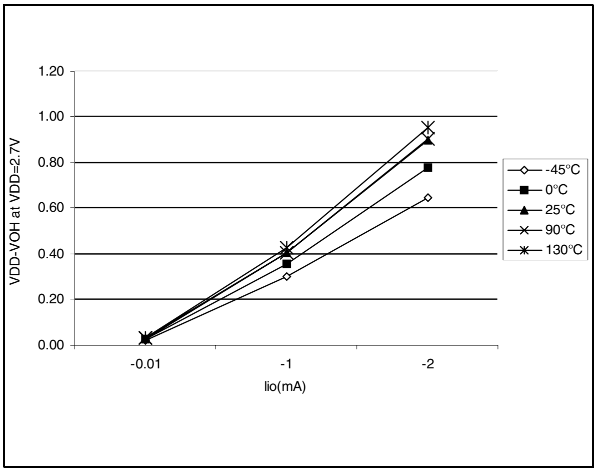


Figure 70. Typical V_{DD}-V_{OH} at V_{DD}=3V

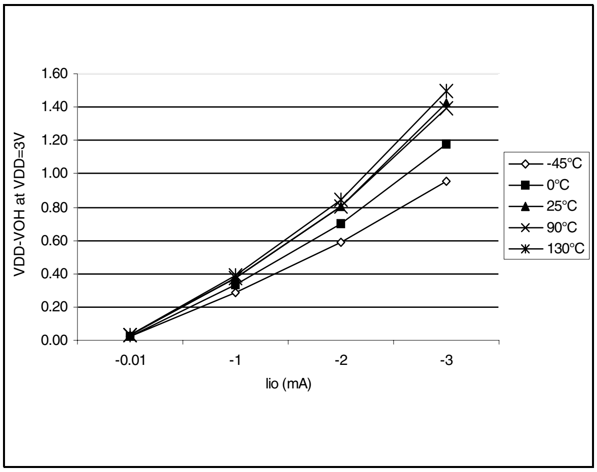


Figure 71. Typical V_{DD}-V_{OH} at V_{DD}=4V

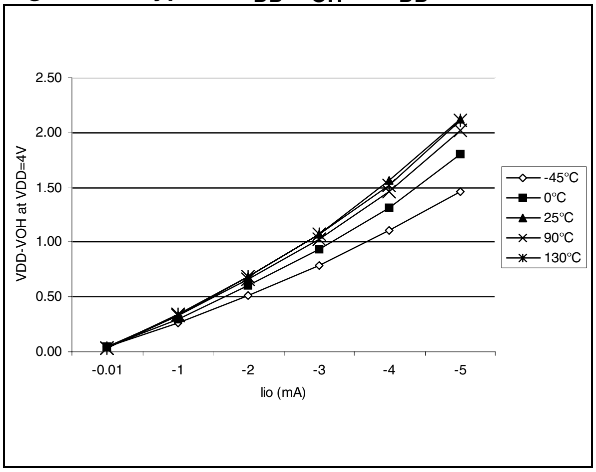


Figure 72. Typical V_{DD}-V_{OH} at V_{DD}=5V

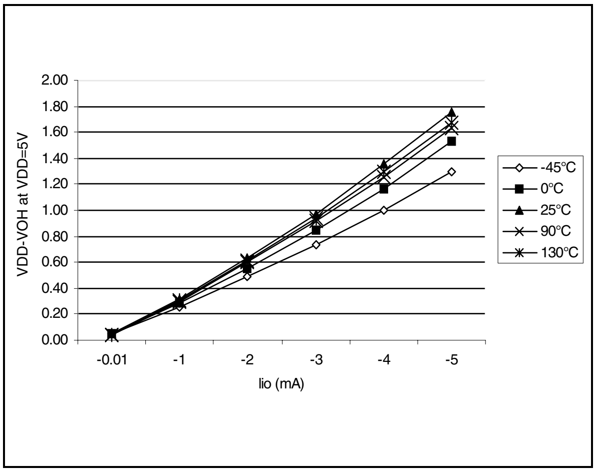
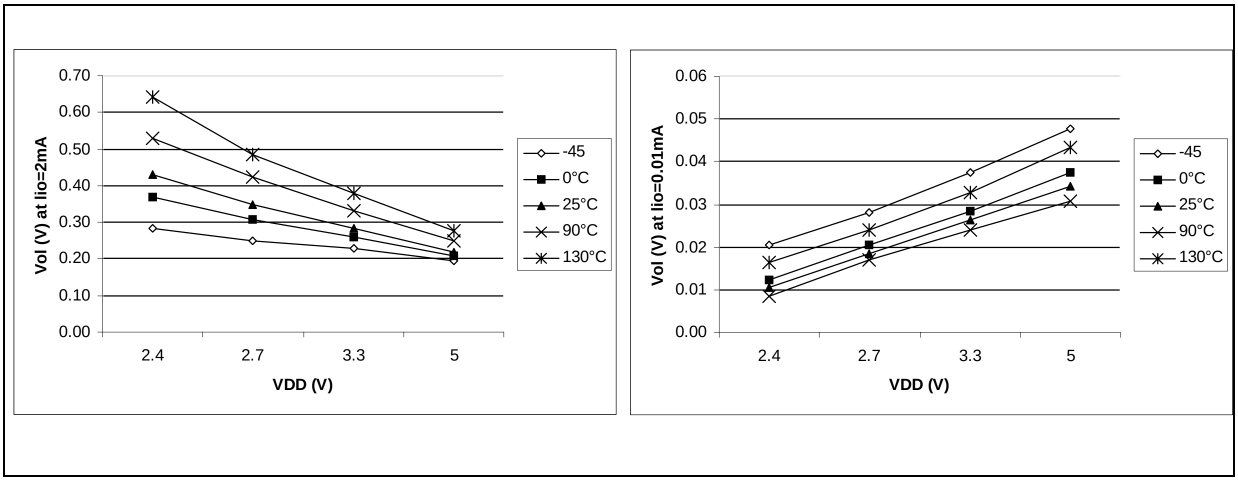


Figure 73. Typical V_{OL} vs. V_{DD} (standard I/Os)



CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 76. RESET pin protection when LVD is enabled. 1)2)3)4)

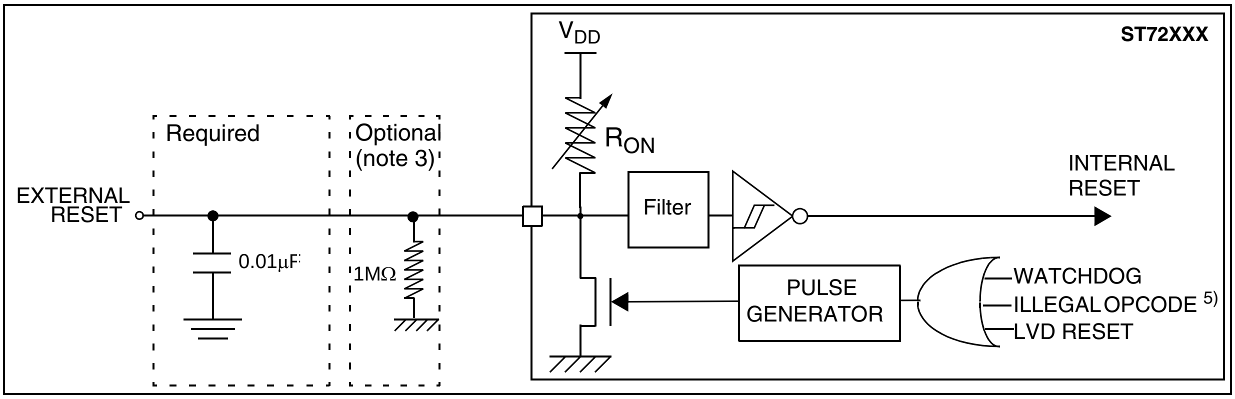
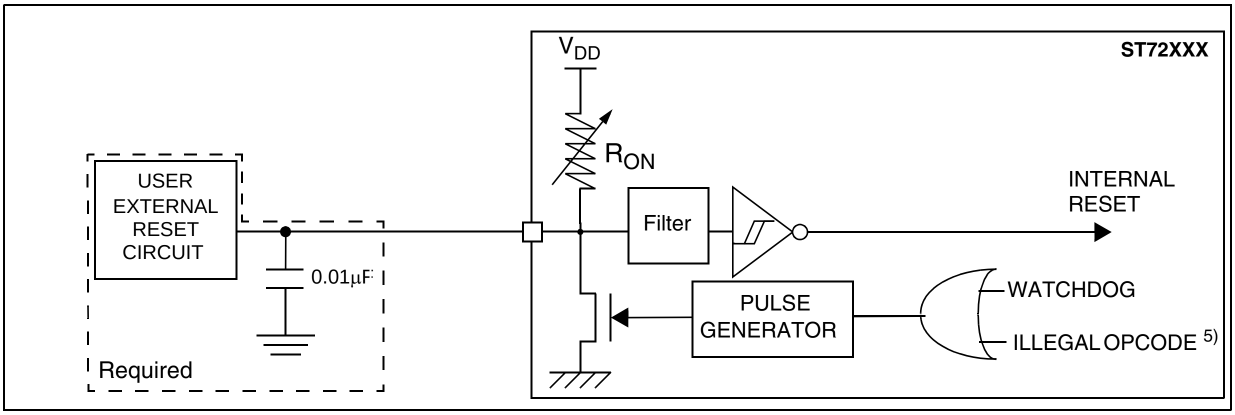


Figure 77. RESET pin protection when LVD is disabled. 1)



Note 1:

– The reset network protects the device against parasitic resets.

– The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).

– Whatever the reset source is (internal or external), the user must ensure that the level on the RESET pin can go below the V_{IL} max. level specified in section 13.9.1 on page 100. Otherwise the reset will not be taken into account internally.

– Because the reset circuit is designed to allow the internal RESET to be output in the RESET pin, the user must ensure that the current sunk on the RESET pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in section 13.2.2 on page 82.

Note 2: When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

Note 3: In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the RESET pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5µA to the power consumption of the MCU).

Note 4: Tips when using the LVD:

– 1. Check that all recommendations related to ICCCLK and reset circuit have been applied (see caution in Table 1 on page 7 and notes above)

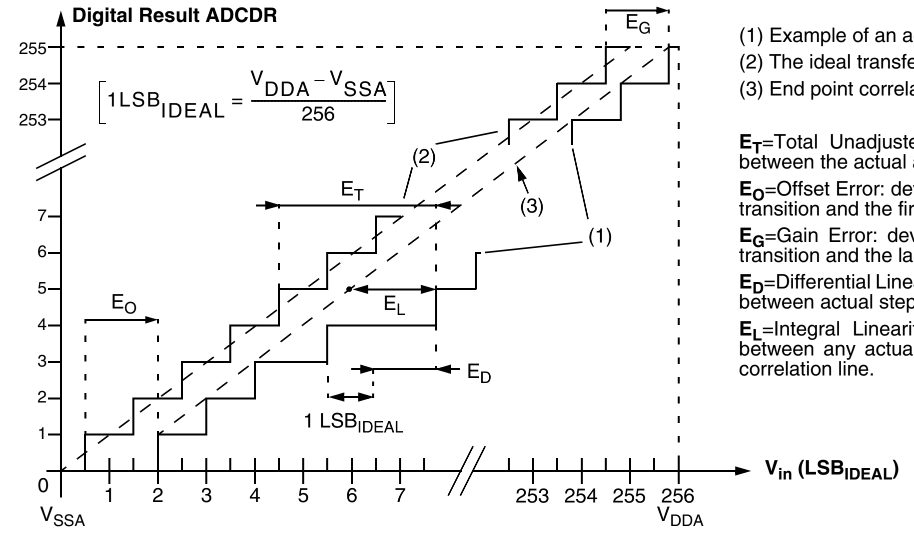
– 2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the RESET pin.

– 3. The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the RESET pin with a 5µF to 20µF capacitor."

Note 5: See "Illegal Opcode Reset" on page 78. for more details on illegal opcode reset conditions

ADC CHARACTERISTICS (Cont'd)

Figure 84. ADC Accuracy Characteristics with Amplifier disabled



(1) Example of an actual transfer curve
(2) The ideal transfer curve
(3) End point correlation line

E_T=Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.

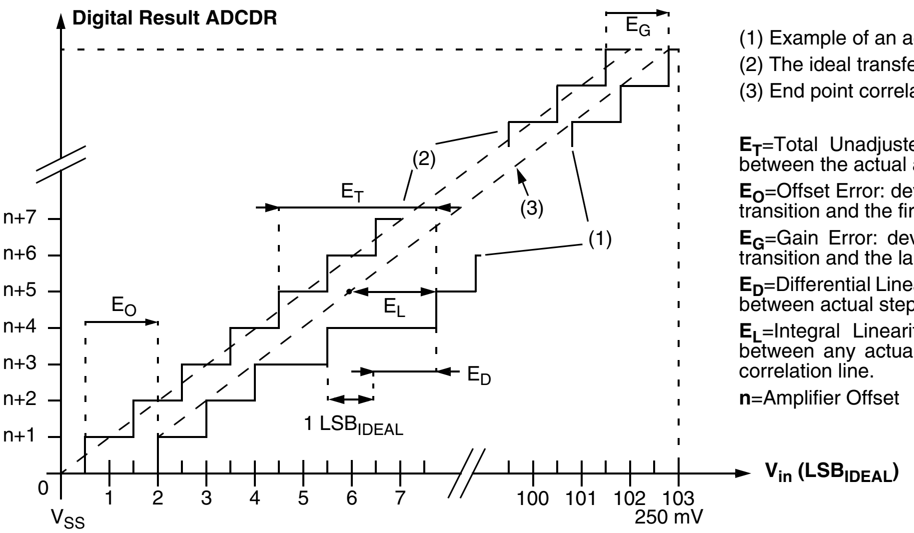
E_O=Offset Error: deviation between the first actual transition and the first ideal one.

E_G=Gain Error: deviation between the last ideal transition and the last actual one.

E_D=Differential Linearity Error: maximum deviation between actual steps and the ideal one.

E_L=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

Figure 85. ADC Accuracy Characteristics with Amplifier enabled



(1) Example of an actual transfer curve
(2) The ideal transfer curve
(3) End point correlation line

E_T=Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.

E_O=Offset Error: deviation between the first actual transition and the first ideal one.

E_G=Gain Error: deviation between the last ideal transition and the last actual one.

E_D=Differential Linearity Error: maximum deviation between actual steps and the ideal one.

E_L=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

n=Amplifier Offset

Note: When the AMPSEL bit in the ADCDRL register is set, it is mandatory that f_{ADC} be less than or equal to 2 MHz. (if f_{CPU}=8MHz. then SPEED=0, SLOW=1).

Figure 89. Ordering information scheme

Example: ST7 F LITES5 Y 0 M 6 TR

Family

ST7 Microcontroller Family

Memory type

F: Flash

P: FASTROM

Sub-family

LITES2, LITES5, LITE02, LITE05 or LITE09

No. of pins

Y = 16

Memory size

0 = 1K (LITESx versions) or 1.5K (LITE0x versions)

Package

B = DIP
M = SO
U = QFN

Temperature range

6 = -40 °C to 85 °C

Shipping Option

TR = Tape & Reel packing
Blank = Tube (DIP16 or SO16) or Tray (QFN20)

For a list of available options (e.g. data EEPROM, package) and orderable part numbers or for further information on any aspect of this device, please contact the ST Sales Office nearest to you.

15.3 DEVELOPMENT TOOLS

Development tools for the ST7 microcontrollers include a complete range of hardware systems and software tools from STMicroelectronics and third-party tool suppliers. The range of tools includes solutions to help you evaluate microcontroller peripherals, develop and debug your application, and program your microcontrollers.

15.3.1 Starter kits

ST offers complete, affordable **starter kits**. Starter kits are complete, affordable hardware/software tool packages that include features and samples to help you quickly start developing your application.

15.3.2 Development and debugging tools

Application development for ST7 is supported by fully optimizing **C Compilers** and the **ST7 Assembler-Linker** toolchain, which are all seamlessly integrated in the ST7 integrated development environments in order to facilitate the debugging and fine-tuning of your application. The Cosmic C Compiler is available in a free version that outputs up to 16KBytes of code.

The range of hardware tools includes full-featured **ST7-EMU3 series emulators**, cost effective **ST7-DVP3 series emulators** and the low-cost **RLink** in-circuit debugger/programmer. These tools are supported by the **ST7 Toolset** from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

15.3.3 Programming tools

During the development cycle, the **ST7-DVP3** and **ST7-EMU3 series emulators** and the **RLink** provide in-circuit programming capability for programming the Flash microcontroller on your application board.

ST also provides a low-cost dedicated in-circuit programmer, the **ST7-STICK**, as well as **ST7 Socket Boards** which provide all the sockets required for programming any of the devices in a specific ST7 sub-family on a platform that can be used with any tool with in-circuit programming capability for ST7.

For production programming of ST7 devices, ST's third-party tool partners also provide a complete range of gang and automated programming solutions, which are ready to integrate into your production environment.

15.3.4 Order Codes for Development and Programming Tools

Table 23 below lists the ordering codes for the ST7LITE0/ST7LITES development and programming tools. For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

15.3.5 Order codes for ST7LITE0/ST7LITES development tools

Table 23. Development tool order codes for the ST7LITE0/ST7LITES family

MCU	In-circuit Debugger, RLink Series[1]		Emulator		Programming Tool	
	Starter Kit without Demo Board	**Starter Kit with Demo Board**	**DVP Series**	**EMU Series**	**In-circuit Programmer**	**ST Socket Boards and EPBs**
ST7FLITE02, ST7FLITE05, ST7FLITE09, ST7FLITES2, ST7FLITES5	STX-RLINK[2]	ST7FLITE-SK/RAIS[2]	ST7MDT10-DVP3[3]	ST7MDT10-EMU3	STX-RLINK ST7-STICK[4][5]	ST7SB10-SU0[4]

Notes:

1. Available from ST or from Raisonance, www.raisonance.com

2. USB connection to PC

3. Includes connection kit for DIP16/SO16 only. See "How to order an EMU or DVP" in ST product and tool selection guide for connection kit ordering information

4. Add suffix /EU, /UK or /US for the power supply for your region

5. Parallel port connection to PC

16 KNOWN LIMITATIONS

16.1 Execution of BTJX Instruction

Description

Executing a BTJx instruction jumps to a random address in the following conditions: the jump goes to a lower address (jump backward) and the test is performed on a data located at the address 00FFh.

16.2 In-Circuit Programming of devices previously programmed with Hardware Watchdog option

Description

In-Circuit Programming of devices configured with Hardware Watchdog (WDGSW bit in option byte 1 programmed to 0) requires certain precautions (see below).

In-Circuit Programming uses ICC mode. In this mode, the Hardware Watchdog is not automatically deactivated as one might expect. As a consequence, internal resets are generated every 2 ms by the watchdog, thus preventing programming.

The device factory configuration is Software Watchdog so this issue is not seen with devices that are programmed for the first time. For the same reason, devices programmed by the user with the Software Watchdog option are not impacted.

The only devices impacted are those that have previously been programmed with the Hardware Watchdog option.

Workaround

Devices configured with Hardware Watchdog must be programmed using a specific programming mode that ignores the option byte settings. In this mode, an external clock, normally provided by the programming tool, has to be used. In ST tools, this mode is called "ICP OPTIONS DISABLED".

Sockets on ST programming tools (such as ST7MDT10-EPB) are controlled using "ICP OPTIONS DISABLED" mode. Devices can therefore be reprogrammed by plugging them in the ST Programming Board socket, whatever the watchdog configuration.

When using third-party tools, please refer the manufacturer's documentation to check how to access specific programming modes. If a tool does not have a mode that ignores the option byte settings, devices programmed with the Hardware watchdog option cannot be reprogrammed using this tool.

16.3 In-Circuit Debugging with Hardware Watchdog

In Circuit Debugging is impacted in the same way as In Circuit Programming by the activation of the hardware watchdog in ICC mode. Please refer to Section 16.2.

16.4 Recommendations when LVD is enabled

When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

16.5 Clearing Active Interrupts Outside Interrupt Routine

When an active interrupt request occurs at the same time as the related flag or interrupt mask is being cleared, the CC register may be corrupted.

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, i.e. when:

– The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request

Ex:

SIM

reset flag or interrupt mask

RIM