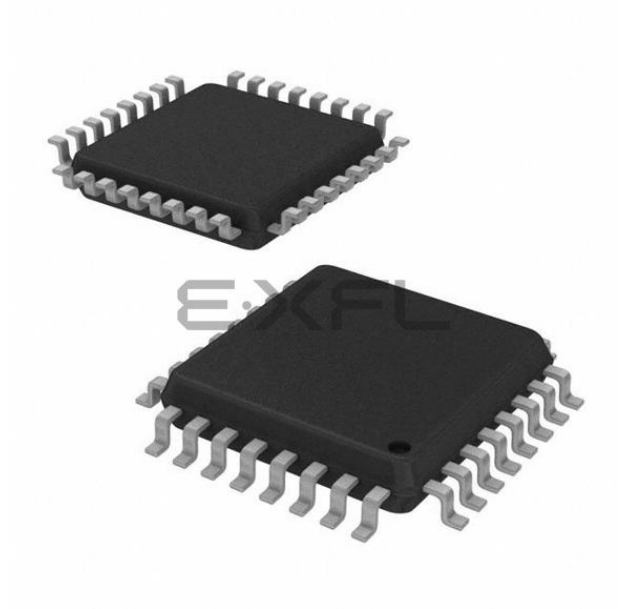


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	I²C, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7flite49k2t6

Contents

1 Description 15

2 Pin description 17

3 Register and memory mapping 21

4 Flash programmable memory 25

4.1 Introduction 25

4.2 Main features 25

4.3 Programming modes 25

4.3.1 In-circuit programming (ICP) 25

4.3.2 In-application programming (IAP) 26

4.4 ICC interface 26

4.5 Memory protection 28

4.5.1 Read-out protection 28

4.5.2 Flash write/erase protection 28

4.6 Related documentation 28

4.7 Description of Flash control/status register (FCSR) 28

5 Data EEPROM 29

5.1 Introduction 29

5.2 Main features 29

5.3 Memory access 30

5.3.1 Read operation (E2LAT=0) 30

5.3.2 Write operation (E2LAT=1) 30

5.4 Power saving modes 31

5.4.1 Wait mode 31

5.4.2 Active-halt mode 31

5.4.3 Halt mode 31

5.5 Access error handling 31

5.6 Data EEPROM read-out protection 32

5.7 EEPROM control/status register (EECSR) 32

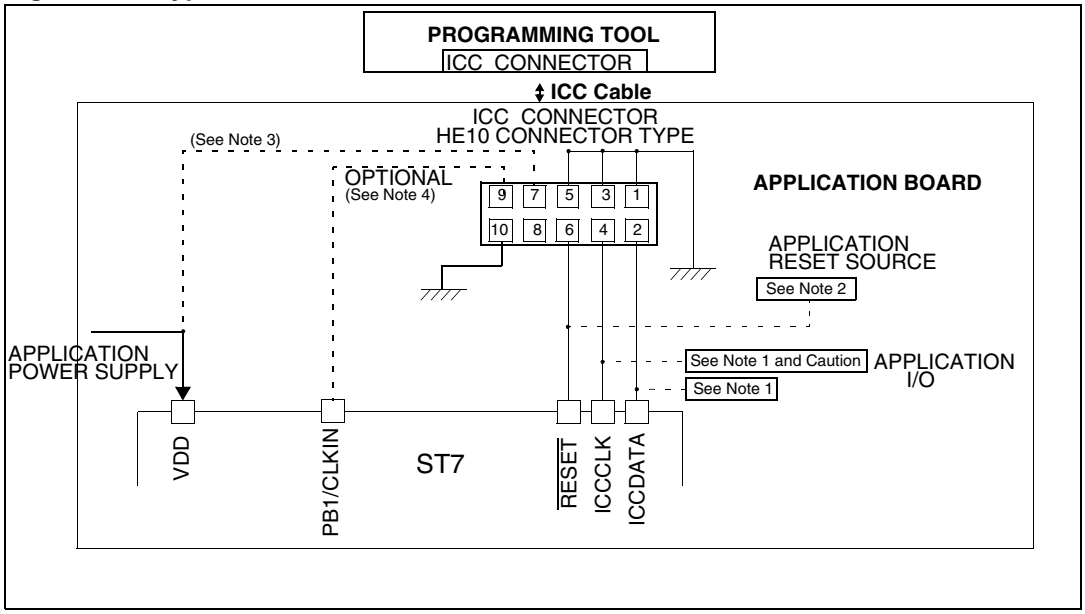
Table 2. ST7LITE49K2 device pin description

Pin number		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input				Output			
LQFP32	SDIP32			Input	Output	float	wpu	int	ana	OD[1]	PP		
26	30	PC4/LTIC/ COMPINB-	I/O	C_T		**x**	ei2			x	x	Port C4	LTIC/Analog Comparator External Reference Input B
27	31	PC5/COMPINB+/ BREAK2	I/O	C_T		**x**				x	x	Port C5	Analog Comparator Input B/ External break 2
28	32	PC6/COMPINA+	I/O	C_T		**x**	ei2			x	x	Port C6	Analog Comparator Input A
29	1	PC7/BREAK1/ COMPOUTA	I/O	C_T		**x**				x	x	Port C7	BREAK1/Analog Comparator Output A
30	2	PA0 /COMPINA- /OCMP1_A	I/O	C_T		**x**	ei0			x	x	Port A0	Analog comparator external reference Input A/ Timer A Output Compare 1
31	3	PA1(HS)/ATIC	I/O	C_T	HS	**x**				x	x	Port A1 (HS)	ATIC
32	4	PA2(HS)/ATPWM0	I/O	C_T	HS	**x**				x	x	Port A2 (HS)	ATPWM0

1. In the open-drain output column, T defines a true open-drain I/O (P-Buffer and protection diode to V_{DD} are not implemented).0
2. It is mandatory to connect all available V_{DD} and V_{DDA} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground.

during a reset. In the application, even if the pin is configured as output, any reset will put it back in input pull-up.

Figure 5. Typical ICC Interface



4.5 Memory protection

There are two different types of memory protection: Read-Out Protection and Write/Erase Protection which can be applied individually.

4.5.1 Read-out protection

Read-Out Protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller. Both program and data EEPROM memory are protected.

In Flash devices, this protection is removed by reprogramming the option. In this case, both program and data EEPROM memory are automatically erased and the device can be reprogrammed.

Read-Out Protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP_R bit in the option byte.
- In ROM devices it is enabled by mask option specified in the option list.

4.5.2 Flash write/erase protection

Write/Erase Protection, when set, makes it impossible to both overwrite and erase program memory. It does not apply to EEPROM data. Its purpose is to provide advanced security to applications and prevent any change being made to the memory content. Write/Erase Protection is enabled through the FMP_W bit in the option byte.

Caution: **Once set, Write/Erase Protection can never be removed. A write-protected Flash device is no longer reprogrammable.**

4.6 Related documentation

For details on Flash programming and ICC protocol, refer to the ST7 Flash Programming Reference Manual and to the ST7 ICC Protocol Reference Manual.

4.7 Description of Flash control/status register (FCSR)

This register controls the XFlash erasing and programming using ICP, IAP or other programming methods.

1st RASS Key: 0101 0110 (56h)

2nd RASS Key: 1010 1110 (AEh)

When an EPB or another programming tool is used (in socket or ICP mode), the RASS keys are sent automatically.

Reset value: 000 0000 (00h)

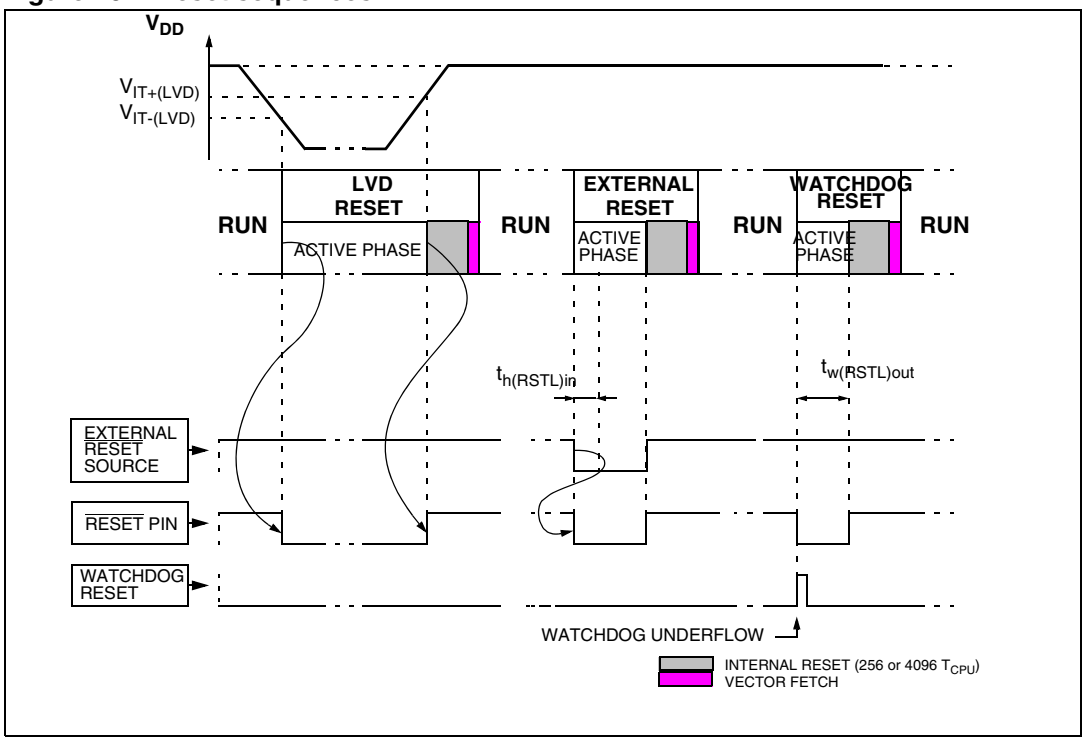
7							0
0	0	0	0	0	OPT	LAT	PGM
Read/write							

7.3.5 Internal watchdog reset

The Reset sequence generated by an internal watchdog counter overflow is shown in *Figure 16: Reset sequences*

Starting from the watchdog counter underflow, the device $\overline{\text{RESET}}$ pin acts as an output that is pulled low during at least $t_{w(RSTL)out}$.

Figure 16. Reset sequences



8 Interrupts

8.1 Introduction

The ST7 enhanced interrupt management provides the following features:

- Hardware interrupts
- Software interrupt (TRAP)
- Nested or concurrent interrupt management with flexible interrupt priority and level management:
  - Up to 4 software programmable nesting levels
  - 13 interrupt vectors fixed by hardware
  - 2 non maskable events: RESET, TRAP

This interrupt management is based on:

- Bit 5 and bit 3 of the CPU CC register (I1:0),
- Interrupt software priority registers (ISPRx),
- Fixed interrupt vector addresses located at the high addresses of the memory mapping (FFE0h to FFFFh) sorted by hardware priority order.

This enhanced interrupt controller guarantees full upward compatibility with the standard (not nested) ST7 interrupt controller.

8.2 Masking and processing flow

The interrupt masking is managed by the I1 and I0 bits of the CC register and the ISPRx registers which give the interrupt software priority level of each interrupt vector (see *Table 14*). The processing flow is shown in *Figure 20*.

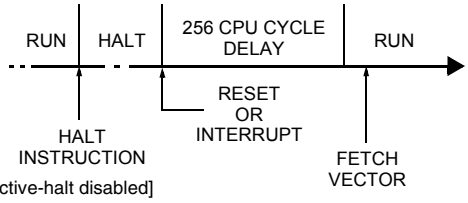
When an interrupt request has to be serviced:

- Normal processing is suspended at the end of the current instruction execution.
- The PC, X, A and CC registers are saved onto the stack.
- I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.
- The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to interrupt mapping table for vector addresses).

The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

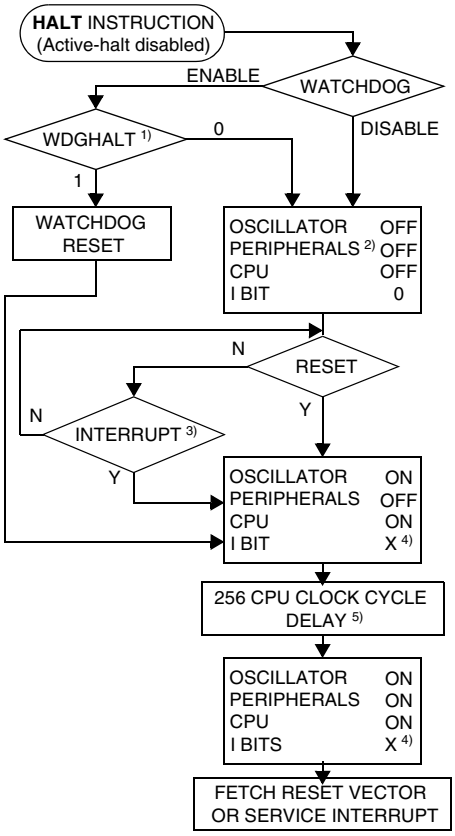
Note: As a consequence of the IRET instruction, the I1 and I0 bits will be restored from the stack and the program in the previous level will resume.

Figure 29. Halt timing overview



1. A reset pulse of at least 42 µs must be applied when exiting from Halt mode.

Figure 30. Halt mode flowchart



1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to *Table 18: ST7LITE49K2 interrupt mapping*for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.
5. The CPU clock must be switched to 1 MHz (RC/8) or AWU RC before entering Halt mode.

9.5.3 AWUFH prescaler register (AWUPR)

Reset value: 1111 1111 (FFh)

7							0
AWUPR7	AWUPR6	AWUPR5	AWUPR4	AWUPR3	AWUPR2	AWUPR1	AWUPR0
Read/Write							

Bits 7:0= **AWUPR[7:0]** *Auto-wakeup prescaler*

These 8 bits define the AWUPR Dividing factor (see *Table 21*).

Table 21. Configuring the dividing factor

AWUPR[7:0]	Dividing factor
00h	Forbidden
01h	1
...	...
FEh	254
FFh	255

In AWU mode, the time during which the MCU stays in Halt mode, t_{AWU}, is given by the equation below. See also *Figure 32 on page 70*.

$$t_{AWU} = 64 \times AWUPR \times \frac{1}{f_{AWURC}} + t_{RCSTRT}$$

The AWUPR prescaler register can be programmed to modify the time during which the MCU stays in Halt mode before waking up automatically.

Note: If 00h is written to AWUPR, the AWUPR remains unchanged.

Table 22. AWU register mapping and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0048h	**AWUCSR** Reset Value	0	0	0	0	0	AWUF	AWUM	AWUEN
0049h	**AWUPR** Reset Value	AWUPR7 1	AWUPR6 1	AWUPR5 1	AWUPR4 1	AWUPR3 1	AWUPR2 1	AWUPR1 1	AWUPR0 1

One pulse mode

One pulse mode can be used to control PWM2/3 signal with an external LTIC pin. This mode is available only in Dual Timer mode i.e. only for CNTR2, when the OP_EN bit in PWM3CSR register is set.

One Pulse mode is activated by the external LTIC input. The active edge of the LTIC pin is selected by the OPEDGE bit in the PWM3CSR register.

After getting the active edge of the LTIC pin, CNTR2 is reset (000h) and PWM3 is set to high. CNTR2 starts counting from 000h, when it reaches the active DCR3 value then PWM3 goes low. Till this time, any further transitions on the LTIC signal will have no effect. If there are LTIC transitions after CNTR2 reaches DCR3 value, CNTR2 is reset again and PWM3 goes high.

If there is no LTIC active edge, CNTR2 counts until it reaches the ATR2 value, then it is reset again and PWM3 is set to high. The counter again starts counting from 000h, when it reaches the active DCR3 value PWM3 goes low, the counter counts until it reaches ATR2, it resets and PWM3 is set to high and so on.

The same operation applies for PWM2, but in this case the comparison is done on DCR2. OP_EN and OPEDGE bits take effect on the fly and are not synchronized with Counter 2 overflow. The output bit OP2/3 can be used to inverse the polarity of PWM2/3 in one-pulse mode. The update of these bits (OP2/3) is synchronized with the counter 2 overflow, they will be updated if the TRAN2 bit is set.

The time taken from activation of LTIC input and CNTR2 reset is between 2 and 3 t_{CNTR2} cycles, that is, from around 62.5 ns to 94 ns (at 32 MHz input frequency).

Lite timer Input Capture interrupt should be disabled while 12-bit ARtimer is in One Pulse mode. This is to avoid spurious interrupts.

The priority of the various conditions for PWM3 is the following: Break > one-pulse mode with active LTIC edge > Forced overflow by s/w > one-pulse mode without active LTIC edge > normal PWM operation.

It is possible to update DCR2/3 and OP2/3 at the counter 2 reset, the update is synchronized with the counter reset. This is managed by the overflow interrupt which is generated if counter is reset either due to ATR match or active pulse at LTIC pin. DCR2/3 and OP2/3 update in one-pulse mode is performed dynamically using a software force update. DCR3 update in this mode is not synchronized with any event. That may lead to a longer next PWM3 cycle duration than expected just after the change.

In One Pulse mode ATR2 value must be greater than DCR2/3 value for PWM2/3. (opposite to normal PWM mode).

If there is an active edge on the LTIC pin after the counter has reset due to an ATR2 match, then the timer again gets reset and appears as modified Duty cycle depending on whether the new DCR value is less than or more than the previous value.

The TRAN2 bit should be set along with the FORCE2 bit with the same instruction after a write to the DCR register.

ATR2 value should be changed after an overflow in one pulse mode to avoid any irregular PWM cycle.

When exiting from one pulse mode, the OP_EN bit in the PWM3CSR register should be reset first and then the ENCNTR2 bit (if counter 2 must be stopped).

Dead time generator register (DTGR)

Reset value: 0000 0000 (00h)

7							0
DTE	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Read/write							

Bit 7 = **DTE** *Dead time enable bit*

This bit is read/write by software. It enables a dead time generation on PWM0/PWM1.

0: No Dead time insertion.

1: Dead time insertion enabled.

Bits 6:0 = **DT[6:0]** *Dead time value*

These bits are read/write by software. They define the dead time inserted between PWM0/PWM1. Dead time is calculated as follows:

Dead Time = DT[6:0] x Tcounter1

Note: If DTE is set and DT[6:0]=0, PWM output signals will be at their reset state.

Table 38. Register mapping and reset values

Add. (Hex)	Register label	7	6	5	4	3	2	1	0
0011	**ATCSR** Reset Value	0	ICF 0	ICIE 0	CK1 0	CK0 0	OVF1 0	OVFIE1 0	CMPIE 0
0012	**CNTR1H** Reset Value	0	0	0	0	CNTR1_11 0	CNTR1_10 0	CNTR1_9 0	CNTR1_8 0
0013	**CNTR1L** Reset Value	CNTR1_7 0	CNTR1_8 0	CNTR1_7 0	CNTR1_6 0	CNTR1_3 0	CNTR1_2 0	CNTR1_1 0	CNTR1_0 0
0014	**ATR1H** Reset Value	0	0	0	0	ATR11 0	ATR10 0	ATR9 0	ATR8 0
0015	**ATR1L** Reset Value	ATR7 0	ATR6 0	ATR5 0	ATR4 0	ATR3 0	ATR2 0	ATR1 0	ATR0 0
0016	**PWMCR** Reset Value	0	OE3 0	0	OE2 0	0	OE1 0	0	OE0 0
0017	**PWM0CSR** Reset Value	0	0	0	0	0	0	OP0 0	CMPF0 0
0018	**PWM1CSR** Reset Value	0	0	0	0	0	0	OP1 0	CMPF1 0
0019	**PWM2CSR** Reset Value	0	0	0	0	0	0	OP2 0	CMPF2 0
001A	**PWM3CSR** Reset Value	0	0	0	0	OP_EN 0	OPEDGE 0	OP3 0	CMPF3 0
001B	**DCR0H** Reset Value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0

Input capture

In this section, the index, *i*, may be 1 or 2 because there are two input capture functions in the 16-bit timer.

The two input capture 16-bit registers (IC1R and IC2R) are used to latch the value of the free-running counter after a transition detected by the ICAP*i* pin (see below).

	MSB	LSB
ICiR	IC*i*HR	IC*i*LR

IC*i*R register is a read-only register.

The active transition is software programmable through the IEDG*i* bit of control registers (CR*i*).

Timing resolution is one count of the free running counter: (f_{CPU}/CC[1:0]).

Procedure

To use the input capture function, select the following in the CR2 register:

- Select the timer clock (CC[1:0]).
- Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as floating input).

Select the following in the CR1 register:

- Set the ICIE bit to generate an interrupt after an input capture coming from either the ICAP1 pin or the ICAP2 pin
- Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1pin must be configured as floating input).

When an input capture occurs:

- The ICF*i* bit is set
- The IC*i*R register contains the value of the free running counter on the active transition on the ICAP*i* pin (see *Figure 61*).
- A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

Clearing the input capture interrupt request (that is, clearing the ICF*i* bit) is done in two steps:

1. By reading the SR register while the ICF*i* bit is set.
2. By accessing (reading or writing) the IC*i*LR register.

Note: 1 After reading the ICiHR register, transfer of input capture data is inhibited and ICFi is never set until the ICiLR register is also read.

2 The ICiR register contains the free running counter value which corresponds to the most recent input capture.

3 The two input capture functions can be used together even if the timer also uses the two output compare functions.

4 In One Pulse mode and PWM mode only the input capture 2 can be used.

5 The alternate inputs (ICAP1 and ICAP2) are always directly connected to the timer. So any transitions on these pins activate the input capture function. Moreover if one of the ICAPi pin is configured as an input and the second one as an output, an interrupt can be generated if

11.4.5 Interrupts

Table 43. 16-bit timer interrupt control/wakeup capability(1)

Interrupt event	Event flag	Enable control bit	Exit from Wait	Exit from Halt
Input capture 1 event/counter reset in PWM mode	ICF1	ICIE	Yes	No
Input capture 2 event	ICF2		Yes	No
Output compare 1 event (not available in PWM mode)	OCF1	OCIE	Yes	No
Output compare 2 event (not available in PWM mode)	OCF2		Yes	No
Timer overflow event	TOF	TOIE	Yes	No

1. The 16-bit timer interrupt events are connected to the same interrupt vector (see *Section 8: Interrupts*). These events generate an interrupt if the corresponding enable control bit is set and the interrupt mask in the CC register is reset (RIM instruction).

11.4.6 Summary of 16-bit timer modes

Table 44. Summary of 16-bit timer modes

Modes	Available resources			
	Input capture 1	Input capture 2	Output compare 1	Output compare 2
Input capture (1) and/or (2)	Yes	Yes	Yes	Yes
Output compare (1) and/or (2)	Yes	Yes	Yes	Yes
One pulse mode	No	Not recommended(1)	No	Partially(2)
PWM mode	No	Not recommended(3)	No	No

1. See note 4 in *One pulse mode on page 125*.
2. See note 5 in *One pulse mode on page 125*.
3. See note 4 in *Pulse width modulation mode on page 127*.

Input capture 2 high register (IC2HR)

Reset value: undefined

This is an 8-bit read-only register that contains the high part of the counter value (transferred by the input capture 2 event).

7							0
MSB							LSB
Read Only							

Input capture 2 low register (IC2LR)

Reset value: undefined

This is an 8-bit read-only register that contains the low part of the counter value (transferred by the input capture 2 event).

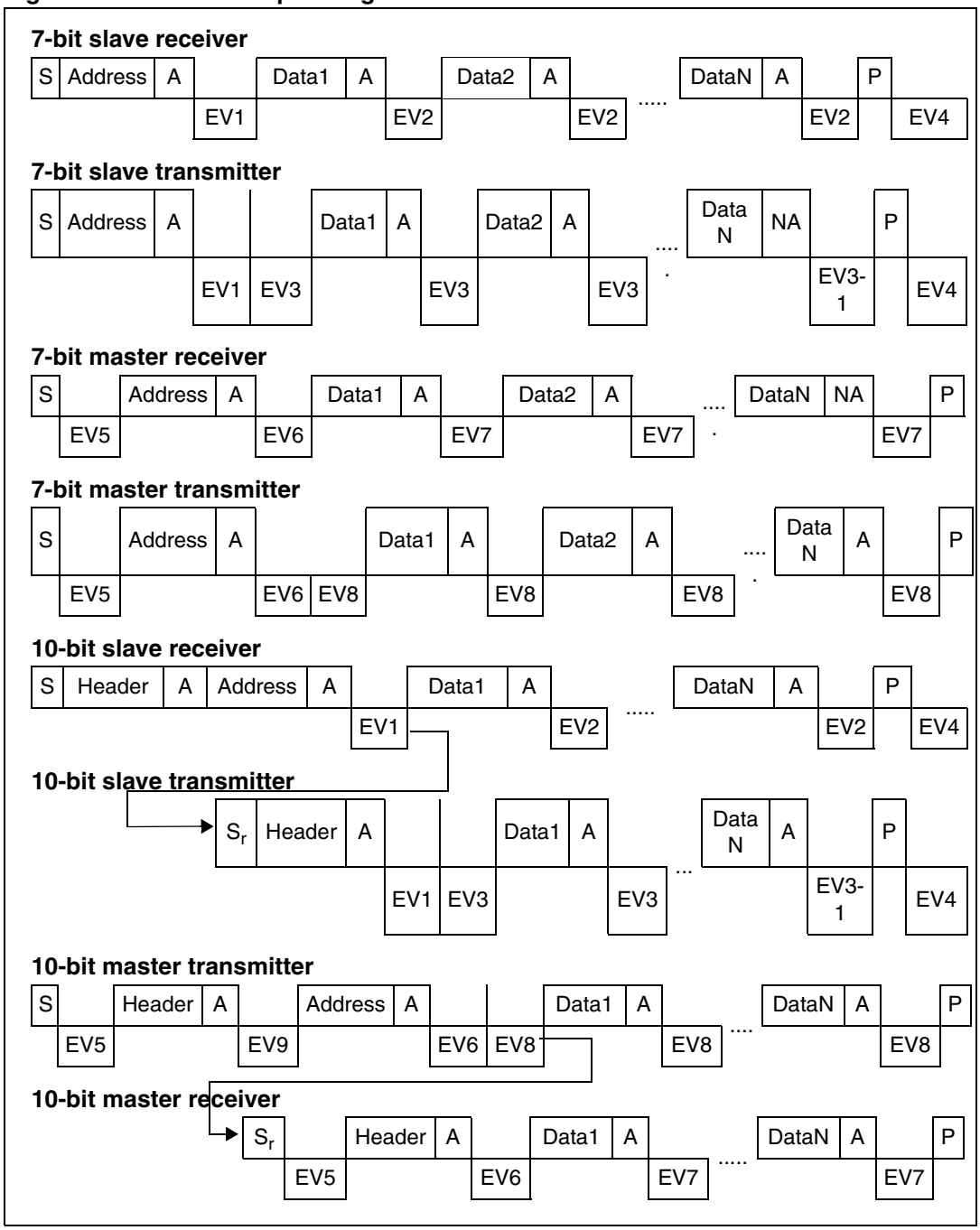
7							0
MSB							LSB
Read Only							

11.4.8 16-bit timer register map and reset values

Table 45. 16-bit timer register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
55	TACR2 Reset value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
56	TACR1 Reset value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
57	TACSR Reset value	ICF1 0	OCF1 0	TOF 0	ICF2 0	OCF2 0	TIMD 0	- 0	- 0
58	TAICHR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
59	TAICLR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
5A	TAOCHR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
5B	TAOCLR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
5C	TACHR Reset value	MSB 1	1	1	1	1	1	1	LSB 1
5D	TACLR Reset value	MSB 1	1	1	1	1	1	0	LSB 0

Figure 71. Transfer sequencing



1. S=Start, S_r = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge, EVx=Event (with interrupt if ITE=1).
2. **EV1:** EVF=1, ADSL=1, cleared by reading SR1 register.
3. **EV2:** EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.
4. **EV3:** EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.
5. **EV3-1:** EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register. BTF is cleared by releasing the lines (STOP=1, STOP=0) or by writing DR register (DR=FFh). If lines are released by STOP=1, STOP=0, the

subsequent EV4 is not seen.

6. **EV4:** EVF=1, STOPF=1, cleared by reading SR2 register.
7. **EV5:** EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.
8. **EV6:** EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).
9. **EV7:** EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.
10. **EV8:** EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.
11. **EV9:** EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.

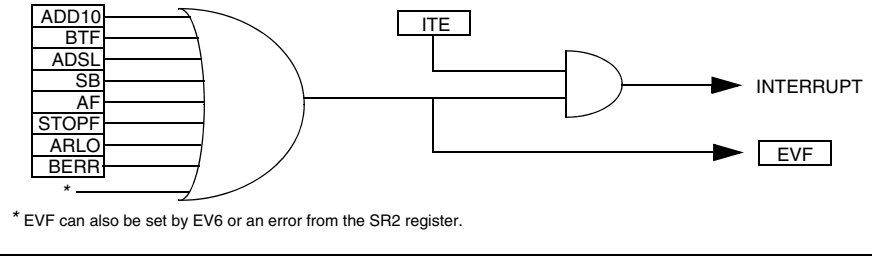
11.5.5 Low power modes

Table 46. Effect of low power modes on the I²C interface

Mode	Description
Wait	No effect on I²C interface. I²C interrupts cause the device to exit from Wait mode.
Halt	I²C registers are frozen. In Halt mode, the I²C interface is inactive and does not acknowledge data on the bus. The I²C interface resumes operation when the MCU is woken up by an interrupt with “exit from Halt mode” capability.

11.5.6 Interrupts

Figure 72. Event flags and interrupt generation



* EVF can also be set by EV6 or an error from the SR2 register.

Table 47. Description of interrupt events

Interrupt event(1)	Event flag	Enable control bit	Exit from Wait	Exit from Halt
10-bit Address Sent Event (Master mode)	ADD10	ITE	Yes	No
End of byte Transfer Event	BTF		Yes	No
Address Matched Event (Slave mode)	ADSL		Yes	No
Start Bit Generation Event (Master mode)	SB		Yes	No
Acknowledge Failure Event	AF		Yes	No
Stop Detection Event (Slave mode)	STOPF		Yes	No
Arbitration Lost Event (Multimaster configuration)	ARLO		Yes	No
Bus Error Event	BERR		Yes	No

1. The I²C interrupt events are connected to the same interrupt vector (see Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the I-bit in the CC register is reset (RIM instruction).

Bit 3 = **BTF** *Byte Transfer Finished bit*

This bit is set by hardware as soon as a byte is correctly received or transmitted with interrupt generation if ITE=1. It is cleared by software reading SR1 register followed by a read or write of DR register. It is also cleared by hardware when the interface is disabled (PE=0).

– Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. In case an address byte is sent, this bit is set only after the EV6 event (See *Figure 71*). BTF is cleared by reading SR1 register followed by writing the next byte in DR register.

– Following a byte reception, this bit is set after transmission of the acknowledge clock pulse if ACK=1. BTF is cleared by reading SR1 register followed by reading the byte from DR register.

The SCL line is held low while BTF=1.

0: byte transfer not done

1: byte transfer succeeded

Bit 2 = **ADSL** *Address matched bit (slave mode).*

This bit is set by hardware as soon as the received slave address matched with the OAR register content or a general call is recognized. An interrupt is generated if ITE=1. It is cleared by software reading SR1 register or by hardware when the interface is disabled (PE=0).

The SCL line is held low while ADSL=1.

0: Address mismatched or not received

1: Received address matched

Bit 1 = **M/SL** *Master/Slave bit*

This bit is set by hardware as soon as the interface is in Master mode (writing START=1). It is cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1). It is also cleared when the interface is disabled (PE=0).

0: Slave mode

1: Master mode

Bit 0 = **SB** *Start bit (master mode).*

This bit is set by hardware as soon as the Start condition is generated (following a write START=1). An interrupt is generated if ITE=1. It is cleared by software reading SR1 register followed by writing the address byte in DR register. It is also cleared by hardware when the interface is disabled (PE=0).

0: No Start condition

1: Start condition generated

12 Instruction set

12.1 ST7 addressing modes

The ST7 core features 17 different addressing modes which can be classified in seven main groups:

Table 61. Description of addressing modes

Addressing mode	Example
Inherent	nop
Immediate	ld A,#$55
Direct	ld A,$55
Indexed	ld A,($55,X)
Indirect	ld A,([$55],X)
Relative	jrne loop
Bit operation	bset byte,#5

The ST7 instruction set is designed to minimize the number of bytes required per instruction: To do so, most of the addressing modes may be subdivided in two submodes called long and short:

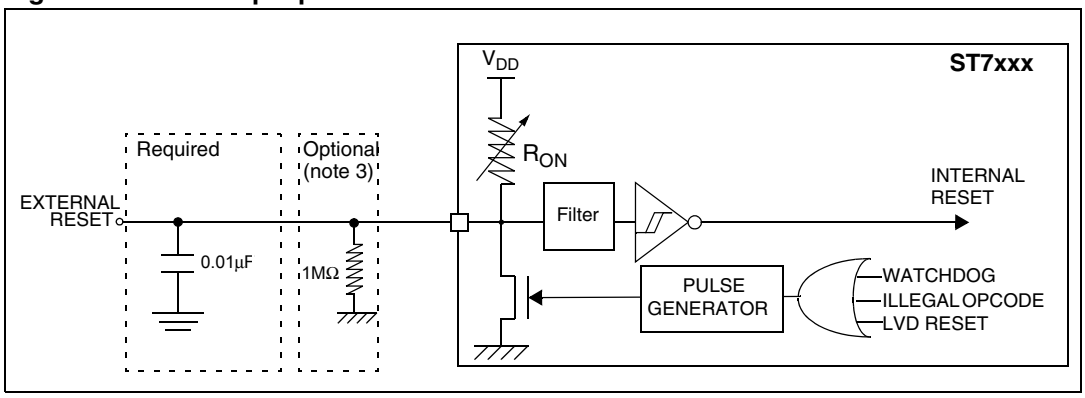
- Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.
- Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

The ST7 Assembler optimizes the use of long and short addressing modes.

Table 62. ST7 addressing mode overview

Mode			Syntax	Destination/ source	Pointer address	Pointer size	Length (bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed	ld A,(X)	00..FF			+ 0 (with X register) + 1 (with Y register)
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2

Figure 122. $\overline{\text{RESET}}$ pin protection when LVD is enabled



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 $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in *Section 13.10.1 on page 224*. 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 $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in *Section Table 70. on page 194*.
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.
3. In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the $\overline{\text{RESET}}$ 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).

Tips when using the LVD

- Check that all recommendations related to ICCCLK and reset circuit have been applied (see caution in *Table 2 on page 18* and notes above).
- Check that the power supply is properly decoupled (100 nF + 10 µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100 nF + 1 MΩ pull-down on the $\overline{\text{RESET}}$ pin.
- The capacitors connected on the $\overline{\text{RESET}}$ 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 10 nF pull-down on the $\overline{\text{RESET}}$ pin with a 5 µF to 20 µF capacitor."

14.3 Transfer of customer code

Customer code is made up of the FASTROM contents and the list of the selected options (if any). The FASTROM contents are to be sent on diskette, or by electronic means, with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh. The selected options are communicated to STMicroelectronics using the correctly completed OPTION LIST appended.

Refer to application note AN1635 for information on the counter listing returned by ST after code has been transferred.

The STMicroelectronics Sales organization will be pleased to provide detailed information on contractual points.

16 Revision history

Table 115. Document revision history

Date	Revision	Changes
08-Nov-2007	1	Initial release
31-Dec-2007	2	Added 16-bit timer on first page. Removed QFN40 pinout. Removed references to PLL x 8. Modified reset configuration for ICCCLK pin (*Table 2: ST7LITE49K2 device pin description on page 19*. Removed reference to ATCSR3 in *Table 3: Hardware register map on page 22*. Modified note 4 in *Section 4.4: ICC interface on page 27*. Modified *Figure 5: Typical ICC Interface on page 28*. Modified *: Break function on page 92*. Added BREAKCR2 in *Table 38: Register mapping and reset values on page 112*. Removed one block diagram in *Section 11.3: Lite timer 2 (LT2) on page 114*. Removed one figure in *Section 11.3.2 on page 114*. Modified *Figure 53: Lite timer 2 block diagram on page 114*. Removed bits 2:0 in LTCSR1 register in *Section 11.3.6: Register description on page 116*. Modified *Section 11.7.1: Introduction on page 176*. Modified *Table 75: Internal RC oscillator characteristics (5.0 V calibration) on page 202*. Modified *Table 76: Internal RC oscillator characteristics (3.3 V calibration) on page 203* . Modified *Table 77: Supply current characteristics on page 205*. Modified *Section 13.6.2: Crystal and ceramic resonator oscillators on page 214*. Added *Section 13.6.3: 32-MHz PLL on page 216*. Modified *Table 95: General characteristics on page 221* and *Table 96: Output driving current characteristics on page 223*. Modified *14.2: Device ordering information on page 243*. Added *14.3: Transfer of customer code on page 244*.
07-Feb-2008	3	Added reference to 16-bit timer on first page.
11-Feb-2009	4	Modified first page (2 analog comparators added) Modified *Figure 13: Clock management block diagram on page 41*, *Figure 37: Single timer mode (ENCNTR2=0) on page 85* and *Figure 38: Dual timer mode (ENCNTR2=1) on page 85* (32-MHz PLL added) Modified text below *Table 40: Description of interrupt events on page 111* Removed watchdog description in *Section 11.3.2: Main features on page 109* and *Section 11.3.3: Functional description on page 110* Modified *Section 11.8.3: Functional description on page 178* Modified *Section 13.6.2: Crystal and ceramic resonator oscillators on page 209* Modified *Section 15: Package mechanical data on page 241*