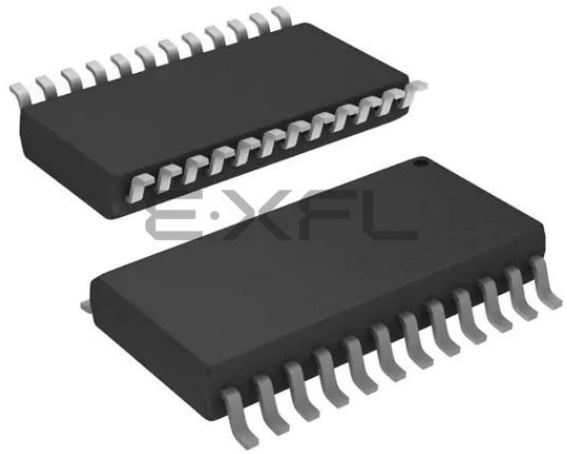


Welcome to E-XFL.COM

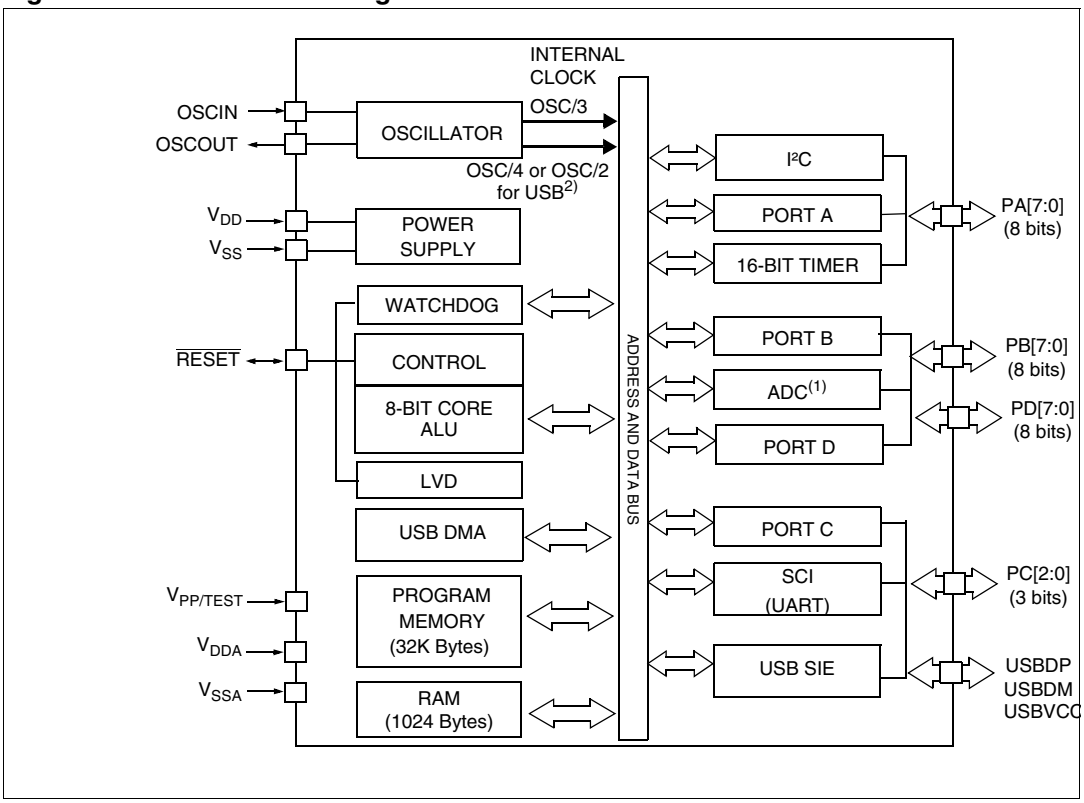
What is "Embedded - Microcontrollers"?

"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, SCI, USB
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	14
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 5.5V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	24-SOIC (0.295", 7.50mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f63be6m1

Figure 1. General block diagram



1. ADC channels:
   12 on 48-pin devices (Port B and Port D[3:0])
   8 on 34 and 32-pin devices (Port B)
   None on 24-pin devices
2. 12 or 24 MHz OSCIN frequency required to generate 6 MHz USB clock.
3. The drive from USBVCC is sufficient to only drive an external pull-up in addition to the internal transceiver.

Figure 4. 34-pin SO package pinout

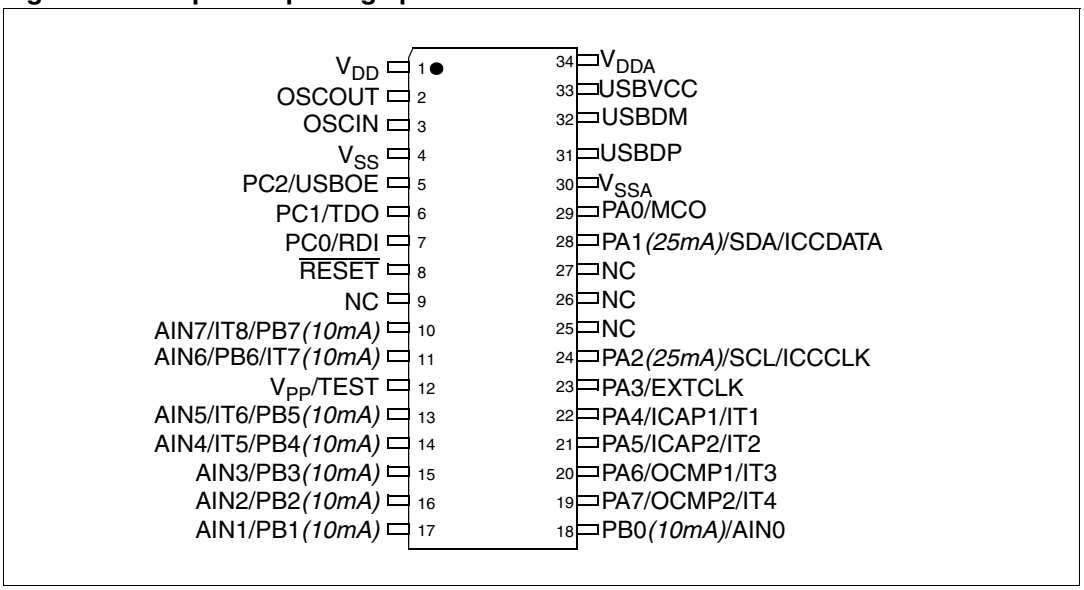


Figure 5. 32-pin SDIP package pinout

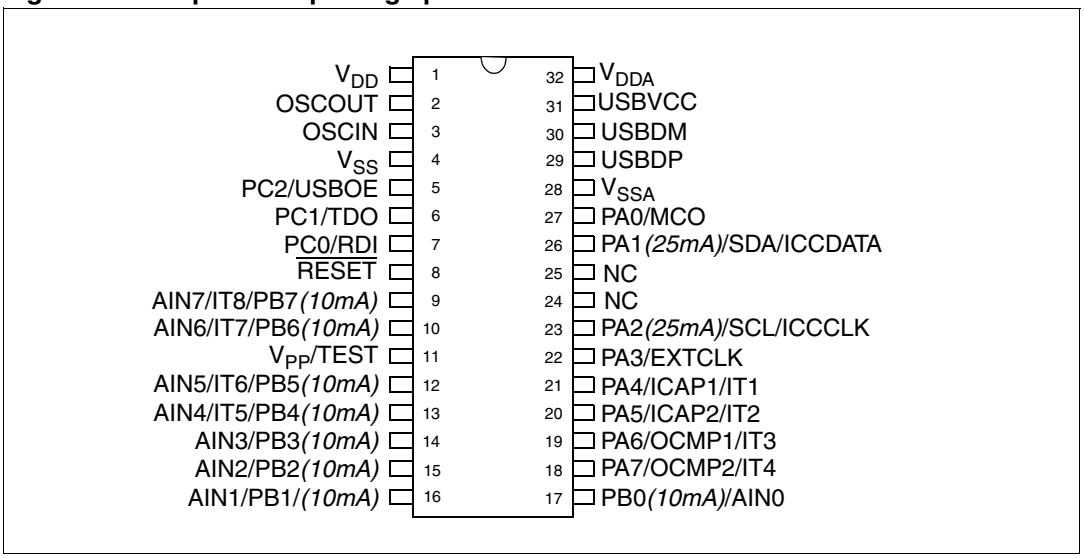
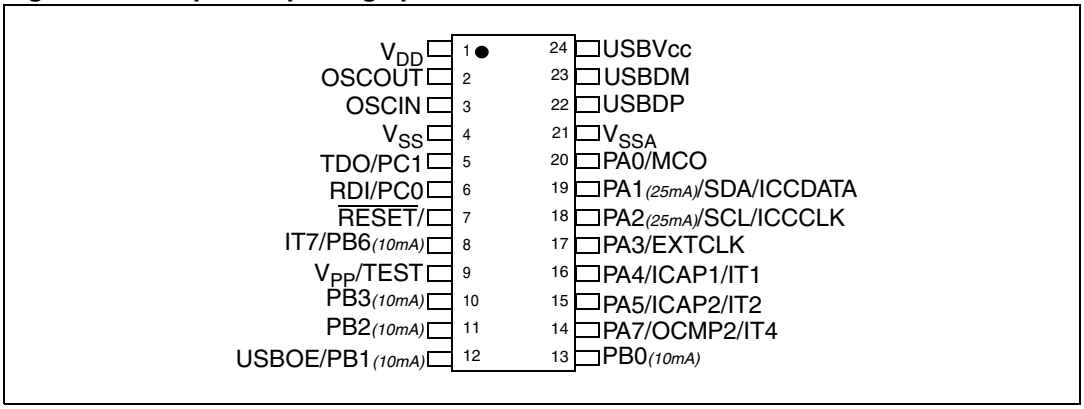


Figure 6. 24-pin SO package pinout



3 Register and memory map

As shown in *Figure 7*, the MCU is capable of addressing 32 Kbytes of memories and I/O registers.

The available memory locations consist of up to 1024 bytes of RAM including 64 bytes of register locations, and up to 32K bytes of user program memory in which the upper 32 bytes are reserved for interrupt vectors. The RAM space includes up to 128 bytes for the stack from 0100h to 017Fh.

The highest address bytes contain the user reset and interrupt vectors.

Caution: Memory locations noted “Reserved” must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 7. Memory map

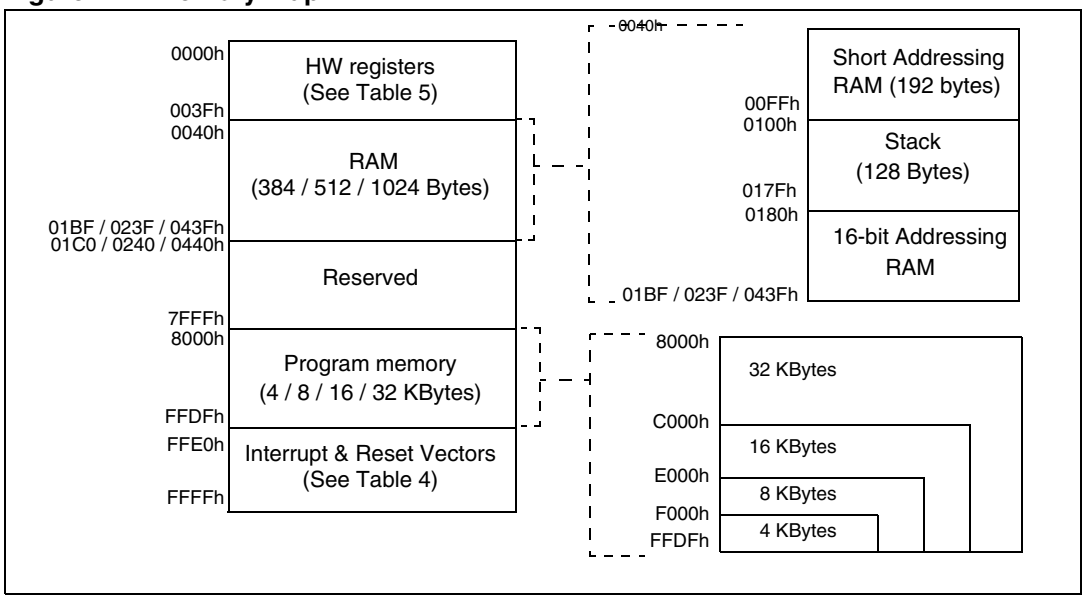


Table 5. Interrupt vector map

Vector address	Description	Masked	Remarks	Exit from Halt
FFE0h-FFEDh	Reserved area			
FFEEh-FFEFh	USB interrupt vector	I- bit	Internal interrupt	No
FFF0h-FFF1h	SCI interrupt vector	I- bit	Internal interrupt	No
FFF2h-FFF3h	I²C interrupt vector	I- bit	Internal interrupt	No
FFF4h-FFF5h	TIMER interrupt vector	I- bit	Internal interrupt	No
FFF6h-FFF7h	IT1 to IT8 interrupt vector	I- bit	External interrupt	Yes
FFF8h-FFF9h	USB End Suspend mode interrupt vector	I- bit	External interrupts	Yes
FFFAh-FFFBh	Flash start programming interrupt vector	I- bit	Internal interrupt	Yes
FFFCh-FFFDh	TRAP (software) interrupt vector	None	CPU interrupt	No
FFFEh-FFFFh	RESET vector	None		Yes

Table 6. Hardware register memory map (continued)

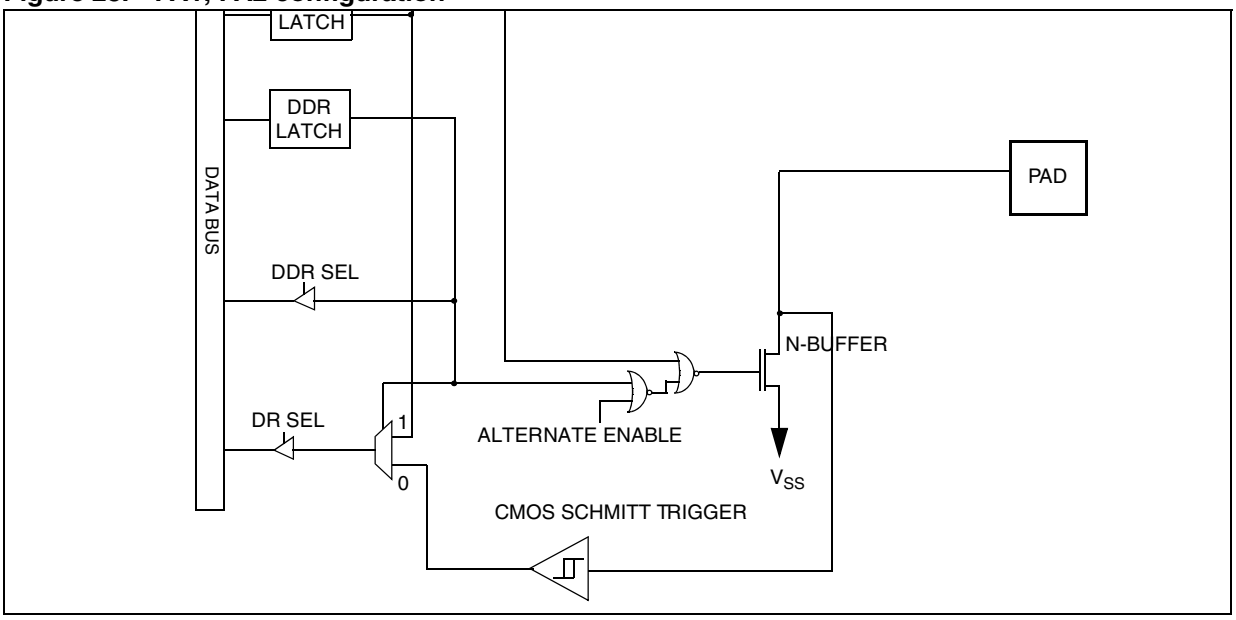
Address	Block	Register label	Register name	Reset status	Remarks
0025h	USB	USBPIDR	USB PID register	x0h	Read only
0026h		USBDMAR	USB DMA address register	xxh	R/W
0027h		USBIDR	USB Interrupt/DMA register	x0h	R/W
0028h		USBISTR	USB Interrupt Status register	00h	R/W
0029h		USBIMR	USB Interrupt Mask register	00h	R/W
002Ah		USBCTLR	USB Control register	06h	R/W
002Bh		USBDADDR	USB Device Address register	00h	R/W
002Ch		USBEP0RA	USB Endpoint 0 register A	0000 xxxxb	R/W
002Dh		USBEP0RB	USB Endpoint 0 register B	80h	R/W
002Eh		USBEP1RA	USB Endpoint 1 register A	0000 xxxxb	R/W
002Fh		USBEP1RB	USB Endpoint 1 register B	0000 xxxxb	R/W
0030h		USBEP2RA	USB Endpoint 2 register A	0000 xxxxb	R/W
0031h		USBEP2RB	USB Endpoint 2 register B	0000 xxxxb	R/W
0032h to 0036h	Reserved (5 bytes)				
0032h 0036h	Reserved (5 Bytes)				
0037h	Flash	FCSR	Flash Control /Status register	00h	R/W
0038h	Reserved (1 byte)				
0039h	I²C	I2CDR	I²C Data register	00h	R/W
003Ah			Reserved	-	
003Bh		I2COAR	I²C (7 Bits) Slave Address register	00h	R/W
003Ch		I2CCCR	I²C Clock Control register	00h	R/W
003Dh		I2CSR2	I²C 2nd Status register	00h	Read only
003Eh		I2CSR1	I²C 1st Status register	00h	Read only
003Fh		I2CCR	I²C Control register	00h	R/W

Table 12. PA1, PA2 description(1)

Port A	I / O		Alternate function	
	Input[1]	Output	Signal	Condition
PA1	without pull-up	Very high current open drain	SDA (I²C data)	I²C enable
PA2	without pull-up	Very high current open drain	SCL (I²C clock)	I²C enable

1. Reset state.

Figure 23. PA1, PA2 configuration



Output Compare

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

This function can be used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the output Compare register and the free running counter, the output compare function:

- Assigns pins with a programmable value if the OC*i*E bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers output Compare register 1 (OC1R) and output Compare register 2 (OC2R) contain the value to be compared to the counter register each timer clock cycle.

Table 20. OC/R register

	MS Byte	LS Byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*R value to 8000h.

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

Procedure

To use the output compare function, select the following in the CR2 register:

1. Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* signal.
2. Select the timer clock (CC[1:0]) (see *Table 24*).
3. Select the following in the CR1 register:
   a) Select the OLVL*i* bit to applied to the OCMP*i* pins after the match occurs.
   b) Set the OCIE bit to generate an interrupt if it is needed.

When a match is found between OC*i*R register and CR register:

- OCF*i* bit is set.
- The OCMP*i* pin takes OLVL*i* bit value (OCMP*i* pin latch is forced low during reset).
- A timer interrupt is generated if the OCIE bit is set in the CR1 register and the I bit is cleared in the CC register (CC).

The OC*i*R register value required for a specific timing application can be calculated using the following formula:

$$\Delta \text{ OC}i\text{R} = \frac{\Delta t * f_{CPU}}{\text{PRESC}}$$

Where:

Δt = Output compare period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

One Pulse mode

One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

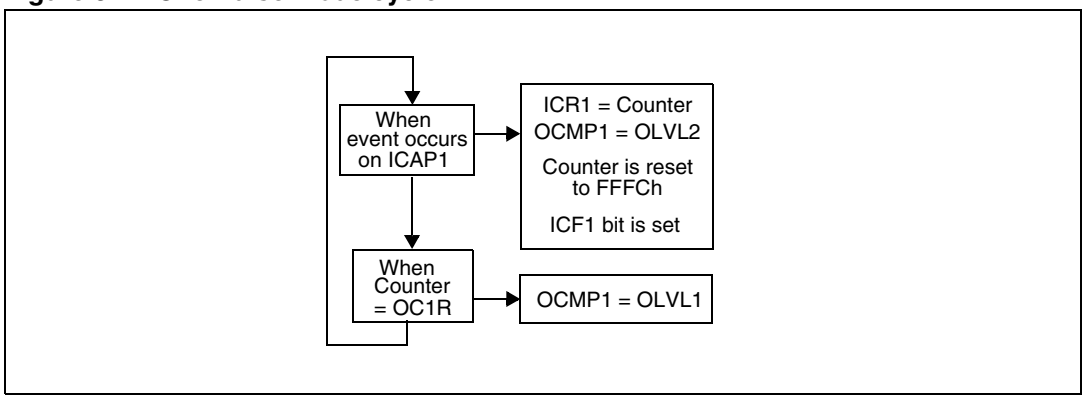
The One Pulse mode uses the input Capture1 function and the output Compare1 function.

Procedure

To use One Pulse mode:

1. Load the OC1R register with the value corresponding to the length of the pulse (see the formula in the opposite column).
2. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after the pulse.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin during the pulse.
   - Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as floating input).
3. Select the following in the CR2 register:
   - Set the OC1E bit, the OCMP1 pin is then dedicated to the output Compare 1 function.
   - Set the OPM bit.
   - Select the timer clock CC[1:0] (see *Table 24*).

Figure 37. One Pulse mode cycle



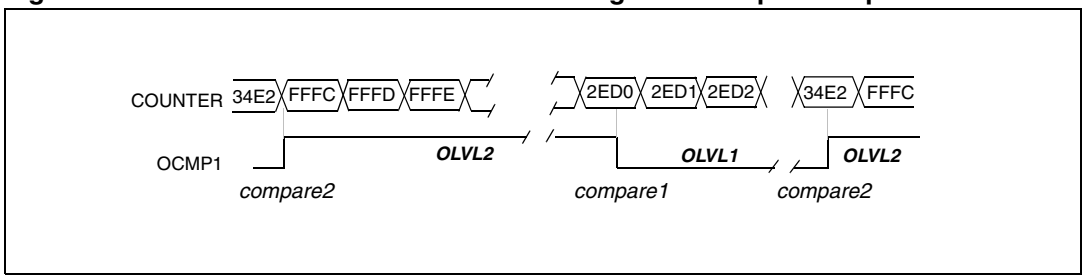
Then, on a valid event on the ICAP1 pin, the counter is initialized to FFFCh and OLVL2 bit is loaded on the OCMP1 pin, the ICF1 bit is set and the value FFFDh is loaded in the IC1R register.

Because the ICF1 bit is set when an active edge occurs, an interrupt can be generated if the ICIE bit is set.

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

1. Reading the SR register while the ICF*i* bit is set.
2. An access (read or write) to the IC*i*LR register.

Figure 39. Pulse Width modulation mode timing with 2 output Compare functions



1. OC1R = 2ED0h, OC2R = 34E2, OLVL1 = 0, OLVL2 = 1

On timers with only one output Compare register, a fixed frequency PWM signal can be generated using the output compare and the counter overflow to define the pulse length.

Pulse width modulation mode

Pulse width modulation (PWM) mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

Pulse Width Modulation mode uses the complete output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

In PWM mode, double buffering is implemented on the output compare registers. Any new values written in the OC1R and OC2R registers are taken into account only at the end of the PWM period (OC2) to avoid spikes on the PWM output pin (OCMP1).

Procedure

To use Pulse Width Modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1 = 0 and OLVL2 = 1) using the formula in the opposite column.
3. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC1R register.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the OC2R register.
4. Select the following in the CR2 register:
   - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
   - Set the PWM bit.
   - Select the timer clock (CC[1:0]) (see *Table 24*).

Control register 2 (SCICR2)

Reset value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Read/write							

7 TIE *Transmitter interrupt enable*.
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TDRE=1 in the SCISR register

6 TCIE *Transmission complete interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TC=1 in the SCISR register

5 **RIE** *Receiver interrupt enable*.
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever OR=1 or RDRF=1 in the SCISR register

4 ILIE *Idle line interrupt enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever IDLE=1 in the SCISR register.

3 TE *Transmitter enable.*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Note: During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.
When TE is set there is a 1 bit-time delay before the transmission starts.

Caution: The TDO pin is free for general purpose I/O only when the TE and RE bits are both cleared (or if TE is never set).

2 RE *Receiver enable.*
This bit enables the receiver. It is set and cleared by software.
0: Receiver is disabled
1: Receiver is enabled and begins searching for a start bit

1 RWU *Receiver wakeup.*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wakeup sequence is recognized.
0: Receiver in Active mode
1: Receiver in Mute mode
Note: Before selecting Mute mode (setting the RWU bit), the SCI must receive some data first, otherwise it cannot function in Mute mode with wakeup by idle line detection.

0 **SBK** *Send break.*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted
Note: If the SBK bit is set to "1" and then to "0", the transmitter will send a BREAK word at the end of the current word.

Data register (SCIDR)

Reset value: Undefined

This register contains the received or transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
Read/write							

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see *Figure 41*).
The RDR register provides the parallel interface between the input shift register and the internal bus (see *Figure 41*).

Following the address reception and after SR1 register has been read, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set
- EVF and BTF bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see *Figure 48* Transfer sequencing EV2).

Slave transmitter

Following the address reception and after the SR1 register has been read, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see *Figure 48* Transfer sequencing EV3).

When the acknowledge pulse is received, the EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Closing Slave communication

After the last data byte is transferred a Stop Condition is generated by the master. The interface detects this condition and sets EVF and STOPF bits with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR2 register (see *Figure 48* Transfer sequencing EV4).

Error cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.
  If it is a Stop, then the interface discards the data, released the lines and waits for another Start condition.
  If it is a Start, then the interface discards the data and waits for the next slave address on the bus.
- **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.
  The AF bit is cleared by reading the I2CSR2 register. However, if read before the completion of the transmission, the AF flag will be set again, thus possibly generating a new interrupt. Software must ensure either that the SCL line is back at 0 before reading the SR2 register, or be able to correctly handle a second interrupt during the 9th pulse of a transmitted byte.

Note: In case of errors, SCL line is not held low; however, the SDA line can remain low if the last bits transmitted are all 0. While AF=1, the SCL line may be held low due to SB or BTF flags that are set at the same time. It is then necessary to release both lines by software.

How to Release the SDA / SCL lines

Set and subsequently clear the STOP bit while BTF is set. The SDA/SCL lines are released after the transfer of the current byte.

Master mode

To switch from default Slave mode to Master mode, a Start condition generation is needed.

Start condition

11.6 8-bit A/D converter (ADC)

11.6.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 8-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 16 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

The result of the conversion is stored in a 8-bit Data register. The A/D converter is controlled through a Control/Status register.

11.6.2 Main features

- 8-bit conversion
- Up to 12 channels with multiplexed input
- Linear successive approximation
- Data register (DR) which contains the results
- Conversion complete status flag
- On/off bit (to reduce consumption)

The block diagram is shown in *Figure 50*.

11.6.3 Functional description

Analog power supply

V_{DDA} and V_{SSA} are the high and low level reference voltage pins. In some devices (refer to device pin out description) they are internally connected to the V_{DD} and V_{SS} pins.

Conversion accuracy may therefore be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines.

See electrical characteristics section for more details.

Software procedure

Refer to the control/status register (CSR) and data register (DR) in *Section 11.6.6* for the bit definitions and to *Figure 51* for the timings.

ADC configuration

The total duration of the A/D conversion is 12 ADC clock periods ($1/f_{ADC}=4/f_{CPU}$).

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the CSR register:

- Select the CH[3:0] bits to assign the analog channel to be converted.

ADC conversion

In the CSR register:

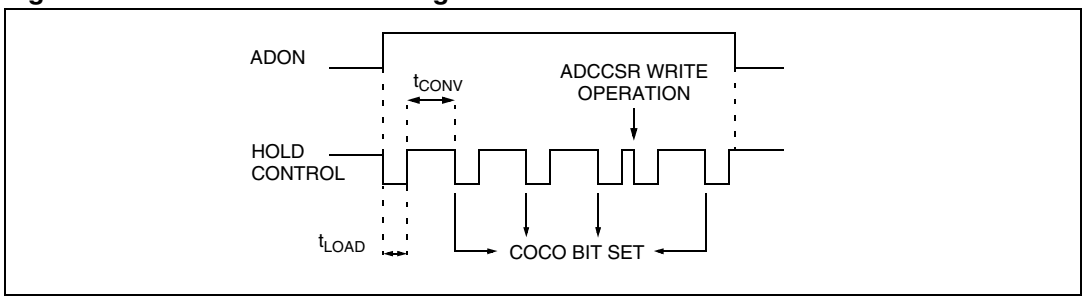
- Set the ADON bit to enable the A/D converter and to start the first conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

When a conversion is complete:

- The COCO bit is set by hardware.
- No interrupt is generated.
- The result is in the DR register and remains valid until the next conversion has ended.

A write to the CSR register (with ADON set) aborts the current conversion, resets the COCO bit and starts a new conversion.

Figure 51. ADC conversion timings



11.6.4 Low power modes

Table 44. Low power modes

Mode	Description
WAIT	No effect on A/D Converter
HALT	A/D Converter disabled. After wakeup from Halt mode, the A/D Converter requires a stabilization time before accurate conversions can be performed.

Note: The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed and between single shot conversions.

Table 51. Instructions supporting Direct, Indexed, Indirect and Indirect Indexed addressing modes (continued)

Long and Short instructions	Function
BCP	Bit Compare
Short Instructions only	**Function**
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations
BTJT, BTJF	Bit Test and Jump Operations
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles
CALL, JP	Call or Jump subroutine

12.1.7 Relative mode (Direct, Indirect)

This addressing mode is used to modify the PC register value by adding an 8-bit signed offset to it.

Table 52. Instructions supporting relative addressing mode

Available relative direct/Indirect instructions	Function
JRxx	Conditional Jump
CALLR	Call Relative

The relative addressing mode consists of two sub-modes:

Relative (Direct)

The offset follows the opcode.

Relative (Indirect)

The offset is defined in memory, of which the address follows the opcode.

13.10.3 I²C interface

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SDAI and SCLI).

The ST7 I²C interface meets the requirements of the standard I²C communication protocol described in the following table.

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 76. I²C characteristics

Symbol	Parameter	Standard mode I²C(1)		Fast mode I²C(1)(2)		Unit
		Min	Max	Min	Max	
$t_{w(SCLL)}$	SCL clock low time	4.7	-	1.3	-	µs
$t_{w(SCLH)}$	SCL clock high time	4.0	-	0.6	-	
$t_{su(SDA)}$	SDA setup time	250	-	100	-	ns
$t_{h(SDA)}$	SDA data hold time	0(3)	-	0(4)	900(3)	
$t_{r(SDA)}$ $t_{r(SCL)}$	SDA and SCL rise time	-	1000	20+0.1C_b	300	
$t_{f(SDA)}$ $t_{f(SCL)}$	SDA and SCL fall time	-	300	20+0.1C_b	300	
$t_{h(STA)}$	START condition hold time	4.0	-	0.6	-	µs
$t_{su(STA)}$	Repeated START condition setup time	4.7	-	0.6	-	
$t_{su(STO)}$	STOP condition setup time	4.0	-	0.6	-	µs
$t_{w(STO:STA)}$	STOP to START condition time (bus free)	4.7	-	1.3	-	µs
C_b	Capacitive load for each bus line	-	400	-	400	pF

1. Data based on standard I²C protocol requirement, not tested in production.
2. At 4 MHz f_{CPU}, max.I²C speed (400 kHz) is not achievable. In this case, max. I²C speed will be approximately 260 KHz.
3. The maximum hold time of the START condition has only to be met if the interface does not stretch the low period of SCL signal.
4. The device must internally provide a hold time of at least 300ns for the SDA signal in order to bridge the undefined region of the falling edge of SCL.

Figure 77. Typical application with ADC

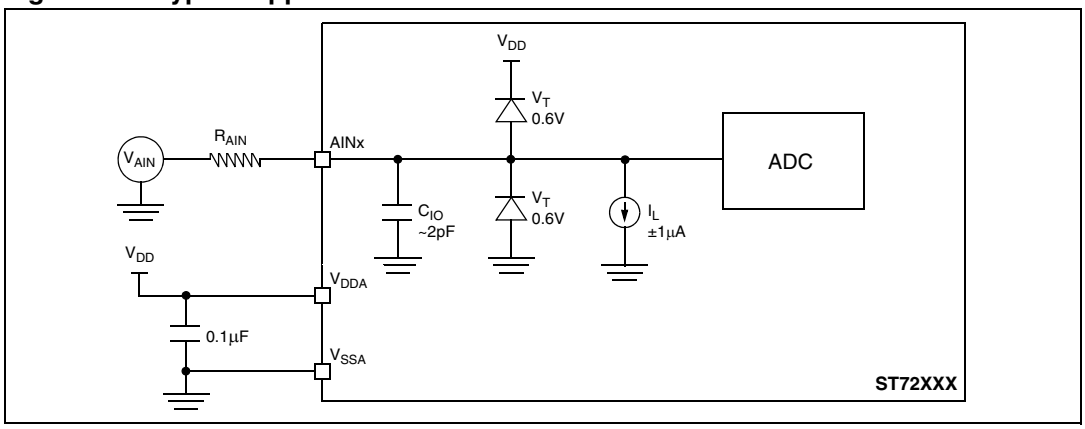


Table 79. ADC accuracy with V_{DD}=5 V, f_{CPU}= 8 MHz, f_{ADC}=4 MHz, R_{AIN}< 10 kΩ

Symbol	Parameter	Typ	Max(1)(2)
$\|E_T\|$	Total unadjusted error(3)	1.5	2
$\|E_{OI}\|$	Offset error(3)	0.5	1
$\|E_{GI}\|$	Gain Error(3)	0.5	1.5
$\|E_D\|$	Differential linearity error(3)	1	1.5
$\|E_L\|$	Integral linearity error(3)	1	1.5

1. Data based on characterization results over the whole temperature range, not tested in production.
2. Data based on characterization results, to guarantee 99.73% within ± max value from 0 to 70 °C (± 3s distribution limits).
3. ADC Accuracy vs. Negative Injection Current:
For I_{INJ-}=0.8mA, the typical leakage induced inside the die is 1.6µA and the effect on the ADC accuracy is a loss of 1 LSB for each 10KΩ increase of the external analog source impedance. This effect on the ADC accuracy has been observed under worst-case conditions for injection:
- negative injection
- injection to an input with analog capability, adjacent to the enabled Analog input
- at 5V V_{DD} supply, and worst case temperature.

Figure 80. 34-pin plastic small outline package, 300-mil width, package outline

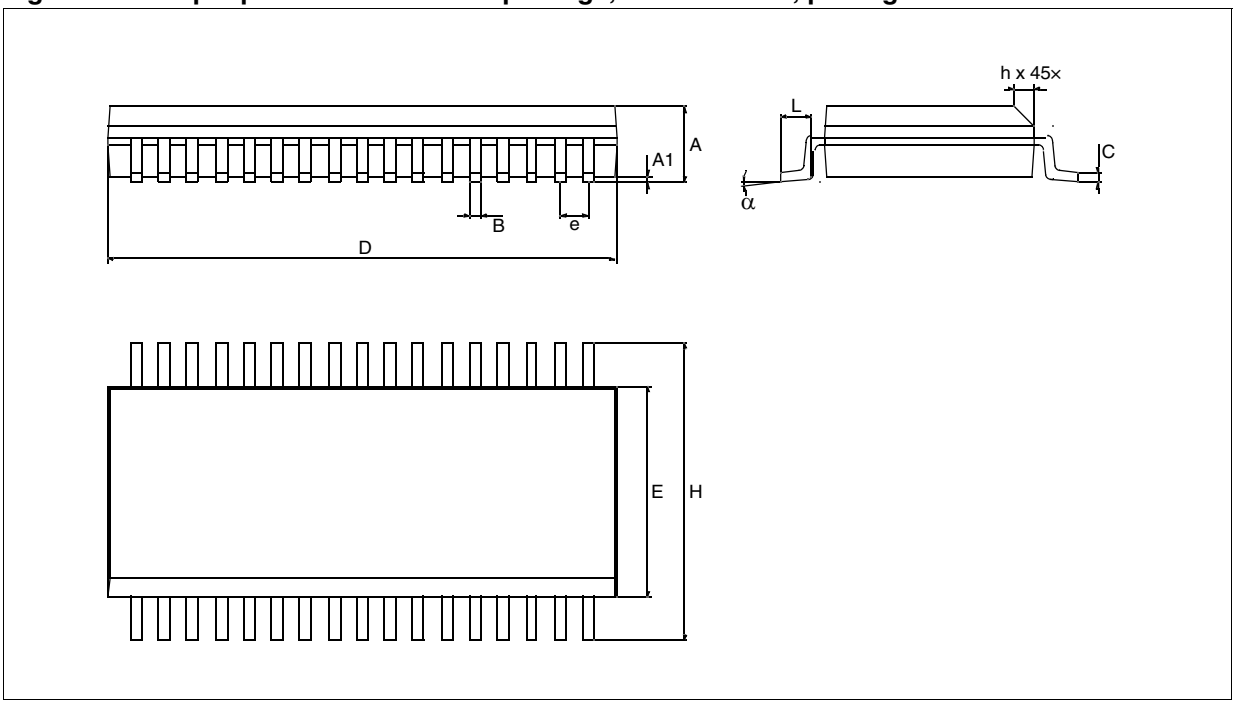


Table 81. 34-pin plastic small outline package, 300-mil width, package mechanical data

Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	2.464		2.642	0.0970		0.1040
A1	0.127		0.292	0.0050		0.0120
B	0.356		0.483	0.0140		0.0190
C	0.231		0.318	0.0090		0.0130
D	17.729		18.059	0.6980		0.7110
E	7.417		7.595	0.2920		0.2990
e		1.016			0.0400	
H	10.160		10.414	0.4000		0.4100
h	0.635		0.737	0.0250		0.0290
α	0°		8°	0°		8°
L	0.610		1.016	0.0240		0.0400
	Number of pins					
N	34					

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Table 86. Supported order codes (continued)

Sales type[1][2]	Program memory (bytes)	RAM (bytes)	Package
ST72F63BH2T1	8K Flash	384	LQFP48
ST72F63BK2U1			QFN40
ST72F63BK2M1			SO34
ST72F63BK2B1			SDIP32
ST72F63BE2M1			SO24
ST72F63BK1M1	4K Flash	384	SO34
ST72F63BK1B1			SDIP32
ST72F63BE1M1			SO24
ST7263BK2M1/xxx	8K ROM	384	SO34
ST7263BK2B1/xxx			SDIP32
ST7263BK1M1/xxx	4K ROM	384	SO34
ST7263BK1B1/xxx			SDIP32

1. /xxx stands for the ROM code name assigned by STMicroelectronics.
2. Contact ST sales office for FASTROM product availability.

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 Evaluation tools and starter kits

ST offers complete, affordable **starter kits** and full-featured **evaluation boards** that allow you to evaluate microcontroller features and quickly start developing ST7 applications. Starter kits are complete, affordable hardware/software tool packages that include features and samples to to help you quickly start developing your application. ST evaluation boards are open-design, embedded systems, which are developed and documented to serve as references for your application design. They include sample application software to help you demonstrate, learn about and implement your ST7's features.

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 16K of code.

The range of hardware tools includes full-featured **ST7-EMU3 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

16.5 Halt mode power consumption with ADC on

Description

If the A/D converter is being used when Halt mode is entered, the power consumption in Halt mode may exceed the maximum specified in the datasheet.

Workaround

Switch off the ADC by software (ADON=0) before executing a HALT instruction.

16.6 SCI wrong BREAK duration

Description

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 20 bits instead of 10 bits if M=0
- 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may generate one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (fCPU=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

1. Disable interrupts
2. Reset and Set TE (IDLE request)
3. Set and Reset SBK (Break Request)
4. Re-enable interrupts

Figure 85. Identifying silicon revision from device marking and box label

The silicon revision can be identified either by Rev letter or obtained via a trace code.

1. Identify the silicon revision letter from either the device package or the box label.
For example, "**B**", etc.

2. If the revision letter is not present, obtain the silicon revision by contacting your local ST office with the trace code information printed on either the box label or the device