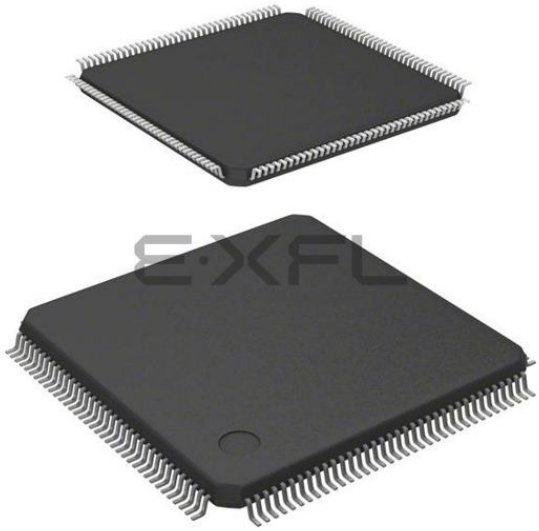


Welcome to E-XFL.COM

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	ST10
Core Size	16-Bit
Speed	40MHz
Connectivity	ASC, CANbus, EBI/EMI, I²C, SSC, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	111
Program Memory Size	832KB (832K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	68K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 24x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	144-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st10f276-4tx3

Contents

1 Introduction 13

2 Pin data 16

3 Functional description 23

4 Internal Flash memory 24

4.1 Overview 24

4.2 Functional description 24

4.2.1 Structure 24

4.2.2 Modules structure 25

4.2.3 Low power mode 27

4.2.4 Write operation 27

4.2.5 Power supply drop 28

4.3 Registers description 28

4.3.1 Flash control register 0 low 28

4.3.2 Flash control register 0 high 29

4.3.3 Flash control register 1 low 31

4.3.4 Flash control register 1 high 32

4.3.5 Flash data register 0 low 33

4.3.6 Flash data register 0 high 33

4.3.7 Flash data register 1 low 33

4.3.8 Flash data register 1 high 34

4.3.9 Flash address register low 34

4.3.10 Flash address register high 35

4.3.11 Flash error register 35

4.3.12 XFlash interface control register 36

4.4 Protection strategy 37

4.4.1 Protection registers 37

4.4.2 Flash non-volatile write protection X register low 37

4.4.3 Flash non-volatile write protection X register high 38

4.4.4 Flash non-volatile write protection I register low 38

4.4.5 Flash non-volatile write protection I register high 38

4.4.6 Flash non-volatile access protection register 0 39

Erase suspend, program and resume

A Sector Erase operation can be suspended in order to program (Word or Double Word) another sector.

Example: Sector Erase of sector B3F1 of Bank 3 in XFLASH Module.

```
FCR0H|= 0x0800;/*Set SER in FCR0H*/
FCR1H|= 0x0002;/*Set B3F1*/
FCR0H|= 0x8000;/*Operation start*/
```

Example: Sector Erase Suspend.

```
FCR0H|= 0x4000;/*Set SUSP in FCR0H*/
do        /*Loop to wait for LOCK=0 and WMS=0*/
{tmp1 = FCR0L;
 tmp2 = FCR0H;
} while ((tmp1 && 0x0010) || (tmp2 && 0x8000));
```

Example: Word Program of data 0x5555AAAA at address 0x0C5554 in XFLASH module.

```
FCR0H&= 0xBFFF;/*Rst SUSP in FCR0H*/
FCR0H|= 0x2000;/*Set WPG in FCR0H*/
FARL = 0x5554; /*Load Add in FARL*/
FARH = 0x000C; /*Load Add in FARH*/
FDR0L = 0xAAAA; /*Load Data in FDR0L*/
FDR0H = 0x5555; /*Load Data in FDR0H*/
FCR0H|= 0x8000; /*Operation start*/
```

Once the Program operation is finished, the Erase operation can be resumed in the following way:

```
FCR0H|= 0x0800;/*Set SER in FCR0H*/
FCR0H|= 0x8000;/*Operation resume*/
```

Notice that during the Program Operation in Erase suspend, bits SER and SUSP are low. A Word or Double Word Program during Erase Suspend cannot be suspended.

To summarize:

- A Sector Erase can be suspended by setting SUSP bit
- To perform a Word Program operation during Erase Suspend, firstly bits SUSP and SER must be reset, then bit WPG and WMS can be set
- To resume the Sector Erase operation bit SER must be set again
- In any case it is forbidden to start any write operation with SUSP bit already set

Set protection

Example 1: Enable Write Protection of sectors B0F3-0 of Bank 0 in IFLASH module.

```
FCR0H|= 0x0100;/*Set SPR in FCR0H*/
FARL = 0xDFB4;/*Load Add of register FNVWPIRL in FARL*/
FARH = 0x000E;/*Load Add of register FNVWPIRL in FARH*/
FDR0L = 0xFFF0;/*Load Data in FDR0L*/
FDR0H = 0xFFFF;/*Load Data in FDR0H*/
FCR0H|= 0x8000;/*Operation start*/
```

Notice that bit SMOD of FCR0H must not be set, since Write Protection bits of IFLASH Module are stored in Test-Flash (XFLASH Module).

The values for *operand0, operand1* and the signature should be such that the sequence shown in the figure below is successfully executed.

```
MOV    Rx, CheckBlock1Addr; 00`0000h for standard reset
ADD    Rx, CheckBlock2Addr; 00`1FFCh for standard reset
CPLB   RLx                ; 1s complement of the lower
                          ; byte of the sum
CMP    Rx, CheckBlock3Addr; 00`1FFEh for standard reset
```

5.6.9 Alternate boot user software aspects

User defined alternate boot code must start at 09'0000h. A new SFR created on the ST10F276E indicates that the device is running in Alternate Boot Mode: Bit 5 of EMUCON (mapped at 0xFE0Ah) is set when the alternate boot is selected by the reset configuration. All the other bits are ignored when checking the content of this register to read the value of bit 5.

This bit is a read-only bit. It remains set until the next software or hardware reset.

5.6.10 EMUCON register

EMUCON (FE0Ah / 05h) SFR Reset value: - xxh:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-										ABM					
-										R	-				

Table 36. ABM bit description

Bit	Function
ABM	ABM Flag (or TMOD3) 0: Alternate Boot Mode is not selected by reset configuration on P0L[5..4] 1: Alternate Boot Mode is selected by reset configuration on P0L[5..4]: This bit is set if P0L[5..4] = '01' during hardware reset.

5.6.11 Internal decoding of test modes

The test mode decoding logic is located inside the ST10F276E Bus Controller.

The decoding is as follows:

- Alternate Boot Mode decoding: ($\overline{\text{P0L.5}}$ & P0L.4)
- Standard Bootstrap decoding: (P0L.5 & $\overline{\text{P0L.4}}$)
- Normal operation: (P0L.5 & P0L.4)

The other configurations select ST internal test modes.

Table 40. Interrupt sources (continued)

Source of interrupt or PEC service request	Request flag	Enable flag	Interrupt vector	Vector location	Trap number
GPT2 Timer 6	T6IR	T6IE	T6INT	00'0098h	26h
GPT2 CAPREL Register	CRIR	CRIE	CRINT	00'009Ch	27h
A/D Conversion Complete	ADCIR	ADCIE	ADCINT	00'00A0h	28h
A/D Overrun Error	ADEIR	ADEIE	ADEINT	00'00A4h	29h
ASC0 Transmit	S0TIR	S0TIE	S0TINT	00'00A8h	2Ah
ASC0 Transmit Buffer	S0TBIR	S0TBIE	S0TBINT	00'011Ch	47h
ASC0 Receive	S0RIR	S0RIE	S0RINT	00'00ACh	2Bh
ASC0 Error	S0EIR	S0EIE	S0EINT	00'00B0h	2Ch
SSC Transmit	SCTIR	SCTIE	SCTINT	00'00B4h	2Dh
SSC Receive	SCRIR	SCRIE	SCRINT	00'00B8h	2Eh
SSC Error	SCEIR	SCEIE	SCEINT	00'00BCh	2Fh
PWM Channel 0...3	PWMIR	PWMIE	PWMINT	00'00FCh	3Fh
See paragraph *8.1*	XP0IR	XP0IE	XP0INT	00'0100h	40h
See paragraph *8.1*	XP1IR	XP1IE	XP1INT	00'0104h	41h
See paragraph *8.1*	XP2IR	XP2IE	XP2INT	00'0108h	42h
See paragraph *8.1*	XP3IR	XP3IE	XP3INT	00'010Ch	43h

Hardware traps are exceptions or error conditions that arise during run-time. They cause immediate non-maskable system reaction similar to a standard interrupt service (branching to a dedicated vector table location).

The occurrence of a hardware trap is additionally signified by an individual bit in the trap flag register (TFR). Except when another higher prioritized trap service is in progress, a hardware trap will interrupt any other program execution. Hardware trap services cannot not be interrupted by standard interrupt or by PEC interrupts.

8.1 X-Peripheral interrupt

The limited number of X-Bus interrupt lines of the present ST10 architecture, imposes some constraints on the implementation of the new functionality. In particular, the additional X-Peripherals SSC1, ASC1, I^2C, PWM1 and RTC need some resources to implement interrupt and PEC transfer capabilities. For this reason, a multiplexed structure for the interrupt management is proposed. In *Figure 16*, the principle is explained through a simple diagram, which shows the basic structure replicated for each of the four X-interrupt available vectors (XP0INT, XP1INT, XP2INT and XP3INT).

It is based on a set of 16-bit registers XIRxSEL (x=0,1,2,3), divided in two portions each:

- Byte High XIRxSEL[15:8] Interrupt Enable bits
- Byte Low XIRxSEL[7:0] Interrupt Flag bits

10.2 GPT2

The GPT2 module provides precise event control and time measurement. It includes two timers (T5, T6) and a capture/reload register (CAPREL). Both timers can be clocked with an input clock which is derived from the CPU clock via a programmable prescaler or with external signals. The count direction (up/down) for each timer is programmable by software or may additionally be altered dynamically by an external signal on a port pin (TxEUD). Concatenation of the timers is supported via the output toggle latch (T6OTL) of timer T6 which changes its state on each timer overflow/underflow.

The state of this latch may be used to clock timer T5, or it may be output on a port pin (T6OUT). The overflow / underflow of timer T6 can additionally be used to clock the CAPCOM timers T0 or T1, and to cause a reload from the CAPREL register. The CAPREL register may capture the contents of timer T5 based on an external signal transition on the corresponding port pin (CAPIN), and timer T5 may optionally be cleared after the capture procedure. This allows absolute time differences to be measured or pulse multiplication to be performed without software overhead.

The capture trigger (timer T5 to CAPREL) may also be generated upon transitions of GPT1 timer T3 inputs T3IN and/or T3EUD. This is advantageous when T3 operates in Incremental Interface Mode.

Table 48 and *Table 49* list the timer input frequencies, resolution and periods for each pre-scaler option at 40 MHz and 64 MHz CPU clock respectively.

Table 48. GPT2 timer input frequencies, resolutions and periods at 40 MHz

f_{CPU} = 40MHz	Timer input selection T5I / T6I							
	000b	001b	010b	011b	100b	101b	110b	111b
Prescaler factor	4	8	16	32	64	128	256	512
Input frequency	10 MHz	5 MHz	2.5 MHz	1.25 MHz	625 kHz	312.5 kHz	156.25 kHz	78.125 kHz
Resolution	100ns	200ns	400ns	0.8µs	1.6µs	3.2µs	6.4µs	12.8µs
Period maximum	6.55ms	13.1ms	26.2ms	52.4ms	104.8ms	209.7ms	419.4ms	838.9ms

Table 49. GPT2 timer input frequencies, resolutions and periods at 64 MHz

f_{CPU} = 64MHz	Timer input selection T5I / T6I							
	000b	001b	010b	011b	100b	101b	110b	111b
Prescaler factor	4	8	16	32	64	128	256	512
Input frequency	16 MHz	8 MHz	4 MHz	2 MHz	1 kHz	500 kHz	250 kHz	128 kHz
Resolution	62.5ns	125ns	250ns	0.5µs	1.0µs	2.0µs	4.0µs	8.0µs
Period maximum	4.1ms	8.2ms	16.4ms	32.8ms	65.5ms	131.1ms	262.1ms	524.3ms

15 I2C interface

The integrated I^2C Bus Module handles the transmission and reception of frames over the two-line SDA/SCL in accordance with the I^2C Bus specification. The I^2C Module can operate in slave mode, in master mode or in multi-master mode. It can receive and transmit data using 7-bit or 10-bit addressing. Data can be transferred at speeds up to 400 Kbit/s (both Standard and Fast I^2C bus modes are supported).

The module can generate three different types of interrupt:

- Requests related to bus events, like start or stop events, arbitration lost, etc.
- Requests related to data transmission
- Requests related to data reception

These requests are issued to the interrupt controller by three different lines, and identified as Error, Transmit, and Receive interrupt lines.

When the I^2C module is enabled by setting bit XI2CEN in XPERCON register, pins P4.4 and P4.7 (where SCL and SDA are respectively mapped as alternate functions) are automatically configured as bidirectional open-drain: the value of the external pull-up resistor depends on the application. P4, DP4 and ODP4 cannot influence the pin configuration.

When the I^2C cell is disabled (clearing bit XI2CEN), P4.4 and P4.7 pins are standard I/ O controlled by P4, DP4 and ODP4.

The speed of the I^2C interface may be selected between Standard mode (0 to 100 kHz) and Fast I^2C mode (100 to 400 kHz).

Figure 21. Connection to single CAN bus via common CAN transceivers

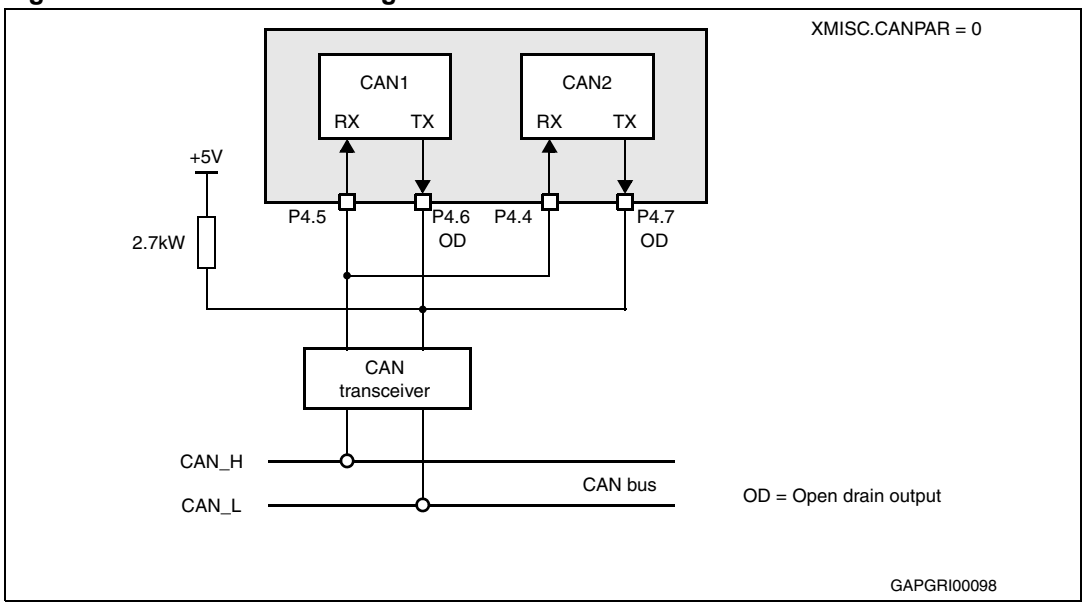


Figure 24. Asynchronous power-on RESET ($\overline{EA}$ = 1)

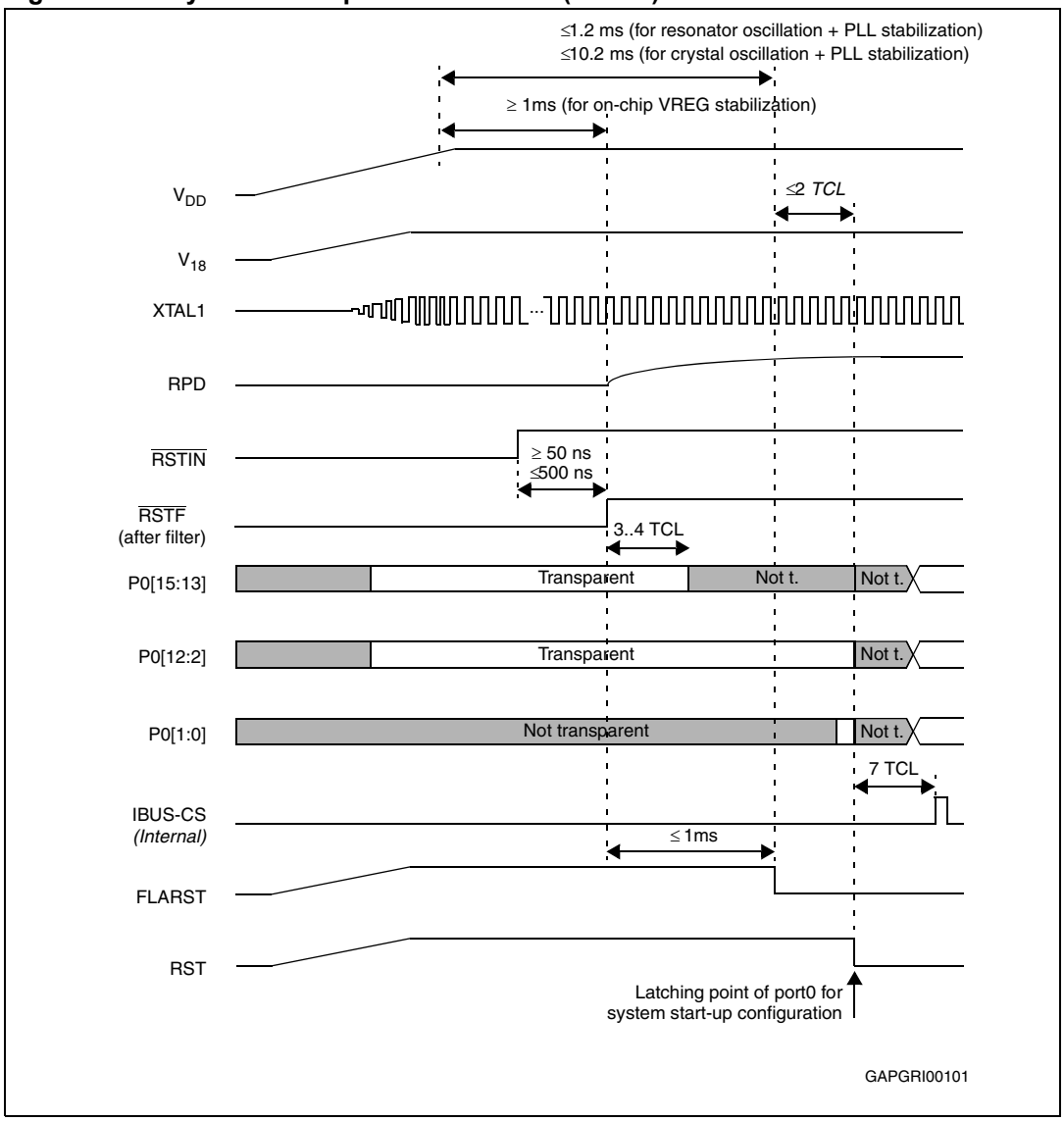
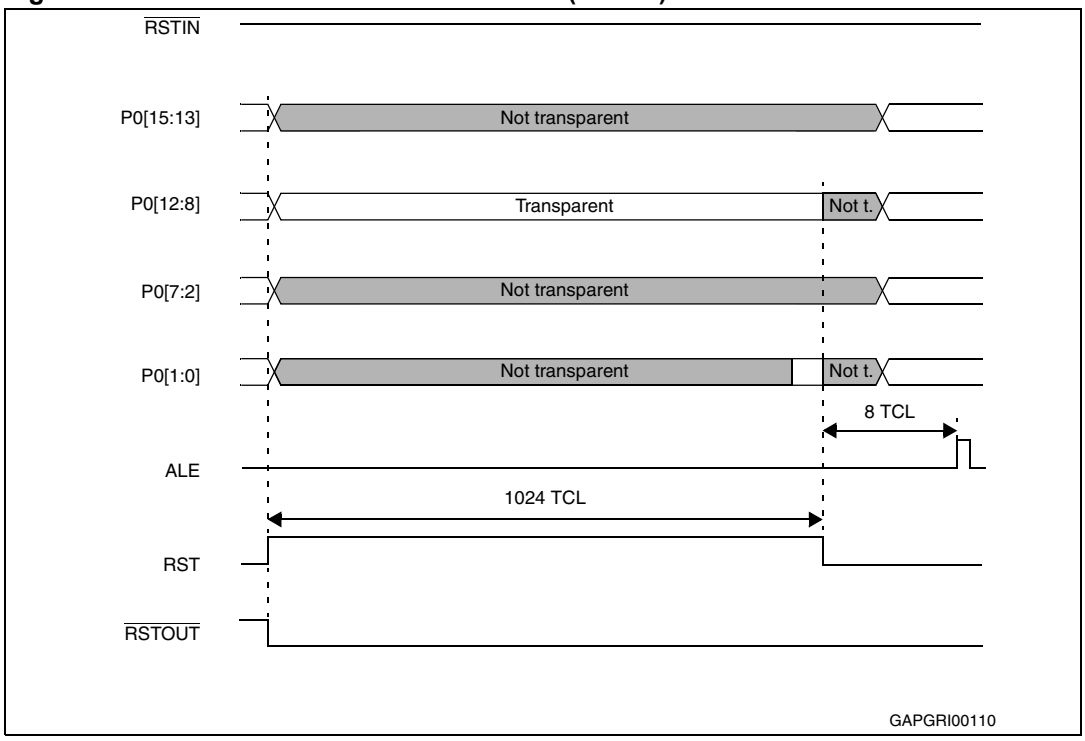


Figure 33. SW / WDT unidirectional RESET ($\overline{EA}$ = 0)



19.6 Bidirectional reset

As shown in the previous sections, the $\overline{RSTOUT}$ pin is driven active (low level) at the beginning of any reset sequence (synchronous/asynchronous hardware, software and watchdog timer resets). $\overline{RSTOUT}$ pin stays active low beyond the end of the initialization routine, until the protected EINIT instruction (End of Initialization) is completed.

The Bidirectional Reset function is useful when external devices require a reset signal but cannot be connected to $\overline{RSTOUT}$ pin, because $\overline{RSTOUT}$ signal lasts during initialization. It is, for instance, the case of external memory running initialization routine before the execution of EINIT instruction.

Bidirectional reset function is enabled by setting bit 3 (BDRSTEN) in SYSCON register. It only can be enabled during the initialization routine, before EINIT instruction is completed.

When enabled, the open drain of the $\overline{RSTIN}$ pin is activated, pulling down the reset signal, for the duration of the internal reset sequence (synchronous/asynchronous hardware, synchronous software and synchronous watchdog timer resets). At the end of the internal reset sequence the pull down is released and:

- After a Short Synchronous Bidirectional Hardware Reset, if $\overline{RSTF}$ is sampled low 8 TCL periods after the internal reset sequence completion (refer to *Figure 28* and *Figure 29*), the Short Reset becomes a Long Reset. On the contrary, if $\overline{RSTF}$ is sampled high the device simply exits reset state.
- After a Software or Watchdog Bidirectional Reset, the device exits from reset. If $\overline{RSTF}$ remains still low for at least 4 TCL periods (minimum time to recognize a Short Hardware reset) after the reset exiting (refer to *Figure 34* and *Figure 35*), the Software or Watchdog Reset become a Short Hardware Reset. On the contrary, if $\overline{RSTF}$ remains low for less than 4 TCL, the device simply exits reset state.

Figure 34. SW / WDT bidirectional RESET ($\overline{EA}$ = 1)

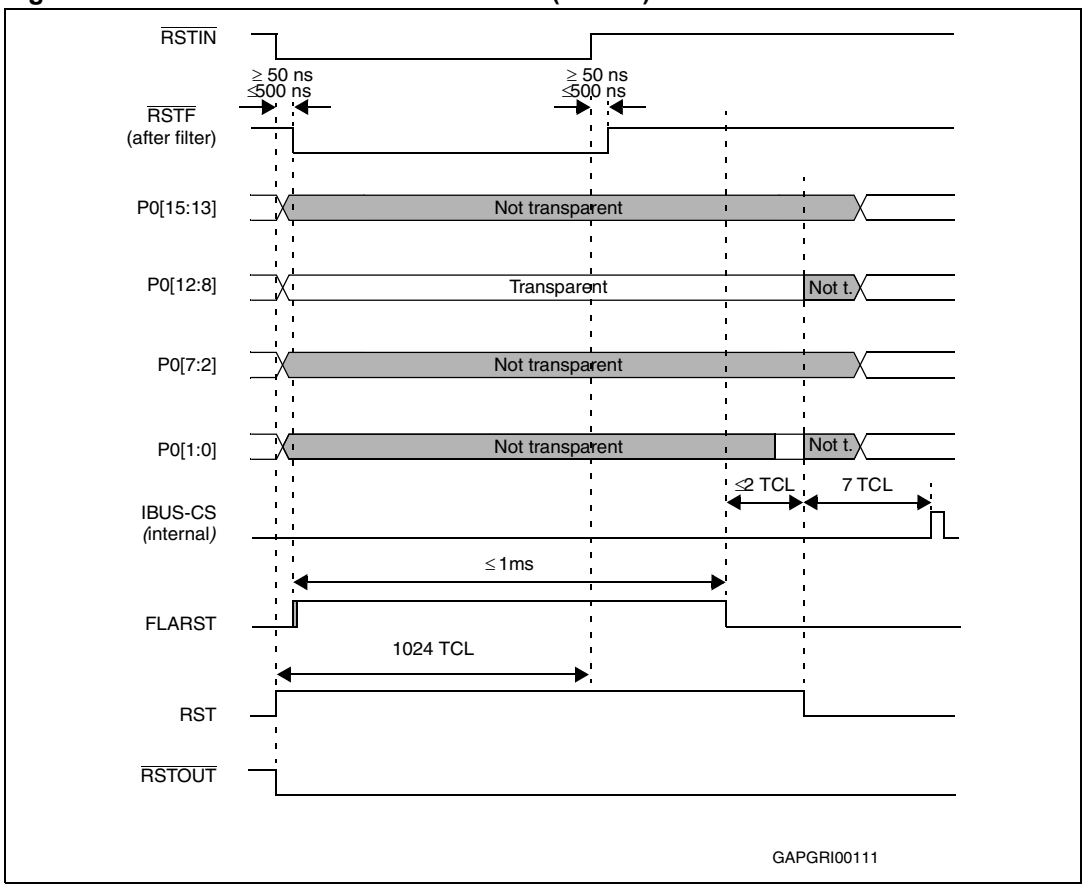


Table 61. Reset event (continued)

Event	RPD	EA	Bidir	Synch. Asynch.	RSTIN		WDTCON flags				
					Min	Max	PONR	LHWR	SHWR	SWR	WDTR
Software Reset (2)	x	0	N	Synch.	Not activated		0	0	0	1	0
	x	0	N	Synch.	Not activated		0	0	0	1	0
	0	1	Y	Synch.	Not activated		0	0	0	1	0
	1	1	Y	Synch.	Activated by internal logic for 1024 TCL		0	0	0	1	0
Watchdog Reset (2)	x	0	N	Synch.	Not activated		0	0	0	1	1
	x	0	N	Synch.	Not activated		0	0	0	1	1
	0	1	Y	Synch.	Not activated		0	0	0	1	1
	1	1	Y	Synch.	Activated by internal logic for 1024 TCL		0	0	0	1	1

1. It can degenerate into a Long Hardware Reset and consequently differently flagged (see *Section 19.3* for details).
2. When Bidirectional is active (and with RPD=0), it can be followed by a Short Hardware Reset and consequently differently flagged (see *Section 19.6* for details).

The start-up configurations and some system features are selected on reset sequences as described in *Table 62* and *Figure 42*.

Table 62 describes what is the system configuration latched on PORT0 in the six different reset modes. *Figure 42* summarizes the state of bits of PORT0 latched in RP0H, SYSCON, BUSCON0 registers.

Table 62. PORT0 latched configuration for the different reset events

X: Pin is sampled -: Pin is not sampled	PORT0															
	Clock options			Segm. addr. lines		Chip selects		WR config.	Bus type		Reserved	BSL	Reserved	Reserved	Adapt mode	Emu mode
Sample event	P0H.7	P0H.6	P0H.5	P0H.4	P0H.3	P0H.2	P0H.1	P0H.0	P0L.7	P0L.6	P0L.5	P0L.4	P0L.3	P0L.2	P0L.1	P0L.0
Software Reset	-	-	-	X	X	X	X	X	X	X	-	-	-	-	-	-
Watchdog Reset	-	-	-	X	X	X	X	X	X	X	-	-	-	-	-	-
Synchronous Short Hardware Reset	-	-	-	X	X	X	X	X	X	X	X	X	X	X	X	X
Synchronous Long Hardware Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Asynchronous Hardware Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Asynchronous Power-On Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Warning: **During power-off phase, it is important that the external hardware maintains a stable ground level on RSTIN pin, without any glitch, in order to avoid spurious exiting from reset status with unstable power supply.**

20.3.2 Exiting stand-by mode

After the system has entered the Stand-by Mode, the procedure to exit this mode consists of a standard Power-on sequence, with the only difference that the RAM is already powered through V_{18SB} internal reference (derived from V_{STBY} pin external voltage).

It is recommended to held the device under RESET ($\overline{RSTIN}$ pin forced low) until external V_{DD} voltage pin is stable. Even though, at the very beginning of the power-on phase, the device is maintained under reset by the internal low voltage detector circuit (implemented inside the main voltage regulator) till the internal V_{18} becomes higher than about 1.0V, **there is no warranty that the device stays under reset status if RSTIN is at high level during power ramp up. So, it is important the external hardware is able to guarantee a stable ground level on RSTIN along the power-on phase, without any temporary glitch.**

The external hardware shall be responsible to drive low the $\overline{RSTIN}$ pin until the V_{DD} is stable, even though the internal LVD is active.

Once the internal Reset signal goes low, the RAM (still frozen) power supply is switched to the main V_{18}.

At this time, everything becomes stable, and the execution of the initialization routines can start: XRAM2EN bit can be set, enabling the RAM.

20.3.3 Real-time clock and stand-by mode

When Stand-by mode is entered (turning off the main supply V_{DD}), the Real-Time Clock counting can be maintained running in case the on-chip 32 kHz oscillator is used to provide the reference to the counter. This is not possible if the main oscillator is used as reference for the counter: Being the main oscillator powered by V_{DD}, once this is switched off, the oscillator is stopped.

Table 67. Special function registers ordered by address (continued)

Name	Physical address	8-bit address	Description	Reset value
P1H **b**	FF06h	83h	Port1 high register (upper half of PORT1)	- - 00h
P1L **b**	FF04h	82h	Port1 low register (lower half of PORT1)	- - 00h
P2 **b**	FFC0h	E0h	Port 2 register	0000h
P3 **b**	FFC4h	E2h	Port 3 register	0000h
P4 **b**	FFC8h	E4h	Port 4 register (8-bit)	- - 00h
P5 **b**	FFA2h	D1h	Port 5 register (read-only)	XXXXh
P5DIDIS **b**	FFA4h	D2h	Port 5 digital disable register	0000h
P6 **b**	FFCCh	E6h	Port 6 register (8-bit)	- - 00h
P7 **b**	FFD0h	E8h	Port 7 register (8-bit)	- - 00h
P8 **b**	FFD4h	EAh	Port 8 register (8-bit)	- - 00h
PECC0	FEC0h	60h	PEC channel 0 control register	0000h
PECC1	FEC2h	61h	PEC channel 1 control register	0000h
PECC2	FEC4h	62h	PEC channel 2 control register	0000h
PECC3	FEC6h	63h	PEC channel 3 control register	0000h
PECC4	FEC8h	64h	PEC channel 4 control register	0000h
PECC5	FECAh	65h	PEC channel 5 control register	0000h
PECC6	FECCh	66h	PEC channel 6 control register	0000h
PECC7	FECEh	67h	PEC channel 7 control register	0000h
PICON **b**	F1C4h**E**	E2h	Port input threshold control register	- - 00h
PP0	F038h**E**	1Ch	PWM module period register 0	0000h
PP1	F03Ah**E**	1Dh	PWM module period register 1	0000h
PP2	F03Ch**E**	1Eh	PWM module period register 2	0000h
PP3	F03Eh**E**	1Fh	PWM module period register 3	0000h
PSW **b**	FF10h	88h	CPU program status word	0000h
PT0	F030h**E**	18h	PWM module up/down counter 0	0000h
PT1	F032h**E**	19h	PWM module up/down counter 1	0000h
PT2	F034h**E**	1Ah	PWM module up/down counter 2	0000h
PT3	F036h**E**	1Bh	PWM module up/down counter 3	0000h
PW0	FE30h	18h	PWM module pulse width register 0	0000h
PW1	FE32h	19h	PWM module pulse width register 1	0000h
PW2	FE34h	1Ah	PWM module pulse width register 2	0000h
PW3	FE36h	1Bh	PWM module pulse width register 3	0000h
PWMCON0 b	FF30h	98h	PWM module control register 0	0000h
PWMCON1 b	FF32h	99h	PWM module control register 1	0000h

Table 68. Special function registers ordered by address (continued)

Name	Physical address	8-bit address	Description	Reset value
SSCEIC **b**	FF76h	BBh	SSC error interrupt control register	- - 00h
CC0IC **b**	FF78h	BCh	CAPCOM register 0 interrupt control register	- - 00h
CC1IC **b**	FF7Ah	BDh	CAPCOM register 1 interrupt control register	- - 00h
CC2IC **b**	FF7Ch	BEh	CAPCOM register 2 interrupt control register	- - 00h
CC3IC **b**	FF7Eh	BFh	CAPCOM register 3 interrupt control register	- - 00h
CC4IC **b**	FF80h	C0h	CAPCOM register 4 interrupt control register	- - 00h
CC5IC **b**	FF82h	C1h	CAPCOM register 5 interrupt control register	- - 00h
CC6IC **b**	FF84h	C2h	CAPCOM register 6 interrupt control register	- - 00h
CC7IC **b**	FF86h	C3h	CAPCOM register 7 interrupt control register	- - 00h
CC8IC **b**	FF88h	C4h	CAPCOM register 8 interrupt control register	- - 00h
CC9IC **b**	FF8Ah	C5h	CAPCOM register 9 interrupt control register	- - 00h
CC10IC **b**	FF8Ch	C6h	CAPCOM register 10 interrupt control register	- - 00h
CC11IC **b**	FF8Eh	C7h	CAPCOM register 11 interrupt control register	- - 00h
CC12IC **b**	FF90h	C8h	CAPCOM register 12 interrupt control register	- - 00h
CC13IC **b**	FF92h	C9h	CAPCOM register 13 interrupt control register	- - 00h
CC14IC **b**	FF94h	CAh	CAPCOM register 14 interrupt control register	- - 00h
CC15IC **b**	FF96h	CBh	CAPCOM register 15 interrupt control register	- - 00h
ADCIC **b**	FF98h	CCh	A/D converter end of conversion interrupt control register	- - 00h
ADEIC **b**	FF9Ah	CDh	A/D converter overrun error interrupt control register	- - 00h
T0IC **b**	FF9Ch	CEh	CAPCOM timer 0 interrupt control register	- - 00h
T1IC **b**	FF9Eh	CFh	CAPCOM timer 1 interrupt control register	- - 00h
ADCON **b**	FFA0h	D0h	A/D converter control register	0000h
P5 **b**	FFA2h	D1h	Port 5 register (read-only)	XXXXh
P5DIDIS **b**	FFA4h	D2h	Port 5 digital disable register	0000h
TFR **b**	FFACh	D6h	Trap flag register	0000h
WDTCON **b**	FFAEh	D7h	Watchdog timer control register	00xxh
S0CON **b**	FFB0h	D8h	Serial channel 0 control register	0000h
SSCCON **b**	FFB2h	D9h	SSC control register	0000h
P2 **b**	FFC0h	E0h	Port 2 register	0000h
DP2 **b**	FFC2h	E1h	Port 2 direction control register	0000h
P3 **b**	FFC4h	E2h	Port 3 register	0000h
DP3 **b**	FFC6h	E3h	Port 3 direction control register	0000h

22.10 Identification registers

The ST10F276E has four Identification registers, mapped in ESFR space. These registers contain:

- the manufacturer identifier
- the chip identifier with revision number
- the internal Flash and size identifier
- the programming voltage description

IDMANUF (F07Eh / 3Fh) ESFR Reset value:0403h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MANUF											0	0	0	1	1
R															

Table 73. MANUF description

Bit	Function
MANUF	Manufacturer identifier 020h: STMicroelectronics manufacturer (JTAG worldwide normalization)

IDCHIP (F07Ch / 3Eh) ESFR Reset value:114xh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDCHIP												REVID			
R												R			

Table 74. IDCHIP description

Bit	Function
IDCHIP	Device identifier 114h: ST10F276E Identifier (276)
REVID	Device revision identifier Xh: According to revision number

IDMEM (F07Ah / 3Dh) ESFR Reset value:30D0h

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEMTYP				MEMSIZE											
R				R											

XPEREMU (EB7Eh) XBUS Reset value xxxxh:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	XMISC EN	XI2C EN	XSSC EN	XASC EN	XPWM EN	XFLAS HEN	XRTC EN	XRAM2 EN	XRAM1 EN	CAN2 EN	CAN1 EN
-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Note: The bit meaning is exactly the same as in XPERCON.

22.13 Emulation dedicated registers

Four additional registers are implemented for emulation purposes only. Similarly to XPEREMU, they are write-only registers.

XEMU0 (EB76h) XBUS Reset value: xxxxh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XEMU0(15:0)															
W															

XEMU1 (EB78h) XBUS Reset value: xxxxh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XEMU1(15:0)															
W															

XEMU2 (EB7Ah) XBUS Reset value: xxxxh:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XEMU2(15:0)															
W															

XEMU3 (EB7Ch) XBUS Reset value: xxxxh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XEMU3(15:0)															
W															

23.8.17 External memory bus timing

In the next sections the external memory bus timings are described. The given values are computed for a maximum CPU clock of 40 MHz.

It is evident that when higher CPU clock frequency is used (up to 64 MHz), some numbers in the timing formulas become zero or negative, which in most cases is not acceptable or meaningful. In these cases, the speed of the bus settings t_A, t_C and t_F must be correctly adjusted.

Note: All external memory bus timings and SSC timings presented in the following tables are given by design characterization and not fully tested in production.

23.8.18 Multiplexed bus

V_{DD} = 5V ±10%, V_{SS} = 0V, T_A = -40 to +125°C, C_L = 50pF,
ALE cycle time = 6 TCL + $2t_A$ + t_C + t_F (75ns at 40 MHz CPU clock without wait states).

Table 104. Multiplexed bus

Symbol		Parameter	f_{CPU} = 40 MHz TCL = 12.5ns		Variable CPU clock 1/2 TCL = 1 to 64 MHz		Unit
			Min.	Max.	Min.	Max.	
t_5	CC	ALE high time	4 + t_A	-	TCL - 8.5 + t_A	-	ns
t_6	CC	Address setup to ALE	1.5 + t_A		TCL - 11 + t_A		ns
t_7	CC	Address hold after ALE	4 + t_A		TCL - 8.5 + t_A		ns
t_8	CC	ALE falling edge to $\overline{RD}$, $\overline{WR}$ (with RW-delay)	4 + t_A		TCL - 8.5 + t_A		ns
t_9	CC	ALE falling edge to $\overline{RD}$, $\overline{WR}$ (no RW-delay)	- 8.5 + t_A		- 8.5 + t_A		ns
t_{10}	CC	Address float after $\overline{RD}$, $\overline{WR}$ (with RW-delay)(1)	-	6	-	6	ns
t_{11}	CC	Address float after $\overline{RD}$, $\overline{WR}$ (no RW-delay)(1)		18.5		TCL + 6	ns
t_{12}	CC	$\overline{RD}$, $\overline{WR}$ low time (with RW-delay)	15.5 + t_C	-	2TCL - 9.5 + t_C	-	ns
t_{13}	CC	$\overline{RD}$, $\overline{WR}$ low time (no RW-delay)	28 + t_C		3TCL - 9.5 + t_C		ns
t_{14}	SR	$\overline{RD}$ to valid data in (with RW-delay)	-	6 + t_C	-	2TCL - 19 + t_C	ns
t_{15}	SR	$\overline{RD}$ to valid data in (no RW-delay)		18.5 + t_C		3TCL - 19 + t_C	ns
t_{16}	SR	ALE low to valid data in		17.5 + t_A + t_C		3TCL - 20 + t_A + t_C	ns
t_{17}	SR	Address/Unlatched $\overline{CS}$ to valid data in		20 + $2t_A$ + t_C		4TCL - 30 + $2t_A$ + t_C	ns
t_{18}	SR	Data hold after $\overline{RD}$ rising edge	0	-	0	-	ns
t_{19}	SR	Data float after $\overline{RD}$	-	16.5 + t_F	-	2TCL - 8.5 + t_F	ns

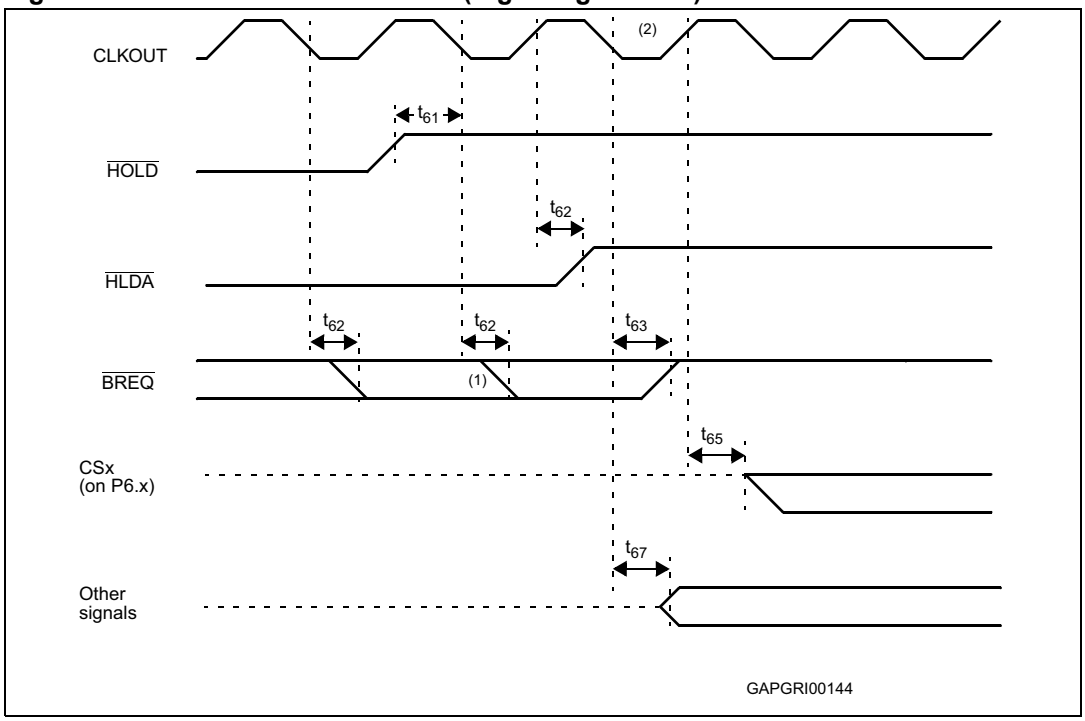
Table 104. Multiplexed bus (continued)

Symbol		Parameter	f_{CPU} = 40 MHz TCL = 12.5ns		Variable CPU clock 1/2 TCL = 1 to 64 MHz		Unit
			Min.	Max.	Min.	Max.	
t_{22}	CC	Data valid to $\overline{WR}$	10 + t_C	-	2TCL - 15 + t_C	-	ns
t_{23}	CC	Data hold after $\overline{WR}$	4 + t_F		2TCL - 8.5 + t_F		ns
t_{25}	CC	ALE rising edge after $\overline{RD}$, $\overline{WR}$	15 + t_F		2TCL - 10 + t_F		ns
t_{27}	CC	Address/Unlatched $\overline{CS}$ hold after $\overline{RD}$, $\overline{WR}$	10 + t_F		2TCL - 15 + t_F		ns
t_{38}	CC	ALE falling edge to $\overline{\text{Latched CS}}$	- 4 - t_A	10 - t_A	- 4 - t_A	10 - t_A	ns
t_{39}	SR	$\overline{\text{Latched CS}}$ low to valid data In	-	16.5 + t_C+ 2t_A	-	3TCL - 21 + t_C + 2t_A	ns
t_{40}	CC	$\overline{\text{Latched CS}}$ hold after $\overline{RD}$, $\overline{WR}$	27 + t_F	-	3TCL - 10.5 + t_F	-	ns
t_{42}	CC	ALE fall. edge to $\overline{RdCS}$, $\overline{WrCS}$ (with RW delay)	7 + t_A		TCL - 5.5 + t_A		ns
t_{43}	CC	ALE fall. edge to $\overline{RdCS}$, $\overline{WrCS}$ (no RW delay)	- 5.5 + t_A		- 5.5 + t_A		ns
t_{44}	CC	Address float after $\overline{RdCS}$, $\overline{WrCS}$ (with RW delay)(1)	-	1.5	-	1.5	ns
t_{45}	CC	Address float after $\overline{RdCS}$, $\overline{WrCS}$ (no RW delay)		14		TCL + 1.5	ns
t_{46}	SR	$\overline{RdCS}$ to valid data In (with RW delay)		4 + t_C		2TCL - 21 + t_C	ns
t_{47}	SR	$\overline{RdCS}$ to valid data In (no RW delay)		16.5 + t_C		3TCL - 21 + t_C	ns
t_{48}	CC	$\overline{RdCS}$, $\overline{WrCS}$ low time (with RW delay)	15.5 + t_C	-	2TCL - 9.5 + t_C	-	ns
t_{49}	CC	$\overline{RdCS}$, $\overline{WrCS}$ low time (no RW delay)	28 + t_C		3TCL - 9.5 + t_C		ns
t_{50}	CC	Data valid to $\overline{WrCS}$	10 + t_C		2TCL - 15 + t_C		ns
t_{51}	SR	Data hold after $\overline{RdCS}$	0		0		ns
t_{52}	SR	Data float after $\overline{RdCS}$(1)	-	16.5 + t_F	-	2TCL - 8.5 + t_F	ns
t_{54}	CC	Address hold after $\overline{RdCS}$, $\overline{WrCS}$	6 + t_F	-	2TCL - 19 + t_F	-	ns
t_{56}	CC	Data hold after $\overline{WrCS}$					ns

1. Partially tested, guaranteed by design characterization.

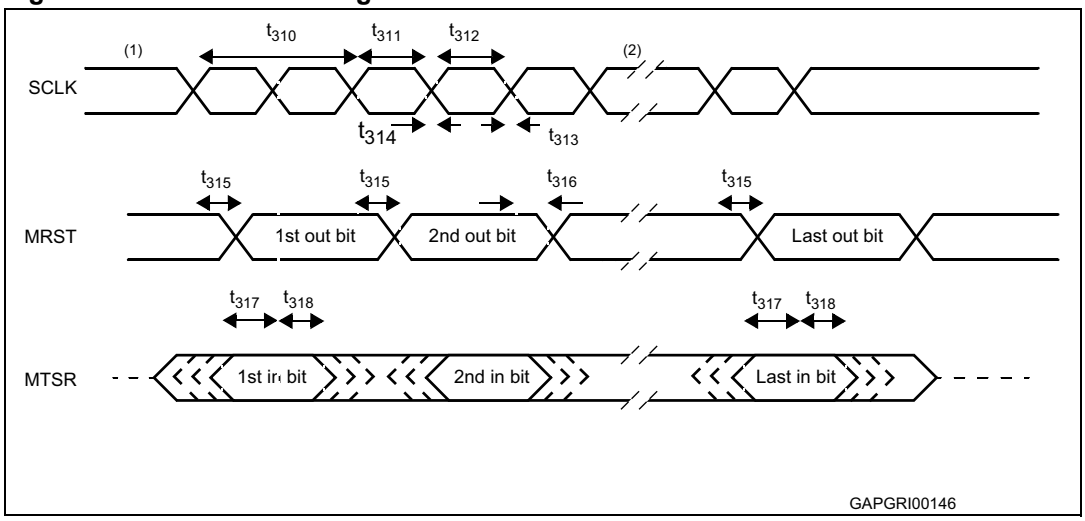
Figures *57* to *60* present the different configurations of external memory cycle for a multiplexed bus.

Figure 67. External bus arbitration (regaining the bus)



1. This is the last chance for $\overline{\text{BREQ}}$ to trigger the indicated regain-sequence. Even if $\overline{\text{BREQ}}$ is activated earlier, the regain-sequence is initiated by $\overline{\text{HOLD}}$ going high. Please note that $\overline{\text{HOLD}}$ may also be deactivated without the ST10F276E requesting the bus.
2. The next ST10F276E driven bus cycle may start here.

Figure 69. SSC slave timing



1. The phase and polarity of shift and latch edge of SCLK is programmable. This figure uses the leading clock edge as shift edge (drawn in bold), with latch on trailing edge (SSCPH = 0b), idle clock line is low, leading clock edge is low-to-high transition (SSCPO = 0b).
2. The bit timing is repeated for all bits to be transmitted or received.