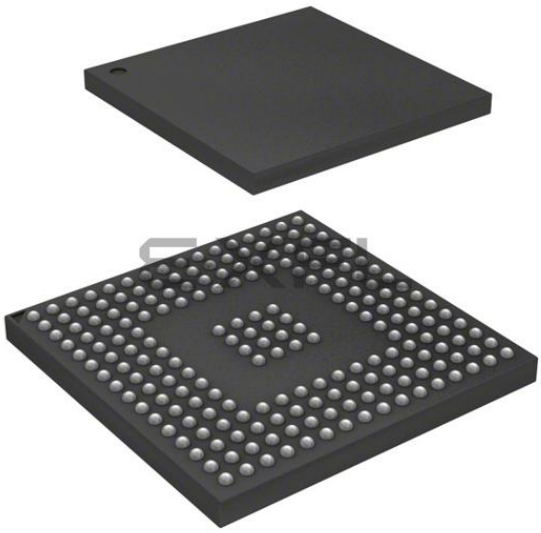


Welcome to E-XFL.COM

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	Active
Core Processor	ST10
Core Size	16-Bit
Speed	64MHz
Connectivity	ASC, CANbus, EBI/EMI, I²C, SSC, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	143
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 32x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	208-BGA
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st10f296tr

17.1 CAN module memory mapping . . . 192
17.1.1 CAN1 . . . 192
17.1.2 CAN2 . . . 192
17.2 Configuration support . . . 193
17.3 Clock prescaling . . . 193
17.4 CAN bus configurations . . . 194
17.4.1 Single CAN bus . . . 194
17.4.2 Multiple CAN bus . . . 195
17.4.3 Parallel mode . . . 195
17.5 System clock tolerance range . . . 196
17.6 Configuration of the CAN controller . . . 199
17.7 Calculation of the bit timing parameters . . . 200
17.7.1 Example of bit timing at high baud rate . . . 201
17.7.2 Example of bit timing at low baud rate . . . 202

18 Real-time clock (RTC) . . . 203
18.1 RTC registers . . . 205
18.2 Programming the RTC . . . 209

19 Watchdog timer . . . 211

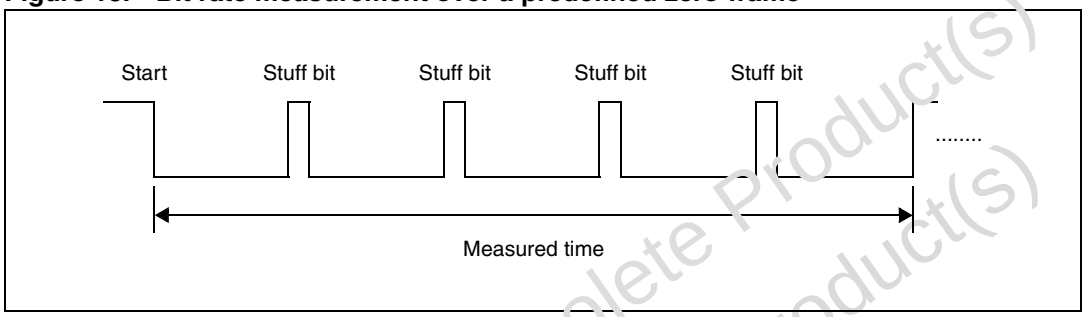
20 System reset . . . 214
20.1 Input filter . . . 214
20.2 Asynchronous reset . . . 215
20.2.1 Power-on reset . . . 215
20.2.2 Hardware reset . . . 218
20.2.3 Exit from asynchronous reset state . . . 219
20.3 Synchronous reset (warm reset) . . . 220
20.3.1 Short and long synchronous reset . . . 221
20.3.2 Exit from synchronous reset state . . . 222
20.3.3 Synchronous reset and the RPD pin . . . 222
20.4 Software reset . . . 226
20.5 Watchdog timer reset . . . 227
20.6 Bidirectional reset . . . 229
20.6.1 WDTCON flags . . . 230
20.7 Reset circuitry . . . 233

6.4.5 Choosing the baud rate for the BSL via CAN

The bootstrap via CAN acts in the same way as the UART bootstrap mode. When the ST10F296E is started in BSL mode, it polls the RxD0 and CAN1_RxD lines. When polling a low level on one of these lines, a timer is launched that is stopped when the line goes back to high level.

For CAN communication, the algorithm is made to receive a zero frame, where the standard identifier is 0x0 and DLC is 0. This frame produces the following levels on the network: 5D, 1R, 5D, 1R, 5D, 1R, 5D, 1R, 5D, 1R, 4D, 1R, 1D, 11R. The algorithm lets the timer run until detection of the 5th recessive bit. In this way, the bit timing is calculated over 29 bit time durations. This minimizes the error introduced by the polling.

Figure 13. Bit rate measurement over a predefined zero-frame



Error induced by the polling

The code used for polling is as follows:

```
WaitCom:
    JNB   P4.5,CAN_Boot          ; if SOF detected on CAN, then go to CAN
                                 ; loader
    JB    P3.11,WaitCom          ; Wait for start bit at RxD0
    BSET  T6R                    ; Start Timer T6
....
CAN_Boot:
    BSET  PWMCON0.0              ; Start PWM Timer0
                                 ; (resolution is 1 CPU clk cycle)
    JMPR  cc_UC,WaitRecessiveBit
WaitDominantBit:
    JB    P4.5,WaitDominantBit   ; wait for end of stuff bit
WaitRecessiveBit:
    JNB   P4.5,WaitRecessiveBit  ; wait for 1st dominant bit = Stuff bit
    CMPI1 R1,#5                  ; Test if 5th stuff bit detected
    JMPR  cc_NE,WaitDominantBit  ; No, go back to count more
    BCLR  PWMCON.0               ; Stop timer
                                 ; here the 5th stuff bit is detected:
                                 ; PT0 = 29 Bit_Time (25D and 4R)
```

The maximum error at detection of communication on the CAN pin is: (1 not taken + 1 taken jumps) + 1 taken jump + 1 bit set: (6) + 6 CPU clock cycles

The error at detection of the 5th recessive bit is: (1 taken jump) + 1 not taken jump + 1 compare + 1 bit clear: (4) + 6 CPU cycles

In the worst case scenario, the induced error is 6 CPU clock cycles. So, polling could induce an error of 6 timer ticks.

6.7 Selective boot mode

Selective boot mode is a sub-case of alternate boot mode.

The following additional check is made when no signature of the alternate boot mode signature check is correct:

Address 00'1FFCh is read again.

- If a value 0000h or FFFFh is obtained, a jump is performed to the standard bootstrap loader.
- If the value obtained is not 0000h or FFFFh:
  - High byte bits are disregarded
  - Low byte bits select which communication channel is enabled (see *Table 44*).

Table 44. Selective boot mode configurations

Bit	Function
0	UART selection 0: UART not watched for a start condition 1: UART is watched for a start condition
1	CAN1 selection 0: CAN1 not watched for a start condition 1: CAN1 is watched for a start condition
2-7	Reserved Must be programmed to 0 for upward compatibility

- 0xXX03 configures the selective bootstrap loader to poll for RxD0 and CAN1_RxD.
- 0xXX01 configures the selective bootstrap loader to poll RxD0 only (no bootloading via CAN).
- 0xXX02 configures the selective bootstrap loader to poll CAN1_RxD only (no bootloading via UART).
- other values will let the ST10F296E executing an endless loop into the Test-Flash.

7 Central processing unit (CPU)

The CPU includes a four-stage instruction pipeline, a 16-bit arithmetic and logic unit (ALU) and dedicated SFRs. Additional hardware has been added for a separate multiply and divide unit, a bit-mask generator and a barrel shifter.

Most instructions of the ST10F296E can be executed in one instruction cycle which requires 31.25 ns at 25 MHz CPU clock. For example, shift and rotate instructions are processed in one instruction cycle independent of the number of bits to be shifted.

Multiple-cycle instructions have been optimized. Branches are carried out in two cycles, 16 x 16-bit multiplication in five cycles and a 32/16-bit division in 10 cycles.

The jump cache reduces the execution time of repeatedly performed jumps in a loop, from two cycles to one cycle.

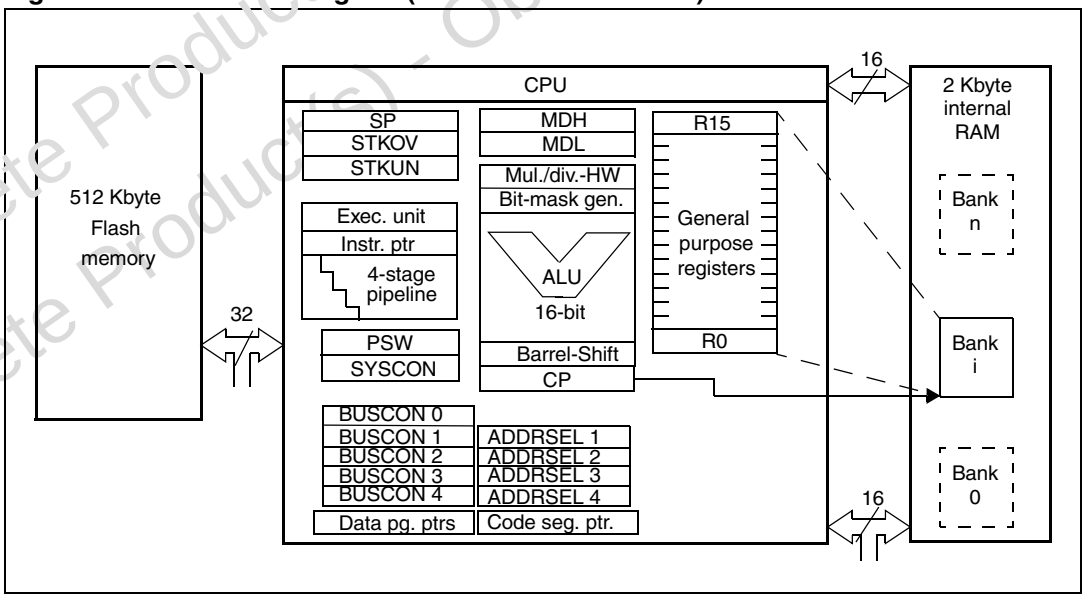
The CPU uses a bank of 16 word registers to run the current context. This bank of general purpose registers (GPR) is physically stored within the on-chip internal RAM (IRAM) area. A context pointer (CP) register determines the base address of the active register bank to be accessed by the CPU.

The number of register banks is restricted by the available internal RAM space. For easy parameter passing, a register bank may overlap others.

A system stack of up to 1024 bytes is provided as a storage for temporary data. The system stack is allocated in the on-chip RAM area, and it is accessed by the CPU via the stack pointer (SP) register.

Two separate SFRs, STKOV and STKUN, are implicitly compared against the stack pointer value upon each stack access for the detection of a stack overflow or underflow.

Figure 16. CPU block diagram (MAC unit not included)



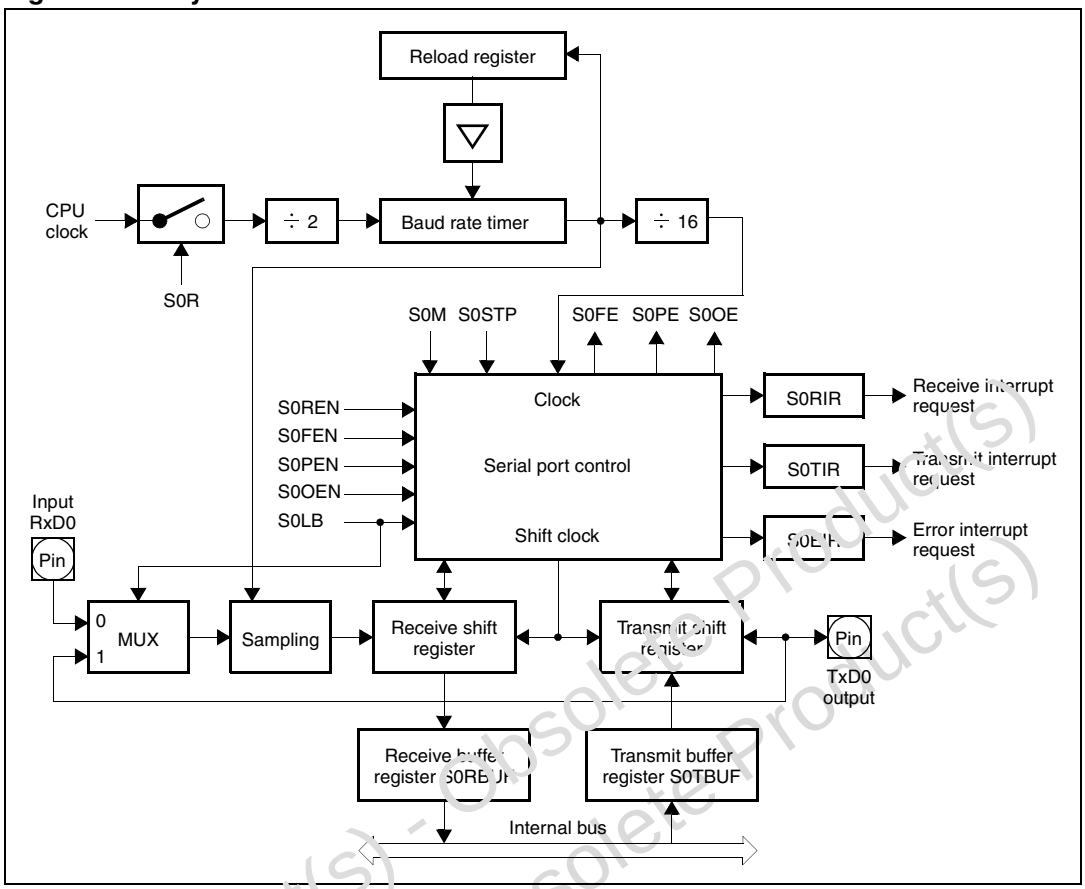
Timer output

The trigger output, XADCINJ, is generated when the current value of the timer (XTCVR) matches the end value stored in the XTEVR register and when the output enable bit is set (XTCR.TOE = 1). If the output enable bit is reset, no event is generated regardless of the timer status (the XADINJ pin is kept at high impedance state).

The XADCINJ output trigger event is a positive pulse of 12 CPU clock cycles width (187 ns @64 MHz). To generate an ADC channel injection it has to be externally connected to the input P7.7/CC31 (CAPCOM2 capture/compare).

The ADC exclusively converts Port 5 or XPort 10 inputs. If one ‘y’ channel has to be used continuously in injection mode, it must be externally connected by hardware to Port5.y and XPort10.y inputs.

Figure 64. Asynchronous mode of serial channel ASC0



15.1.2 Asynchronous mode baud rates

For asynchronous operation, the baud rate generator provides a clock with 16 times the rate of the established baud rate. Every received bit is sampled at the 7th, 8th and 9th cycle of this clock. The baud rate for asynchronous operations of serial channel ASC0 and the required reload value for a given baud rate can be determined by the following formulae:

$$B_{Async} = f_{CPU} / 16 \times [2 + (S0BRS)] \times [(S0BRL) + 1]$$

$$S0BRL = (f_{CPU} / 16 \times [2 + (S0BRS)] \times B_{Async}) - 1$$

(S0BRL) represents the content of the reload register, taken as an unsigned 13-bit integer. (S0BRS) represents the value of the S0BRS bit (0 or 1), taken as an integer.

Using the above equations, the maximum baud rate can be calculated for any given clock speed. Baud rate versus the reload register value (for both S0BRS = 0 and S0BRS = 1) is described in *Table 118* and *Table 119* for a CPU clock frequency equal to 40 MHz and 64 MHz respectively.

Warning: **It is recommended to provide the external hardware with a current limitation circuitry. This is necessary to avoid permanent damage to the device during the power-on transient, when the capacitance on V_{18} pin is charged. For the on-chip voltage regulator functionality, 10 nF is sufficient. A maximum of 100 nF on the V_{18} pin should not generate problems of overcurrent (a higher value is allowed if the current is limited by the external hardware). External current limitation is also recommended to avoid risks of damage in case of temporary shorts between V_{18} and ground. The internal 1.8 V drivers are sized to drive currents of several tens of ampere, so, the current must be limited by the external hardware. The current limit is imposed by power dissipation considerations (refer to *Section 24: Electrical characteristics*).**

Figure 75 and *Figure 76* show the asynchronous power-on timing diagrams with boot from internal or external memory respectively. The reset phase extension that is introduced by the embedded Flash module, is highlighted.

Note: Never power the device without keeping the $\overline{RSTIN}$ pin grounded as the device could enter unpredictable states which could permanently damage it.

20.9 Reset summary

Table 132 summarizes the different reset events.

Table 132. Reset events summary

Event	RPD	$\overline{EA}$	Bidirectional	synchronous/ asynchronous	$\overline{RSTIN}$		WDTCON flags				
					Min	Max	PONR	LHWR	SHWR	SWR	WDTR
Power-on reset	0	0	N	Asynch.	1 ms (VREG) 1.2 ms (reson. + PLL) 10.2 ms (crystal + PLL)	-	1	1	1	1	0
	0	1	N	Asynch.	1 ms (VREG)	-	1	1	1	1	0
	1	x	x	Forbidden							
	x	x	Y	Not applicable							
Hardware reset (asynchronous)	0	0	N	Asynch.	500 ns	-	0	1	1	1	0
	0	1	N	Asynch.	500 ns	-	0	1	1	1	0
	0	0	Y	Asynch.	500 ns	-	0	1	1	1	0
	0	1	Y	Asynch.	500 ns	-	0	1	1	1	0
Short hardware reset (synchronous)(1)	1	0	N	Synch.	Max (4 TCL, 500 ns)	1032 + 12 TCL + max (4 TCL, 500 ns)	0	0	1	1	0
	1	1	N	Synch.	Max (4 TCL, 500 ns)	1032 + 12 TCL + max (4 TCL, 500ns)	0	0	1	1	0
	1	0	Y	Synch.	Max (4 TCL, 500 ns)	1032 + 12 TCL + max (4 TCL, 500 ns)	0	0	1	1	0
					Activated by internal logic for 1024 TCL						
	1	1	Y	Synch.	Max (4 TCL, 500 ns)	1032 + 12 TCL + max (4 TCL, 500 ns)	0	0	1	1	0
					Activated by internal logic for 1024 TCL						
Long hardware reset (synchronous)	1	0	N	Synch.	1032 + 12 TCL + max (4 TCL, 500 ns)	-	0	1	1	1	0
	1	1	N	Synch.	1032 + 12 TCL + max(4 TCL, 500 ns)	-	0	1	1	1	0
	1	0	Y	Synch.	1032 + 12 TCL + max (4 TCL, 500 ns)	-	0	1	1	1	0
					Activated by internal logic only for 1024 TCL						
	1	1	Y	Synch.	1032 + 12 TCL + max (4 TCL, 500 ns)	-	0	1	1	1	0
					Activated by internal logic only for 1024 TCL						

22 Programmable output clock divider

A specific register mapped on the XBus allows the division factor on the CLKOUT signal (P3.15) to be chosen. This register, XCLKOUTDIV, is mapped on the XMiscellaneous memory address range.

XCLKOUTDICV register

XCLKOUTDIV (EB02h) XBus Reset value: --00h

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

Table 136. XCLKOUTDIV register description

Bit	Bit name	Function
7-0	DIV	Clock divider setting 00h: $F_{CLKOUT} = F_{CPU}/DIV+1$

The CPU clock is output on P3.15, by default, when the CLKOUT function is enabled (setting the CLKEN bit of the SYSCON register)

By setting the XMISCEN and XPEN bits of the XPERCON and SYSCON registers respectively, the clock prescaling factor can be programmed. In this way, a prescaled value of the CPU clock can be output on P3.15.

When the CLKOUT function is not enabled (clearing the CLKEN bit of the SYSCON on P3.15), P3.15 does not output a clock signal, even though the XCLKOUTDIV register is programmed.

Table 140. SFRs ordered by name (continued)

Name	Physical address	8-bit address	Description	Reset value
CC6IC **(b)**	FF84h	C2h	CAPCOM register 6 interrupt control register	--00h
CC7	FE8Eh	47h	CAPCOM register 7	0000h
CC7IC **(b)**	FF86h	C3h	CAPCOM register 7 interrupt control register	--00h
CC8	FE90h	48h	CAPCOM register 8	0000h
CC8IC **(b)**	FF88h	C4h	CAPCOM register 8 interrupt control register	--00h
CC9	FE92h	49h	CAPCOM register 9	0000h
CC9IC **(b)**	FF8Ah	C5h	CAPCOM register 9 interrupt control register	--00h
CC10	FE94h	4Ah	CAPCOM register 10	0000h
CC10IC **(b)**	FF8Ch	C6h	CAPCOM register 10 interrupt control register	--00h
CC11	FE96h	4Bh	CAPCOM register 11	0000h
CC11IC **(b)**	FF8Eh	C7h	CAPCOM register 11 interrupt control register	--00h
CC12	FE98h	4Ch	CAPCOM register 12	0000h
CC12IC **(b)**	FF90h	C8h	CAPCOM register 12 interrupt control register	--00h
CC13	FE9Ah	4Dh	CAPCOM register 13	0000h
CC13IC **(b)**	FF92h	C9h	CAPCOM register 13 interrupt control register	--00h
CC14	FE9Ch	4Eh	CAPCOM register 14	0000h
CC14IC **(b)**	FF94h	CAh	CAPCOM register 14 interrupt control register	--00h
CC15	FE9Eh	4Fh	CAPCOM register 15	0000h
CC15IC **(b)**	FF96h	CBh	CAPCOM register 15 interrupt control register	--00h
CC16	FE60h	30h	CAPCOM register 16	0000h
CC16IC **(b)**	F160h **(E)**	B0h	CAPCOM register 16 interrupt control register	--00h
CC17	FE62h	31h	CAPCOM register 17	0000h
CC17IC **(b)**	F162h **(E)**	B1h	CAPCOM register 17 interrupt control register	--00h
CC18	FE64h	32h	CAPCOM register 18	0000h
CC18IC **(b)**	F164h **(E)**	B2h	CAPCOM register 18 interrupt control register	--00h
CC19	FE66h	33h	CAPCOM register 19	0000h
CC19IC **(b)**	F166h **(E)**	B3h	CAPCOM register 19 interrupt control register	--00h
CC20	FE68h	34h	CAPCOM register 20	0000h
CC20IC **(b)**	F168h **(E)**	B4h	CAPCOM register 20 interrupt control register	--00h
CC21	FE6Ah	35h	CAPCOM register 21	0000h
CC21IC **(b)**	F16Ah **(E)**	B5h	CAPCOM register 21 interrupt control register	--00h
CC22	FE6Ch	36h	CAPCOM register 22	0000h
CC22IC **(b)**	F16Ch **(E)**	B6h	CAPCOM register 22 interrupt control register	--00h
CC23	FE6Eh	37h	CAPCOM register 23	0000h

Table 143. X registers ordered by address (continued)

Name	Physical address	Description	Reset value
CAN2TR2	EE82h	CAN2 transmission request 2	0000h
CAN2ND1	EE90h	CAN2 new data 1	0000h
CAN2ND2	EE92h	CAN2 new data 2	0000h
CAN2IP1	EEA0h	CAN2 interrupt pending 1	0000h
CAN2IP2	EEA2h	CAN2 interrupt pending 2	0000h
CAN2MV1	EEB0h	CAN2 message valid 1	0000h
CAN2MV2	EEB2h	CAN2 message valid 2	0000h
CAN1CR	EF00h	CAN1 CAN control register	0001h
CAN1SR	EF02h	CAN1 status register	0000h
CAN1EC	EF04h	CAN1 error counter	0000h
CAN1BTR	EF06h	CAN1 bit timing register	2301h
CAN1IR	EF08h	CAN1 interrupt register	0000h
CAN1TR	EF0Ah	CAN1 test register	00x0h
CAN1BRPER	EF0Ch	CAN1 BRP extension register	0000h
CAN1IF1CR	EF10h	CAN1 IF1 command request	0001h
CAN1IF1CM	EF12h	CAN1 IF1 command mask	0000h
CAN1IF1M1	EF14h	CAN1 IF1 mask 1	FFFFh
CAN1IF1M2	EF16h	CAN1 IF1 mask 2	FFFFh
CAN1IF1A1	EF18h	CAN1 IF1 arbitration 1	0000h
CAN1IF1A2	EF1Ah	CAN1 IF1 arbitration 2	0000h
CAN1IF1MC	EF1Ch	CAN1 IF1 message control	0000h
CAN1IF1DA1	EF1Eh	CAN1 IF1 data A 1	0000h
CAN1IF1DA2	EF20h	CAN1 IF1 data A 2	0000h
CAN1IF1DB1	EF22h	CAN1 IF1 data B 1	0000h
CAN1IF1DB2	EF24h	CAN1 IF1 data B 2	0000h
CAN1IF2CR	EF40h	CAN1 IF2 command request	0001h
CAN1IF2CM	EF42h	CAN1 IF2 command mask	0000h
CAN1IF2M1	EF44h	CAN1 IF2 mask 1	FFFFh
CAN1IF2M2	EF46h	CAN1 IF2 mask 2	FFFFh
CAN1IF2A1	EF48h	CAN1 IF2 arbitration 1	0000h
CAN1IF2A2	EF4Ah	CAN1 IF2 arbitration 2	0000h
CAN1IF2MC	EF4Ch	CAN1 IF2 message control	0000h
CAN1IF2DA1	EF4Eh	CAN1 IF2 data A 1	0000h
CAN1IF2DA2	EF50h	CAN1 IF2 data A 2	0000h

23.7 Flash registers ordered by name

Table 144 lists all Flash control registers which are implemented in the ST10F296E ordered by their name. As these registers are physically mapped on the XBus, they are not bit-addressable.

Table 144. Flash registers ordered by name

Name	Physical address	Description	Reset value
FARH	0x000E 0012	Flash address register high	0000h
FARL	0x000E 0010	Flash address register low	0000h
FCR0H	0x000E 0002	Flash control register 0 - high	0000h
FCR0L	0x000E 0000	Flash control register 0 - low	0000h
FCR1H	0x000E 0006	Flash control register 1 - high	0000h
FCR1L	0x000E 0004	Flash control register 1 - low	0000h
FDR0H	0x000E 000A	Flash data register 0 - high	FFFFh
FDR0L	0x000E 0008	Flash data register 0 - low	FFFFh
FDR1H	0x000E 000E	Flash data register 1 - high	FFFFh
FDR1L	0x000E 000C	Flash data register 1 - low	FFFFh
FER	0x000E 0014	Flash error register	0000h
FNVAPR0	0x000E DFB8	Flash non volatile access protection register 0	ACFFh
FNVAPR1H	0x000E DFBE	Flash non volatile access protection register 1 - high	FFFFh
FNVAPR1L	0x000E DFBC	Flash non volatile access protection register 1 - low	FFFFh
FNVWPIRH	0x000E DFB6	Flash non volatile protection I register high	FFFFh
FNVWPIRL	0x000E DFB4	Flash non volatile protection I register low	FFFFh
FNVWPXRH	0x000E DFB2	Flash non volatile protection X register high	FFFFh
FNVWPXRL	0x000E DFB0	Flash non volatile protection X register low	FFFFh
XFICR	0x000E E000	XFlash interface control register	000Fh

In emulation mode, all XPeripherals are enabled (all XPERCON bits are set). The access to the external memory and/or the XBus is controlled by the bondout chip.

Reserved bits of the XPERCON register must always be written to 0.

When the RTC is disabled (RTCEN = 0) the main clock oscillator is switched off if the ST10 enters power-down mode. When the RTC is enabled, the RTCOFF bit of the RTCCON register allows the power-down mode of the main clock oscillator to be chosen (eee *Section 18: Real-time clock (RTC) on page 203*).

Table 158 summarizes the address range mapping on segment 8 for programming the ROMEN and XPEN bits (of the SYSCON register) and the XRAM2EN and XFLASHEN bits (of the XPERCON register).

Table 158. Segment 8 address range mapping

ROMEN	XPEN	XRAM2EN	XFLASHEN	Segment 8
0	0	x(1)	x(1)	External memory
0	1	0	0	External memory
0	1	1	x(1)	Reserved
0	1	x(1)	1	Reserved
1	x(1)	x(1)	x(1)	IFlash (B1F1)

1. Don't care

23.10.1 XPEREMU register

The XPEREMU register is a write-only register that is mapped on the XBus memory space at address EB7Eh. It contrasts with the XPERCON register, a read/write ESFR register, which must be programmed to enable the single XBus modules separately.

Once the XPEN bit of the SYSCON register is set and at least one of the XPeripherals (except the memories) is activated, the XPEREMU register must be written with the same contents as the XPERCON register. This is to allow a correct emulation of the new set of features introduced on the XBus for the new ST10 generation. The following instructions must be added inside the initialization routine:

```
if (SYSCON.XPEN && (XPERCON & 0x07D3))
then { XPEREMU = XPERCON }
```

XPEREMU must be programmed after both the XPERCON and SYSCON registers in such a way that the final configuration for the XPeripherals is stored in the XPEREMU register and used for the emulation hardware setup.

XPEREMU (EB7Eh) XBus Reset value: xxxxh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	XPORTEN	XMISCEN	XI2CEN	XSSCEN	XASCEN	XPWMEN	XFLASHEN	XRTCEN	XRAM2EN	XRAM1EN	CAN2EN	CAN1EN
-	-	-	-	W	W	W	W	W	W	W	W	W	W	W	W

XPEREMU bit descriptition follows the XPERCON register (see *Table 5* and *Table 157*).

24.7.3 Analog reference pins

The accuracy of the ADC converter depends on the accuracy of its analog reference. A noise in the reference results in the same proportion of error in a conversion. A low pass filter on the ADC converter reference source (supplied through the V_{AREF} and V_{AGND} pins), is recommended to clean the signal thereby minimizing the noise. A simple capacitive bypassing may be sufficient in most cases. In the presence of high RF noise energy, inductors or ferrite beads may be necessary.

In the ST10F296E architecture, the V_{AREF} and V_{AGND} pins also represent the power supply of the analog circuitry of the ADC. An effective DC current is required from the reference voltage to the internal resistor string in the R-C DAC array and to the rest of the analog circuitry.

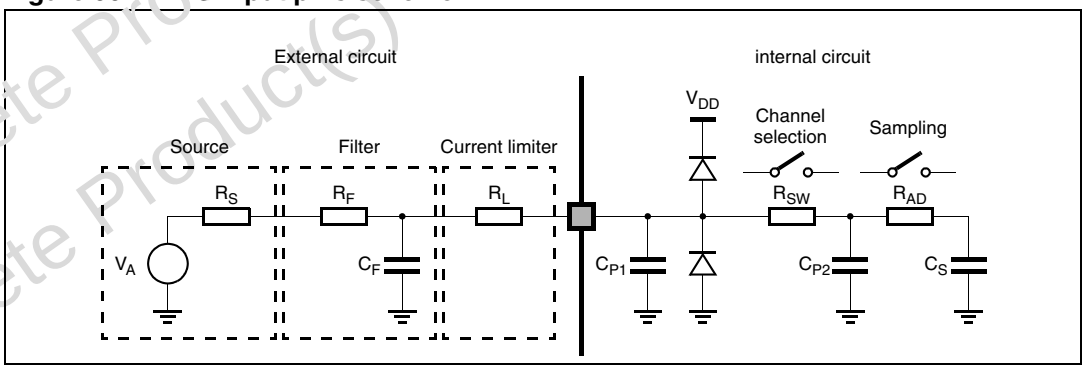
An external resistance on V_{AREF} could introduce error under certain conditions. For this reason, series resistance is not advisable. Any series devices in the filter network should be designed to minimize the DC resistance.

24.7.4 Analog input pins

To improve the accuracy of the ADC, analog input pins must have low AC impedance. Placing a capacitor with good high frequency characteristics at the input pin of the device can be effective. The capacitor should be as large as possible, ideally infinite. This capacitor contributes to attenuating the noise present on the input pin. Moreover, the source of the capacitor charges during the sampling phase, when the analog signal source is a high-impedance source.

A real filter is typically obtained by using a series resistance with a capacitor on the input pin (simple RC filter). RC filtering may be limited according to the value of the impedance source of the transducer or circuit supplying the analog signal to be measured. The filter at the input pins must be designed to account for the dynamic characteristics of the input signal (bandwidth).

Figure 99. ADC input pins scheme



1. Legend:
   R_S: Source impedance
   R_F: Filter resistance
   C_F: Filter capacitance
   R_L: Current limiter resistance
   R_{SW}: Channel selection switch impedance
   R_{AD}: Sampling switch impedance
   c_p: Pin capacitance (two contributions, CP1 and CP2)
   C_S: Sampling capacitance
   V_A: Source voltage

Input leakage and external circuit

The series resistor used to limit the current to a pin (see R_L in *Figure 99*), in combination with a large source of impedance, can lead to a degradation of the ADC accuracy when input leakage is present.

Data about maximum input leakage current at each pin is provided in *Section 24.5: DC characteristics*. Input leakage is greatest at high operating temperatures and generally decreases by one half a degree for each 10 °C decrease in temperature.

Considering that one count of a 10-bit ADC is about 5 mV (assuming V_{AREF} = 5 V), an input leakage of 100 nA acting though an R_L = 50 kΩ of external resistance, leads to an error of exactly one count (5 mV). If the resistance is 100 kΩ, the error is two counts (10 mV).

Additional leakage due to external clamping diodes must also be taken into account in computing the total leakage affecting the ADC measurements. Another contribution to the total leakage is represented by the charge sharing effects with the sampling capacitance. The sampling capacitance, C_S, is essentially a switched capacitance with a frequency equal to the conversion rate of a single channel (maximum when the fixed channel continuous conversion mode is selected). It can be seen as a resistive path to ground. For instance, assuming a conversion rate of 250 kHz and a C_S of 4 pF, a resistance of 1 MΩ is obtained ($R_{EQ} = 1/f_C C_S$, where f_C represents the conversion rate at the considered channel). To minimize the error induced by the voltage partitioning between this resistance (sampled voltage on C_S) and the sum of $R_S + R_F + R_L + R_{SW} + R_{AD}$, the external circuit must be designed to respect the following relation:

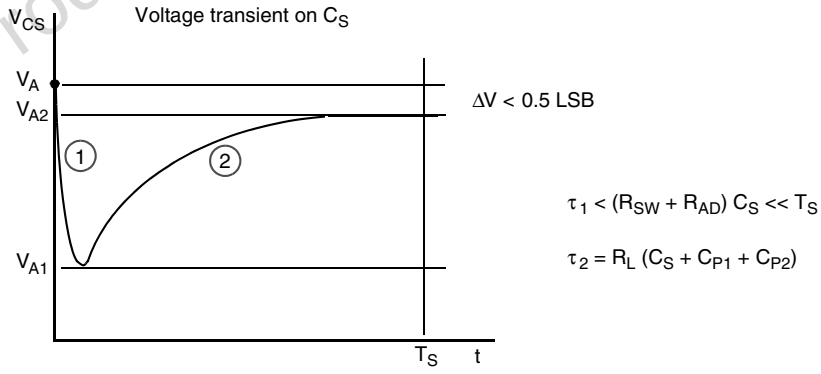
Equation 25

$$V_A \times (R_S + R_F + R_L + R_{SW} + R_{AD}) / R_{EQ} < (1/2)LSB$$

Equation 25 places constraints on the external network design, in particular on the resistive path.

A second aspect of the capacitance network must be considered. Assuming the three capacitances C_F, C_{P1} and C_{P2}, are initially charged at the source voltage V_A (see *Figure 99*), when the sampling phase is started (ADC switch closed), a charge-sharing phenomena begins (see *Figure 100*).

Figure 100. Charge sharing timing diagram during sampling phase



Two different transient periods can be distinguished in *Figure 100*. They are described below.

Figure 111. Multiplexed bus with/without R/ W delay, extended ALE, R/W $\overline{CS}$

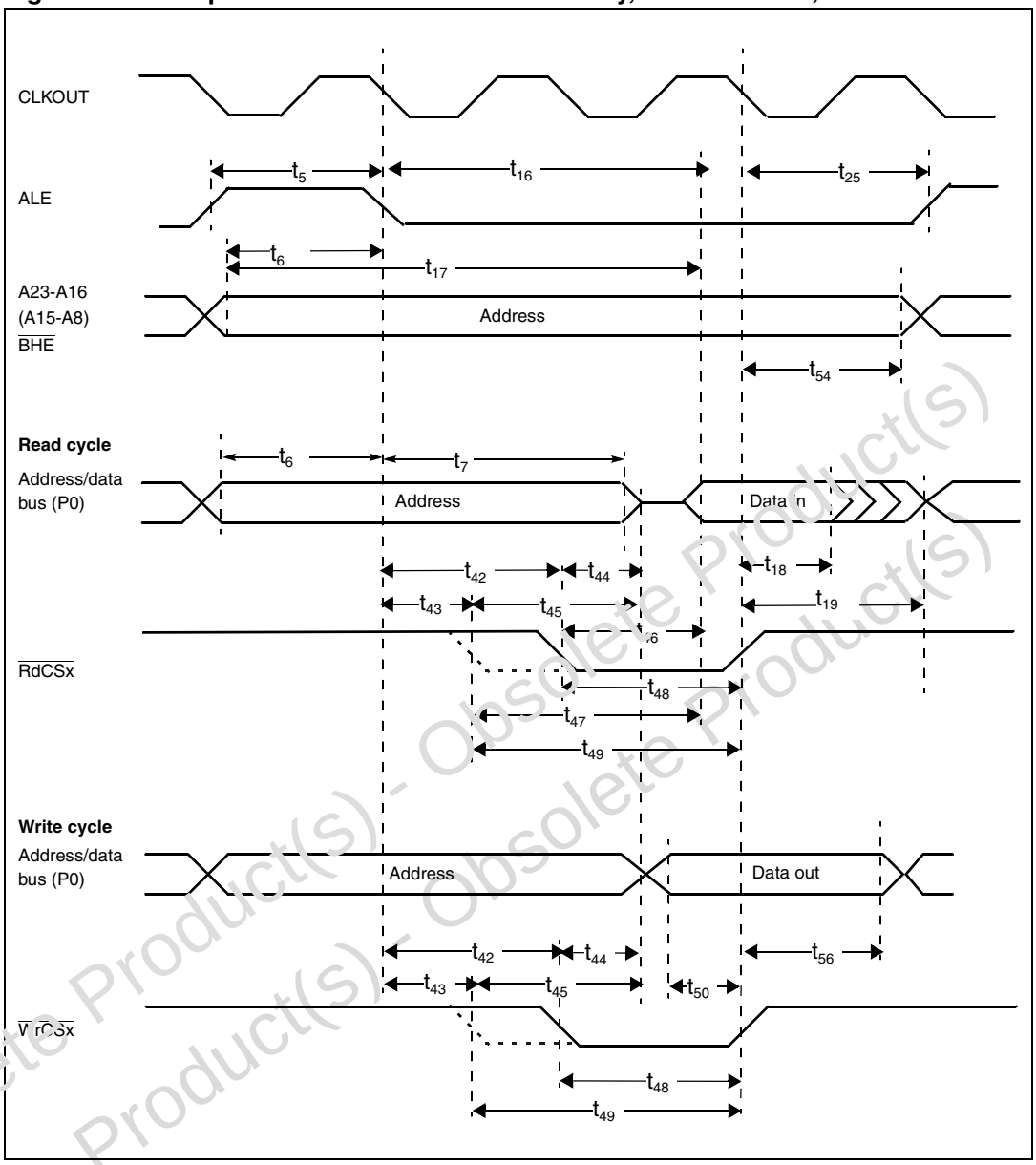
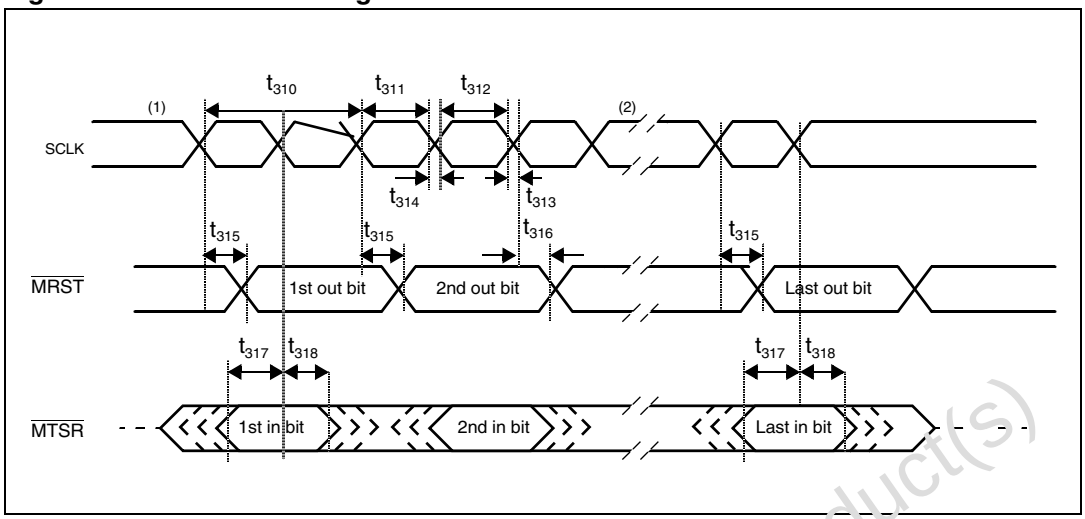


Figure 120. SSC slave timing



1. The phase and polarity of the shift and latch edges of SCLK are programmable. Figure 120 uses the leading clock edge as the shift edge with the latch on the trailing edge (SSCPH = 0b). The idle clock line is low and the leading clock edge is low-to-high transition (SSCPO = 0b).
2. The bit timing is repeated for all bits that have to be transmitted or received.

26 Ordering information

Table 182. Order codes

Order codes	Package	Packing	Temperature range (°C)	CPU frequency range (MHz)
ST10F296	PBGA208	Tray	-40 to 125	1 to 64
ST10F296TR		Tape and reel		