STMicroelectronics - ST92F150CV1QB Datasheet

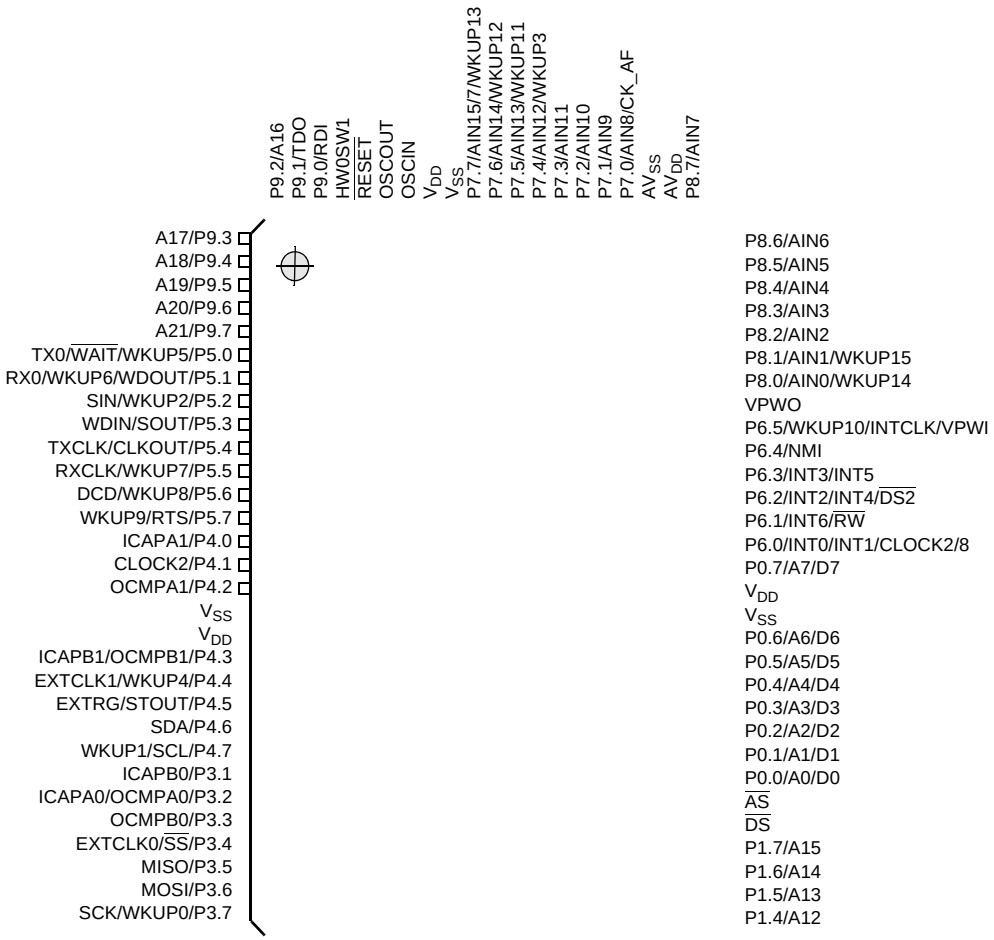
Welcome to E-XFL.COM

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	ST9
Core Size	8/16-Bit
Speed	24MHz
Connectivity	CANbus, EBI/EMI, I²C, LINbus, SCI, SPI
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	77
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st92f150cv1qb


P9.2/A16
P9.1/TDO
P9.0/RDI
HW0SW1
RESET
OSCOUT
OSCIN
VDD
VSS
P7.7/AIN15/7/WKUP13
P7.6/AIN14/WKUP12
P7.5/AIN13/WKUP11
P7.4/AIN12/WKUP3
P7.3/AIN11
P7.2/AIN10
P7.1/AIN9
P7.0/AIN8/CK_AF
AVSS
AVDD
P8.7/AIN7
A17/P9.3
A18/P9.4
A19/P9.5
A20/P9.6
A21/P9.7
TX0/WAIT/WKUP5/P5.0
RX0/WKUP6/WDOUT/P5.1
SIN/WKUP2/P5.2
WDIN/SOUT/P5.3
TXCLK/CLKOUT/P5.4
RXCLK/WKUP7/P5.5
DCD/WKUP8/P5.6
WKUP9/RTS/P5.7
ICAPA1/P4.0
CLOCK2/P4.1
OCMPA1/P4.2
VSS
VDD
ICAPB1/OCMPB1/P4.3
EXTCLK1/WKUP4/P4.4
EXTRG/STOUT/P4.5
SDA/P4.6
WKUP1/SCL/P4.7
ICAPB0/P3.1
ICAPA0/OCMPA0/P3.2
OCMPB0/P3.3
EXTCLK0/SS/P3.4
MISO/P3.5
MOSI/P3.6
SCK/WKUP0/P3.7
P8.6/AIN6
P8.5/AIN5
P8.4/AIN4
P8.3/AIN3
P8.2/AIN2
P8.1/AIN1/WKUP15
P8.0/AIN0/WKUP14
VPWO
P6.5/WKUP10/INTCLK/VPWI
P6.4/NMI
P6.3/INT3/INT5
P6.2/INT2/INT4/DS2
P6.1/INT6/RW
P6.0/INT0/INT1/CLOCK2/8
P0.7/A7/D7
VDD
VSS
P0.6/A6/D6
P0.5/A5/D5
P0.4/A4/D4
P0.3/A3/D3
P0.2/A2/D2
P0.1/A1/D1
P0.0/A0/D0
AS
DS
P1.7/A15
P1.6/A14
P1.5/A13
P1.4/A12

1: PLL Free Running Mode enabled

When this bit is set, even if the DX[2:0] bits are all set to 1, the PLL is not stopped but provides a slow frequency back-up clock, selectable by the CSU_CKSEL bit of the CLK_FLAG register (without needing to have the LOCK bit equal to ‘1’).

Bits 5:4 = : *PLL Multiplication Factor.*
Refer to *Table 37* for multiplier settings.

■ ■
■

Bits 2:0 = : *PLL output clock divider factor.* Refer to *Table 38* for divider settings.

.
● ■

1	0	14
0	0	10
1	1	8
0	1	6

●

0	0	0	PLL CLOCK/1
0	0	1	PLL CLOCK/2
0	1	0	PLL CLOCK/3
0	1	1	PLL CLOCK/4
1	0	0	PLL CLOCK/5
1	0	1	PLL CLOCK/6
1	1	0	PLL CLOCK/7
1	1	1	CLOCK2 (PLL OFF, Reset State)

●

WFI	High Impedance or next address (depending on the last memory operation performed on Port)	Next Address	Not Affected (clock outputs running)
HALT	High Impedance	Next Address	Not Affected (clock outputs stopped)
RESET	Alternate function push-pull (ROMless device)		Bidirectional Weak Pull-up (High impedance when disabled in hardware).

1. Depending on device.

■

This mode can be used for pulse width measurement. The Timer is clocked by INTCLK/4, and is started and stopped by means of the input pin and the ST_SP bit. When the input pin is high, the Timer counts. When it is low, counting stops. The maximum input pin frequency is equivalent to INTCLK/8.

■

The Timer (clocked internally by INTCLK/4) is started by the following sequence:

- setting the Start-Stop bit, followed by
- a High to Low transition on the input pin.

■

	●	
Address Pointers	Comp0 16 bit Addr Pointer	YYYYYY11(l) YYYYYY10(h)
	Capt0 16 bit Addr Pointer	YYYYYY01(l) YYYYYY00(h)
DMA Counters	Comp0 DMA 16 bit Counter	XXXXXX11(l) XXXXXX10(h)
	Capt0 DMA 16 bit Counter	XXXXXX01(l) XXXXXX00(h)

■

Register File		
8 bit Counter	XXXXXX11	Compare 0
8 bit Addr Pointer	XXXXXX10	
8 bit Counter	XXXXXX01	Capture 0
8 bit Addr Pointer	XXXXXX00	

■

Each Timer DMA transaction is a 16-bit operation, therefore two bytes must be transferred sequentially, by means of two DMA transfers. In order to speed up each word transfer, the second byte transfer is executed by automatically forcing the peripheral priority to the highest level (000), regardless of the previously set level. It is then restored to its original value after executing the transfer. Thus, once a request is being serviced, its hardware priority is kept at the highest level regardless of the other Timer internal sources, i.e. once a Comp0 request is being serviced, it maintains a higher priority, even if a Capt0 request occurs between the two byte transfers.

After a complete data table transfer, the transaction counter is reset and an End Of Block (EOB) condition occurs, the block transfer is completed.

The End Of Block Interrupt routine must at this point reload both address and counter pointers of the channel referred to by the End Of Block interrupt source, if the application

PRESCALER REGISTER (PRSR)
R251 - Read/Write
Register Page: 10
Reset value: 0000 0000 (00h)

7							0
P7	P6	P5	P4	P3	P2	P1	P0

This register holds the preset value for the 8-bit prescaler. The PRSR content may be modified at any time, but it will be loaded into the prescaler at the following prescaler underflow, or as a consequence of a counter reload (either by software or upon external request).

Following a RESET condition, the prescaler is automatically loaded with 00h, so that the prescaler divides by 1 and the maximum counter clock is generated (Crystal oscillator clock frequency divided by 6 when MODER.5 = DIV2 bit is set).

The binary value programmed in the PRSR register is equal to the divider value minus one. For example, loading PRSR with 24 causes the prescaler to divide by 25.

OUTPUT A CONTROL REGISTER (OACR)
R252 - Read/Write
Register Page: 10
Reset value: 0000 0000

7							0
C0E0	C0E1	C1E0	C1E1	OUE0	OUE1	CEV	0P

Bits 7:6 = C0E[0:1] : COMP0 action bits.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when a successful compare of the CMP0R register occurs. Refer to Table for the list of actions that can be configured.

Bits 5:4 = C1E[0:1]: COMP1 action bits.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when a successful compare of the CMP1R register occurs. Refer to Table for the list of actions that can be configured.

Bits 3:2 = OUE[0:1] : OVF/UNF action bits.
These bits are set and cleared by software. They configure the action to be performed on the TxOUTA pin when an Overflow or Underflow of the U/D counter occurs. Refer to Table for the list of actions that can be configured.

Table 59. Output A action bits

xxE0	xxE1	Action on TxOUTA pin when an xx event occurs
0	0	Set
0	1	Toggle

byte. Any byte received after an SOF but before an EODM is considered a message byte type. Any byte received after an SOF, EODM and NBx is an IFR type.
RDT gets set or cleared at the same time that RDRF gets set.
RDT is cleared on reset or while CONTROL.JE is reset, or while CONTROL.JDIS bit is set.

0: Last RXDATA byte was a message type byte

1: Last RXDATA byte was a IRF type byte

Bit 2 = *End of Data Minimum Flag.*
The EODM flag is set when the JBLPD decoded VPWI pin has been in a passive state for longer that the minimum Tv3 symbol time unless the EODM is inhibited by a sleep, filter or CRCE, IBD, IFD or RBRK error condition during a frame. EODM bit does not get set when in the sleep mode or when a message is filtered.
The EODM bit can be cleared by software writing a logic “zero” in the EODM position. EODM is cleared on reset, while CONTROL.JE is reset or while CONTROL.JDIS bit is set.
If the EODM_M bit of the IMR register is set, when this bit is set an interrupt request occurs.

0: No EOD symbol detected

1: EOD symbol detected

Note: The EODM bit is not an error flag. It means that the minimum time related to the passive Tv3 symbol is passed.

Bit 1 = *End of Frame Minimum Flag.*
The EOFM flag is set when the JBLPD decoded VPWI pin has been in a passive state for longer that the minimum Tv4 symbol time. EOFM will still get set at the end of filtered frames or frames where sleep mode was invoked. Consequently, multiple EOFM flags may be encountered between frames of interest.
The EOFM bit can be cleared by software writing a logic “zero” in the EOFM position. EOFM is cleared on reset, while CONTROL.JE is reset or while CONTROL.JDIS bit is set.
If the EOFM_M bit of the IMR register is set, when this bit is set an interrupt request occurs.

0: No EOF symbol detected

1: EOF symbol detected

Note: The EOFM bit is not an error flag. It means that the minimum time related to the passive Tv4 symbol is passed.

Bit 0 = *Idle Bus Flag*
IDLE is set when the JBLPD decoded VPWI pin recognized an IFS symbol. That is, an idle bus is when the bus has been in a passive state for longer that the Tv6 symbol time. The IDLE flag will remain set as long as the decoded VPWI pin is passive. IDLE is cleared when the decoded VPWI pin transitions to an active state.
Note that if the VPWI pin remains in a passive state after JE is set, then the IDLE bit may go high sometime before a Tv6 symbol is timed on VPWI (since VPWI timers may be active when JE is clear).
IDLE is cleared on reset or while the CONTROL.JDIS bit is set.

0: J1850 bus not in idle state

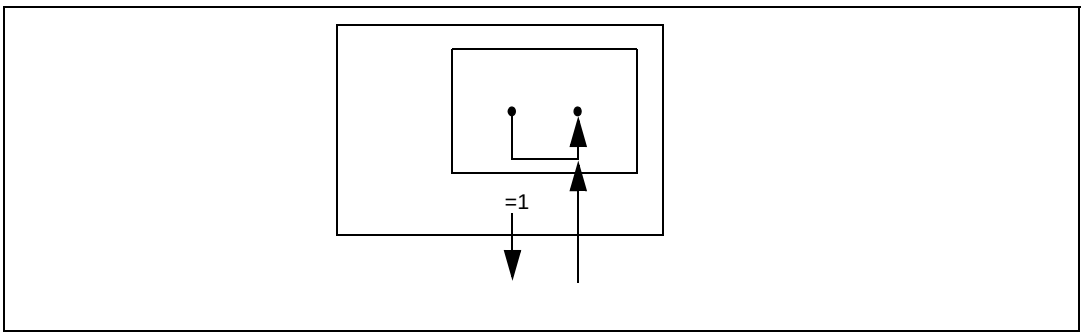
1: J1850 bus in idle state

After the SLEEP bit has been cleared, sleep mode is exited once bxCAN has synchronized with the CAN bus, refer to <Blue HT>Figure 157.bxCAN operating modes. The sleep mode is exited once the SLAK bit has been cleared by hardware.

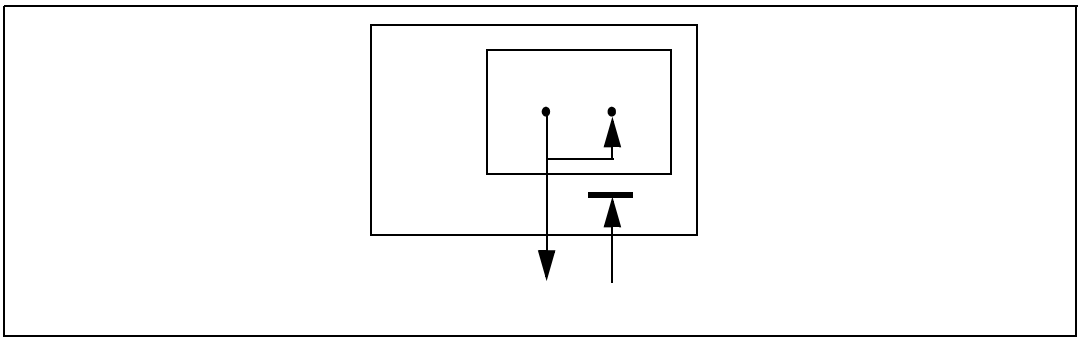
Test mode can be selected by the SILM and LBKM bits in the CDGR register. These bits must be configured while bxCAN is in Initialization mode. Once test mode has been selected, the INRQ bit in the CMCR register must be reset to enter Normal mode.

The bxCAN can be put in Silent mode by setting the SILM bit in the CDGR register.

In Silent mode, the bxCAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the bxCAN has to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. Silent mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits (Acknowledge Bits, Error Frames).



The bxCAN can be set in Loop Back Mode by setting the LBKM bit in the CDGR register. In Loop Back Mode, the bxCAN treats its own transmitted messages as received messages and stores th.em (if they pass acceptance filtering) in a Receive mailbox. bxCAN in Loop Back Mode.



This mode is provided for self-test functions. To be independent of external events, the CAN Core ignores acknowledge errors (no dominant bit sampled in the acknowledge slot of a

0: No interrupt on FMP[1:0] bits transition from 00b to 01b.
1: Interrupt generated on FMP[1:0] bits transition from 00b to 01b.

Bit 0 = *Transmit Mailbox Empty Interrupt Enable*
0: No interrupt when RQCPx bit is set.
1: Interrupt generated when RQCPx bit is set.

Note: Refer to Standard Interrupts Section.

Read / Write
Reset Value: 0000 0000 (00h)

7							0
0	LEC2	LEC1	LEC0	0	BOFF	EPVF	EWGF

Bit 7 = Reserved. Forced to 0 by hardware.

Bit 6:4 = *Last Error Code*
- Read/Set/Clear
This field holds a code which indicates the type of the last error detected on the CAN bus. If a message has been transferred (reception or transmission) without error, this field will be cleared to ‘0’. The code 7 is unused and may be written by the CPU to check for update

●

0	No Error
1	Stuff Error
2	Form Error
3	Acknowledgment Error
4	Bit recessive Error
5	Bit dominant Error
6	CRC Error
7	Set by software

Bit 3 = Reserved. Forced to 0 by hardware.

Bit 2 = *Bus-Off Flag*
- Read
This bit is set by hardware when it enters the bus-off state. The bus-off state is entered on TECR overrun, TEC greater than 255, refer to *Section on page 421*.

Bit 5 = *Transmission Error*
- Read/Clear
This bit is updated by hardware after each transmission attempt.

0: The previous transmission was successful
1: The previous transmission failed due to an error

Bit 4 = *Arbitration Lost*
- Read/Clear
This bit is updated by hardware after each transmission attempt.

0: The previous transmission was successful
1: The previous transmission failed due to an arbitration lost

Bit 3 = *Transmission OK*
- Read/Clear
The hardware updates this bit after each transmission attempt.

0: The previous transmission failed
1: The previous transmission was successful

Note: This bit has the same value as the corresponding TXOKx bit in the CTSR register.

Bit 2 = *Request Completed*
- Read/Clear
Set by hardware when the last request (transmit or abort) has been performed.

Cleared by software writing a “1” or by hardware on transmission request.

Note: This bit has the same value as the corresponding RQCPx bit of the CTSR register.

Clearing this bit clears all the status bits (TXOK, ALST and TERR) in the MCSR register and the RQCP and TXOK bits in the CTSR register.

Bit 1 = *Abort Request for Mailbox*
- Read/Set
Set by software to abort the transmission request for the corresponding mailbox.

Cleared by hardware when the mailbox becomes empty.

Setting this bit has no effect when the mailbox is not pending for transmission.

Bit 0 = *Transmit Mailbox Request*

- Read/Set
Set by software to request the transmission for the corresponding mailbox.

Cleared by hardware when the mailbox becomes empty.

Note: This register is implemented only in transmit mailboxes. In receive mailboxes, the MFMI register is mapped at this location.

7							0
D3.9	D3.8	D3.7	D3.6	D3.5	D3.4	D3.3	D3.2

R247 - Read/Write
Register Page: 61
Reset Value: xx00 0000

7							0
D3.1	D3.0	0	0	0	0	0	0

Bits 7:0 = : *Channel 3 1:0 bit Data*

Bits 5:0 = Reserved, forced by hardware to 0.

R248 - Read/Write
Register Page: 61
Reset Value: undefined

7							0
D4.9	D4.8	D4.7	D4.6	D4.5	D4.4	D4.3	D4.2

Bits 7:0 = : *Channel 4 9:2 bit Data*

R249 - Read/Write
Register Page: 61
Reset Value: xx00 0000

7							0
D4.1	D4.0	0	0	0	0	0	0

Bits 7:6 = : *Channel 4 1:0 bit Data*

Bits 5:0 = Reserved, forced by hardware to 0.

R250 - Read/Write
Register Page: 61
Reset Value: undefined

7							0
D5.9	D5.8	D5.7	D5.6	D5.5	D5.4	D5.3	D5.2

Bits 7:0 = : *Channel 5 9:2 bit Data*

R252 - Read/Write
Register Page: 62
Reset Value: undefined

7							0
D14.9	D14.8	D14.7	D14.6	D14.5	D14.4	D14.3	D14.2

Bits 7:0 = : *Channel 14 9:2 bit Data*

R253 - Read/Write
Register Page: 62
Reset Value: xx00 0000

7							0
D14.1	D14.0	0	0	0	0	0	0

Bits 7:0 = : *Channel 14 1:0 bit Data*

Bits 5:0 = Reserved, forced by hardware to 0.

R254 - Read/Write
Register Page: 62
Reset Value: undefined

7							0
D15.9	D15.8	D15.7	D15.6	D15.5	D15.4	D15.3	D15.2

Bits 7:0 = : *Channel 15 9:2 bit Data*

R255- Read/Write
Register Page: 62
Reset Value: xx00 0000

7							0
D15.1	D15.0	0	0	0	0	0	0

Bits 7:0 = : *Channel 15 1:0 bit Data*

Bits 5:0 = Reserved, forced by hardware to 0.

Note: *If only 8-bit accuracy is required, each Data High Register can be used to get the conversion result, ignoring the corresponding DxLR register content.*

●● (continued)

●	■		●			■
			■			
I_{LKIO}	I/O Pin Input Leakage	Input or Tri-State mode, $0V < V_{IN} < V_{DD}$	– 1		1	µA
I_{LKIOD}	I/O Pin Open Drain Input Leakage	Input or Tri-State mode, $0V < V_{IN} < V_{DD}$	– 1		1	µA
$\lvert I_{L-KADC} \rvert$	ADC Conv.Input leakage current on robust pins	$V_{IN}<V_{SS}$, $\lvert I_{IN} \rvert < 400µA$ on robust analog pin			6	µA
	ADC Conv.Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			1	µA
I_{IO}	Load current	P4[7:6]-P6[5:4] EMR1.BSZ bit = 1 (3)			8 (4)	mA
		P4[7:6]-P6[5:4] EMR1.BSZ bit = 0 (3)			2 (4)	
		All other pins except OSCOUT			2 (4)	
$\lvert I_{OV} \rvert$	Overload Current	(5)			5 (4)	mA
SR_R	Slew Rate Rise	(6)	20		30	ns
SR_F	Slew Rate Fall	(6)	20		30	ns

1. Unless otherwise stated, typical data are based on T_A= 25°C and V_{DD}= 5V. They are only reported for design guide lines not tested in production.
2. Value guaranteed by characterization.
3. For a description of the EMR1 Register - BSZ bit refer to the External Memory Interface Chapter.
4. Value guaranteed by Design.
5. Not tested in production, guaranteed by product characterisation. An overload condition occurs when the input voltage on any pin exceeds the specified voltage range.
6. Indicative values extracted from design simulation, 20% to 80% on 50pF load, EMR1.BSZ bit =0.

●

Test conditions: V_{DD} = 5 V ± 10%, T_A = –40° C to +125° C for Max values and 25° C for Typ values, unless otherwise specified.

●●

●	■				■	
I_{DDRUN}	Run Mode Current	CPU running with code execution from RAM memory, all peripherals in reset state, clock input (OSCIN) driven by external square wave. f_{INTCLK} in [MHz].	24 MHz	45	60	mA
			any frequency	2.5 + $1.8xf_{INTCLK}$/MHz		mA
ΔI_{DD1}	FLASH/E^3 TM Supply Current (Read) (2)		-	2		mA

■

Because the while loop lasts 16 CPU cycles, if $f_{CPU} \leq 16MHz$ at high baud rate, it is possible to miss a dominant state on the bus if it lasts just one CAN bit time and the bus speed is high enough (see *Table 143*)

●■

	■
24 MHz	No dominant bit missed
16 MHz	1 MBaud
8 MHz	> 500 kHz
4 MHz	> 250 kHz
f_{CPU}	> f_{CPU} / 16

Note: *As can be seen from the above table, no side effect occurs in cases when f_{CPU} is 16MHz or higher and if the CAN baud rate is below 1 MBaud.*

If this happens, we will continue waiting in the while loop instead of releasing the FIFO immediately. The workaround is still valid because we will not release the FIFO during the critical period. But the application may lose additional time waiting in the while loop as we are no longer able to guarantee a maximum of 6 CAN bit times spent in the workaround.

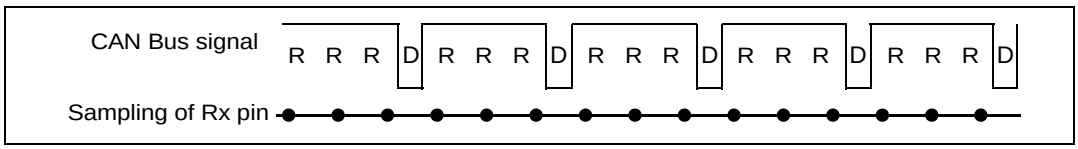
In this particular case the time the application can spend in the workaround may increase up to a full CAN frame, depending of the frame contents. This case is very rare but happens when a specific sequence is present on in the CAN frame.

The example in *Figure 206* shows reception if TCAN is $12/f_{CPU}$ and the sampling time is $16/f_{CPU}$.

If the application is using the maximum baud rate and the possible delay caused by the workaround is not acceptable, there is another workaround which reduces the Rx pin sampling time.

■

Workaround 2 (see *Figure 207*) first tests that FMP=2 and the CAN cell is receiving, if not the FIFO can be released immediately. If yes, the program goes through a sequence of test instructions on the RX pin that last longer than the time between the acknowledge dominant bit and the critical time slot. If the Rx pin is in recessive state for more than 8 CAN bit times, it means we are now after the acknowledge and the critical slot. If a dominant bit is read on the bus, we can release the FIFO immediately. This workaround has to be written in assembly language to avoid the compiler optimizing the test sequence.

The implementation shown here is for the CAN bus maximum speed (1MBd @ 8MHz CPU clock).

■ ■