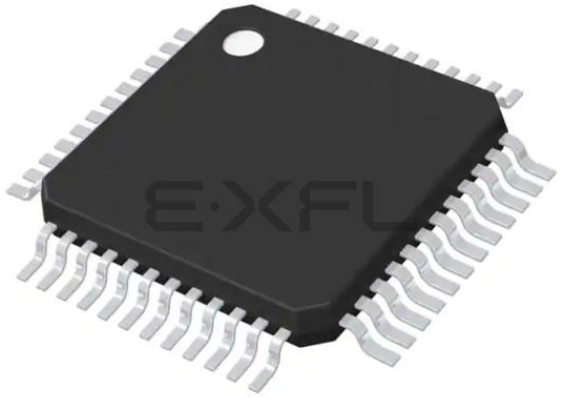
Infineon Technologies - XC8866FFI5VACFXUMA1 Datasheet



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	Last Time Buy
Core Processor	XC800
Core Size	8-Bit
Speed	24MHz
Connectivity	SSI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	34
Program Memory Size	24KB (24K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.75K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	PG-TQFP-48
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/xc8866ffi5vacfxuma1

2.2 Logic Symbol

The logic symbols of the XC886/888 are shown in **Figure 3**.

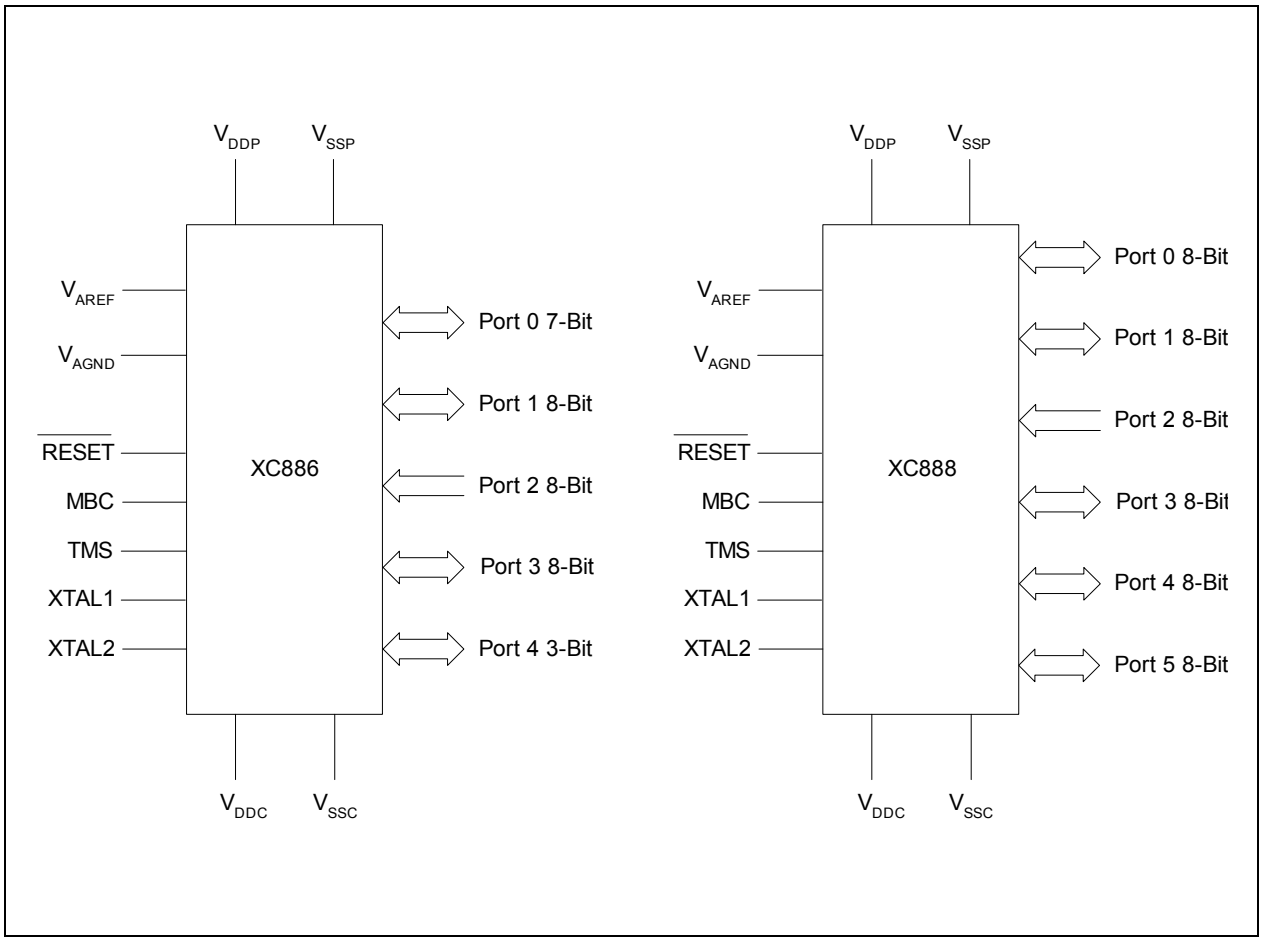


Figure 3 XC886/888 Logic Symbol

Table 3 Pin Definitions and Functions (cont'd)

Symbol	Pin Number (TQFP-48/64)	Type	Reset State	Function	
P4		I/O		**Port 4** Port 4 is an 8-bit bidirectional general purpose I/O port. It can be used as alternate functions for CCU6, Timer 0, Timer 1, Timer 21 and MultiCAN.	
P4.0	45/59		Hi-Z	RXDC0_3 CC60_1	MultiCAN Node 0 Receiver Input Output of Capture/Compare channel 0
P4.1	46/60		Hi-Z	TXDC0_3 COUT60_1	MultiCAN Node 0 Transmitter Output Output of Capture/Compare channel 0
P4.2	–/61		PU	EXINT6_1 T21_0	External Interrupt Input 6 Timer 21 Input
P4.3	32/40		Hi-Z	EXF21_1 COUT63_2	Timer 21 External Flag Output Output of Capture/Compare channel 3
P4.4	–/45		Hi-Z	CCPOS0_3 T0_0 CC61_4	CCU6 Hall Input 0 Timer 0 Input Output of Capture/Compare channel 1
P4.5	–/46		Hi-Z	CCPOS1_3 T1_0 COUT61_2	CCU6 Hall Input 1 Timer 1 Input Output of Capture/Compare channel 1
P4.6	–/47		Hi-Z	CCPOS2_3 T2_0 CC62_2	CCU6 Hall Input 2 Timer 2 Input Output of Capture/Compare channel 2
P4.7	–/48		Hi-Z	$\overline{\text{CTRAP}}$_3 COUT62_2	CCU6 Trap Input Output of Capture/Compare channel 2

Table 10 Port Register Overview (cont'd)

Addr	Register Name	Bit	7	6	5	4	3	2	1	0
93_H	**P5_ALTSEL1 Reset: 00_H** P5 Alternate Select 1 Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$B0_H$	**P3_ALTSEL0 Reset: 00_H** P3 Alternate Select 0 Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$B1_H$	**P3_ALTSEL1 Reset: 00_H** P3 Alternate Select 1 Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$C8_H$	**P4_ALTSEL0 Reset: 00_H** P4 Alternate Select 0 Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$C9_H$	**P4_ALTSEL1 Reset: 00_H** P4 Alternate Select 1 Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
RMAP = 0, PAGE 3										
80_H	**P0_OD Reset: 00_H** P0 Open Drain Control Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
90_H	**P1_OD Reset: 00_H** P1 Open Drain Control Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
92_H	**P5_OD Reset: 00_H** P5 Open Drain Control Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$B0_H$	**P3_OD Reset: 00_H** P3 Open Drain Control Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw
$C8_H$	**P4_OD Reset: 00_H** P4 Open Drain Control Register	Bit Field	P7	P6	P5	P4	P3	P2	P1	P0
		Type	rw	rw	rw	rw	rw	rw	rw	rw

3.2.4.7 ADC Registers

The ADC SFRs can be accessed in the standard memory area (RMAP = 0).

Table 11 ADC Register Overview

Addr	Register Name	Bit	7	6	5	4	3	2	1	0
RMAP = 0										
$D1_H$	**ADC_PAGE Reset: 00_H** Page Register	Bit Field	OP		STNR		0	PAGE		
		Type	w		w		r	rw		
RMAP = 0, PAGE 0										
CA_H	**ADC_GLOBCTR Reset: 30_H** Global Control Register	Bit Field	ANON	DW	CTC		0			
		Type	rw	rw	rw		r			
CB_H	**ADC_GLOBSTR Reset: 00_H** Global Status Register	Bit Field	0		CHNR			0	SAMP LE	BUSY
		Type	r		rh			r	rh	rh
CC_H	**ADC_PRAR Reset: 00_H** Priority and Arbitration Register	Bit Field	ASEN 1	ASEN 0	0	ARBM	CSM1	PRIO1	CSM0	PRIO0
		Type	rw	rw	r	rw	rw	rw	rw	rw

Table 17 CAN Register Overview (cont'd)

Addr	Register Name	Bit	7	6	5	4	3	2	1	0
DB_H	**DATA0** **Reset: 00_H** CAN Data Register 0	Bit Field	CD							
		Type	rwh							
DC_H	**DATA1** **Reset: 00_H** CAN Data Register 1	Bit Field	CD							
		Type	rwh							
DD_H	**DATA2** **Reset: 00_H** CAN Data Register 2	Bit Field	CD							
		Type	rwh							
DE_H	**DATA3** **Reset: 00_H** CAN Data Register 3	Bit Field	CD							
		Type	rwh							

3.2.4.14 OCDS Registers

The OCDS SFRs can be accessed in the mapped memory area (RMAP = 1).

Table 18 OCDS Register Overview

Addr	Register Name	Bit	7	6	5	4	3	2	1	0
RMAP = 1										
$E9_H$	**MMCR2** **Reset: $1U_H$** Monitor Mode Control 2 Register	Bit Field	STMODE	EXBC	DSUSP	MBCON	ALTDI	MMEP	MMODE	JENA
		Type	rw	rw	rw	rwh	rw	rwh	rh	rh
$F1_H$	**MMCR** **Reset: 00_H** Monitor Mode Control Register	Bit Field	MEXIT_P	MEXIT	0	MSTEP	MRAMS_P	MRAMS	TRF	RRF
		Type	w	rwh	r	rw	w	rwh	rh	rh
$F2_H$	**MMSR** **Reset: 00_H** Monitor Mode Status Register	Bit Field	MBCAM	MBCIN	EXBF	SWBF	HWB3F	HWB2F	HWB1F	HWB0F
		Type	rw	rwh	rwh	rwh	rwh	rwh	rwh	rwh
$F3_H$	**MMBPCR** **Reset: 00_H** Breakpoints Control Register	Bit Field	SWBC	HWB3C		HWB2C		HWB1C	HWB0C	
		Type	rw	rw		rw		rw	rw	
$F4_H$	**MMICR** **Reset: 00_H** Monitor Mode Interrupt Control Register	Bit Field	DVECT	DRETR	COMRST	MSTSEL	MMUIE_P	MMUIE	RRIE_P	RRIE
		Type	rwh	rwh	rwh	rh	w	rw	w	rw
$F5_H$	**MMDR** **Reset: 00_H** Monitor Mode Data Transfer Register Receive	Bit Field	MMRR							
		Type	rh							
$F6_H$	**HWBPSR** **Reset: 00_H** Hardware Breakpoints Select Register	Bit Field	0			BPSEL_P	BPSEL			
		Type	r			w	rw			
$F7_H$	**HWBPDR** **Reset: 00_H** Hardware Breakpoints Data Register	Bit Field	HWBPxx							
		Type	rw							
EB_H	**MMWR1** **Reset: 00_H** Monitor Work Register 1	Bit Field	MMWR1							
		Type	rw							

3.4.3 Interrupt Priority

An interrupt that is currently being serviced can only be interrupted by a higher-priority interrupt, but not by another interrupt of the same or lower priority. Hence, an interrupt of the highest priority cannot be interrupted by any other interrupt request.

If two or more requests of different priority levels are received simultaneously, the request of the highest priority is serviced first. If requests of the same priority are received simultaneously, then an internal polling sequence determines which request is serviced first. Thus, within each priority level, there is a second priority structure determined by the polling sequence shown in **Table 21**.

Table 21 Priority Structure within Interrupt Level

Source	Level
Non-Maskable Interrupt (NMI)	(highest)
External Interrupt 0	1
Timer 0 Interrupt	2
External Interrupt 1	3
Timer 1 Interrupt	4
UART Interrupt	5
Timer 2,UART Normal Divider Overflow, MultiCAN, LIN Interrupt	6
ADC, MultiCAN Interrupt	7
SSC Interrupt	8
External Interrupt 2, Timer 21, UART1, UART1 Normal Divider Overflow, MDU, CORDIC Interrupt	9
External Interrupt [6:3], MultiCAN Interrupt	10
CCU6 Interrupt Node Pointer 0, MultiCAN interrupt	11
CCU6 Interrupt Node Pointer 1, MultiCAN Interrupt	12
CCU6 Interrupt Node Pointer 2, MultiCAN Interrupt	13
CCU6 Interrupt Node Pointer 3, MultiCAN Interrupt	14

Figure 20 shows the structure of an input-only port pin.

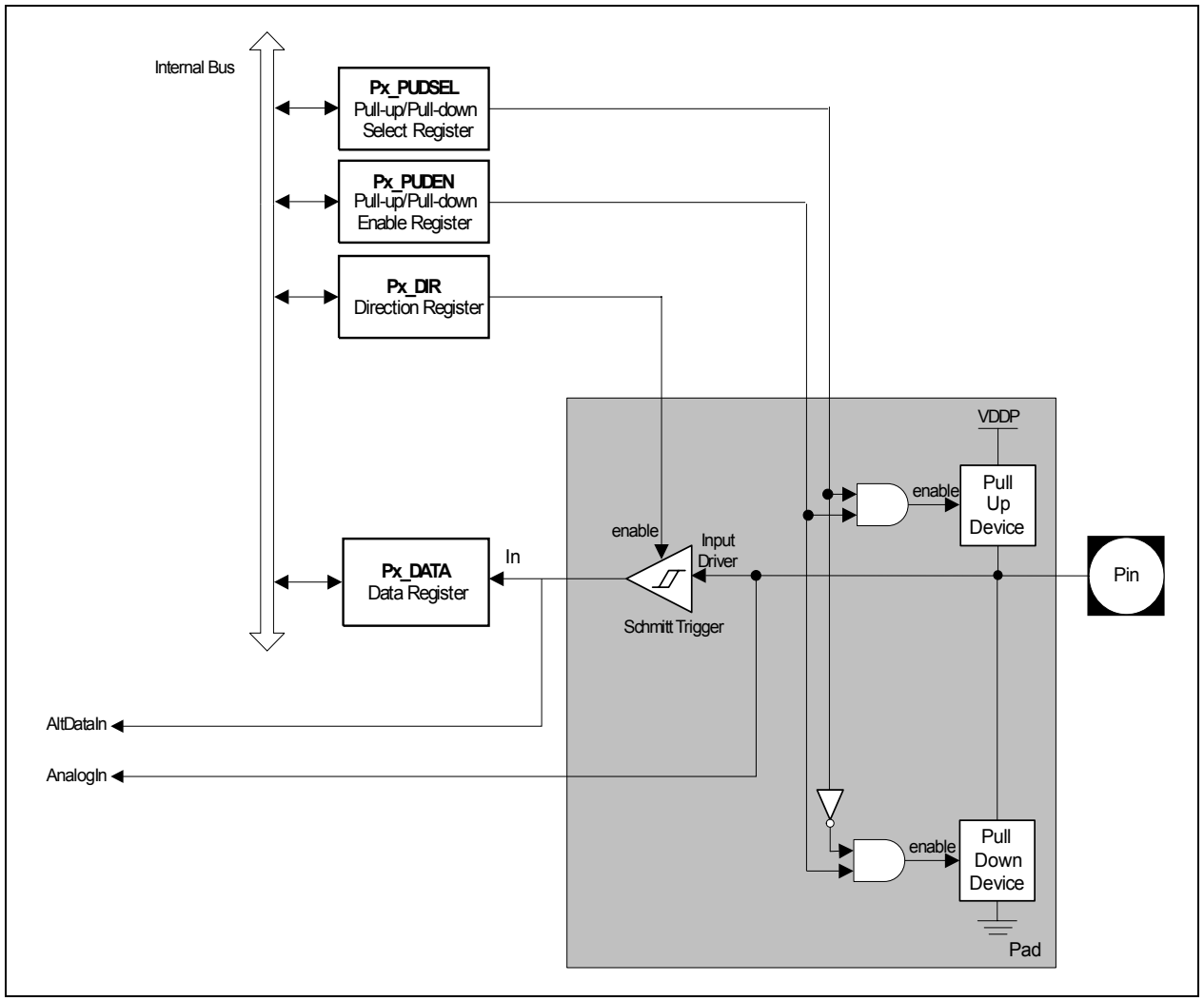


Figure 20 General Structure of Input Port

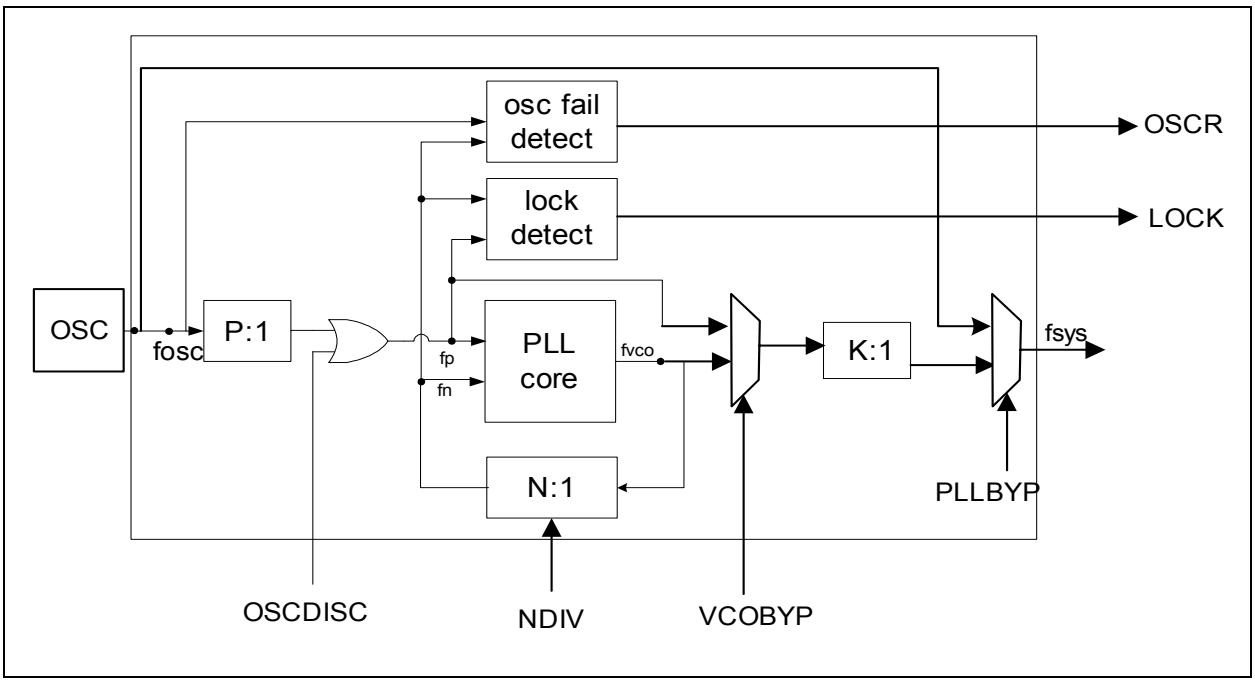


Figure 24 CGU Block Diagram

PLL Base Mode

When the oscillator is disconnected from the PLL, the system clock is derived from the VCO base (free running) frequency clock (**Table 25**) divided by the K factor.

$$f_{SYS} = f_{VCObase} \times \frac{1}{K} \tag{3.1}$$

Prescaler Mode (VCO Bypass Operation)

In VCO bypass operation, the system clock is derived from the oscillator clock, divided by the P and K factors.

$$f_{SYS} = f_{OSC} \times \frac{1}{P \times K} \tag{3.2}$$

For power saving purposes, the clocks may be disabled or slowed down according to Table 26.

Table 26 System frequency (f_{sys} = 96 MHz)

Power Saving Mode	Action
Idle	Clock to the CPU is disabled.
Slow-down	Clocks to the CPU and all the peripherals are divided by a common programmable factor defined by bit field CMCON.CLKREL.
Power-down	Oscillator and PLL are switched off.

3.10 Watchdog Timer

The Watchdog Timer (WDT) provides a highly reliable and secure way to detect and recover from software or hardware failures. The WDT is reset at a regular interval that is predefined by the user. The CPU must service the WDT within this interval to prevent the WDT from causing an XC886/888 system reset. Hence, routine service of the WDT confirms that the system is functioning properly. This ensures that an accidental malfunction of the XC886/888 will be aborted in a user-specified time period.

In debug mode, the WDT is default suspended and stops counting. Therefore, there is no need to refresh the WDT during debugging.

Features

- 16-bit Watchdog Timer
- Programmable reload value for upper 8 bits of timer
- Programmable window boundary
- Selectable input frequency of $f_{PCLK}/2$ or $f_{PCLK}/128$
- Time-out detection with NMI generation and reset prewarning activation (after which a system reset will be performed)

The WDT is a 16-bit timer incremented by a count rate of $f_{PCLK}/2$ or $f_{PCLK}/128$. This 16-bit timer is realized as two concatenated 8-bit timers. The upper 8 bits of the WDT can be preset to a user-programmable value via a watchdog service access in order to modify the watchdog expire time period. The lower 8 bits are reset on each service access. **Figure 28** shows the block diagram of the WDT unit.

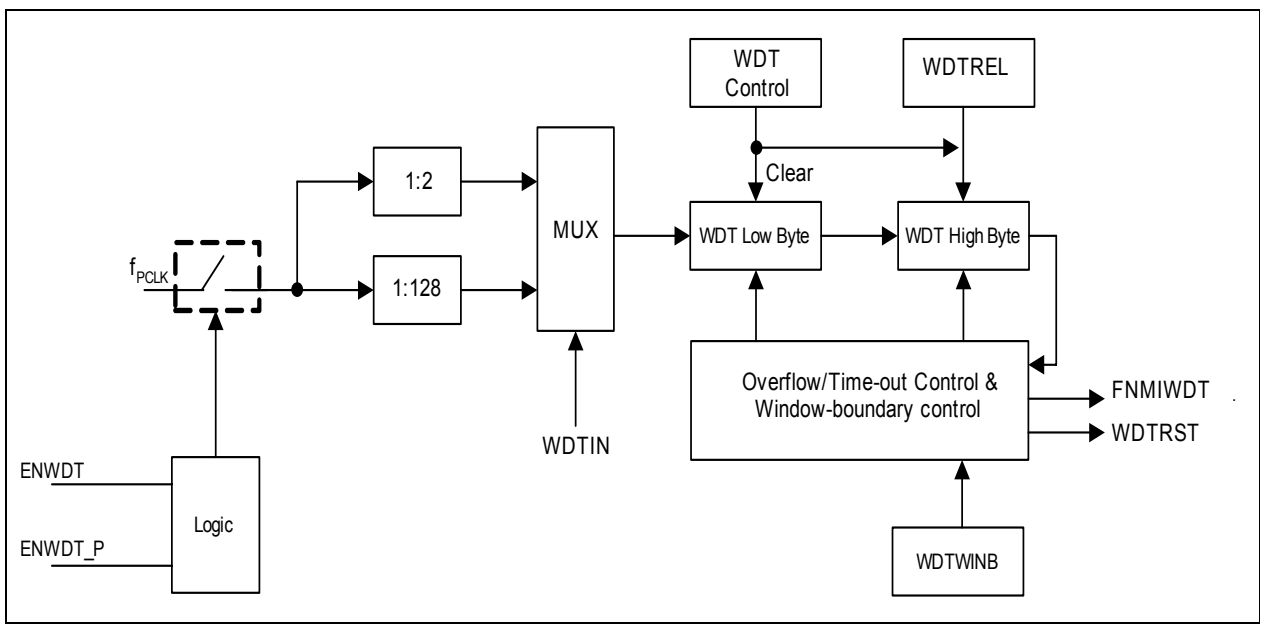


Figure 28 WDT Block Diagram

3.12 CORDIC Coprocessor

The CORDIC Coprocessor provides CPU with hardware support for the solving of circular (trigonometric), linear (multiply-add, divide-add) and hyperbolic functions.

Features

- Modes of operation
  - Supports all CORDIC operating modes for solving circular (trigonometric), linear (multiply-add, divide-add) and hyperbolic functions
  - Integrated look-up tables (LUTs) for all operating modes
- Circular vectoring mode: Extended support for values of initial X and Y data up to full range of $[-2^{15},(2^{15}-1)]$ for solving angle and magnitude
- Circular rotation mode: Extended support for values of initial Z data up to full range of $[-2^{15},(2^{15}-1)]$, representing angles in the range $[-\pi,((2^{15}-1)/2^{15})\pi]$ for solving trigonometry
- Implementation-dependent operational frequency of up to 80 MHz
- Gated clock input to support disabling of module
- 16-bit accessible data width
  - 24-bit kernel data width plus 2 overflow bits for X and Y each
  - 20-bit kernel data width plus 1 overflow bit for Z
  - With KEEP bit to retain the last value in the kernel register for a new calculation
- 16 iterations per calculation: Approximately 41 clock-cycles or less, from set of start (ST) bit to set of end-of-calculation flag, excluding time taken for write and read access of data bytes.
- Twos complement data processing
  - Only exception: X result data with user selectable option for unsigned result
- X and Y data generally accepted as integer or rational number; X and Y must be of the same data form
- Entries of LUTs are 20-bit signed integers
  - Entries of atan and atanh LUTs are integer representations (S19) of angles with the scaling such that $[-2^{15},(2^{15}-1)]$ represents the range $[-\pi,((2^{15}-1)/2^{15})\pi]$
  - Accessible Z result data for circular and hyperbolic functions is integer in data form of S15
- Emulated LUT for linear function
  - Data form is 1 integer bit and 15-bit fractional part (1.15)
  - Accessible Z result data for linear function is rational number with fixed data form of S4.11 (signed 4Q16)
- Truncation Error
  - The result of a CORDIC calculation may return an approximation due to truncation of LSBs
  - Good accuracy of the CORDIC calculated result data, especially in circular mode
- Interrupt
  - On completion of a calculation

Table 31 Deviation Error for UART with Fractional Divider enabled

f_{PCLK}	Prescaling Factor (2BRPRE)	Reload Value (BR_VALUE + 1)	STEP	Deviation Error
24 MHz	1	10 (A_H)	197 ($C5_H$)	+0.20 %
12 MHz	1	6 (6_H)	236 (EC_H)	+0.03 %
8 MHz	1	4 (4_H)	236 (EC_H)	+0.03 %
6 MHz	1	3 (3_H)	236 (EC_H)	+0.03 %

3.13.2 Baud Rate Generation using Timer 1

In UART modes 1 and 3 of UART module, Timer 1 can be used for generating the variable baud rates. In theory, this timer could be used in any of its modes. But in practice, it should be set into auto-reload mode (Timer 1 mode 2), with its high byte set to the appropriate value for the required baud rate. The baud rate is determined by the Timer 1 overflow rate and the value of SMOD as follows:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD} \times f_{PCLK}}{32 \times 2 \times (256 - TH1)} \quad (3.7)$$

3.14 Normal Divider Mode (8-bit Auto-reload Timer)

Setting bit FDM in register FDCON to 1 configures the fractional divider to normal divider mode, while at the same time disables baud rate generation (see **Figure 30**). Once the fractional divider is enabled (FDEN = 1), it functions as an 8-bit auto-reload timer (with no relation to baud rate generation) and counts up from the reload value with each input clock pulse. Bit field RESULT in register FDRES represents the timer value, while bit field STEP in register FDSTEP defines the reload value. At each timer overflow, an overflow flag (FDCON.NDOV) will be set and an interrupt request generated. This gives an output clock f_{MOD} that is 1/n of the input clock f_{DIV}, where n is defined by 256 - STEP.

The output frequency in normal divider mode is derived as follows:

$$f_{MOD} = f_{DIV} \times \frac{1}{256 - STEP} \quad (3.8)$$

3.15 LIN Protocol

The UART module can be used to support the Local Interconnect Network (LIN) protocol for both master and slave operations. The LIN baud rate detection feature, which consists of the hardware logic for Break and Synch Byte detection, provides the capability to detect the baud rate within LIN protocol using Timer 2. This allows the UART to be synchronized to the LIN baud rate for data transmission and reception.

Note: The LIN baud rate detection feature is available for use only with UART. To use UART1 for LIN communication, software has to be implemented to detect the Break and Synch Byte.

LIN is a holistic communication concept for local interconnected networks in vehicles. The communication is based on the SCI (UART) data format, a single-master/multiple-slave concept, a clock synchronization for nodes without stabilized time base. An attractive feature of LIN is self-synchronization of the slave nodes without a crystal or ceramic resonator, which significantly reduces the cost of hardware platform. Hence, the baud rate must be calculated and returned with every message frame.

The structure of a LIN frame is shown in **Figure 31**. The frame consists of the:

- Header, which comprises a Break (13-bit time low), Synch Byte (55_H), and ID field
- Response time
- Data bytes (according to UART protocol)
- Checksum

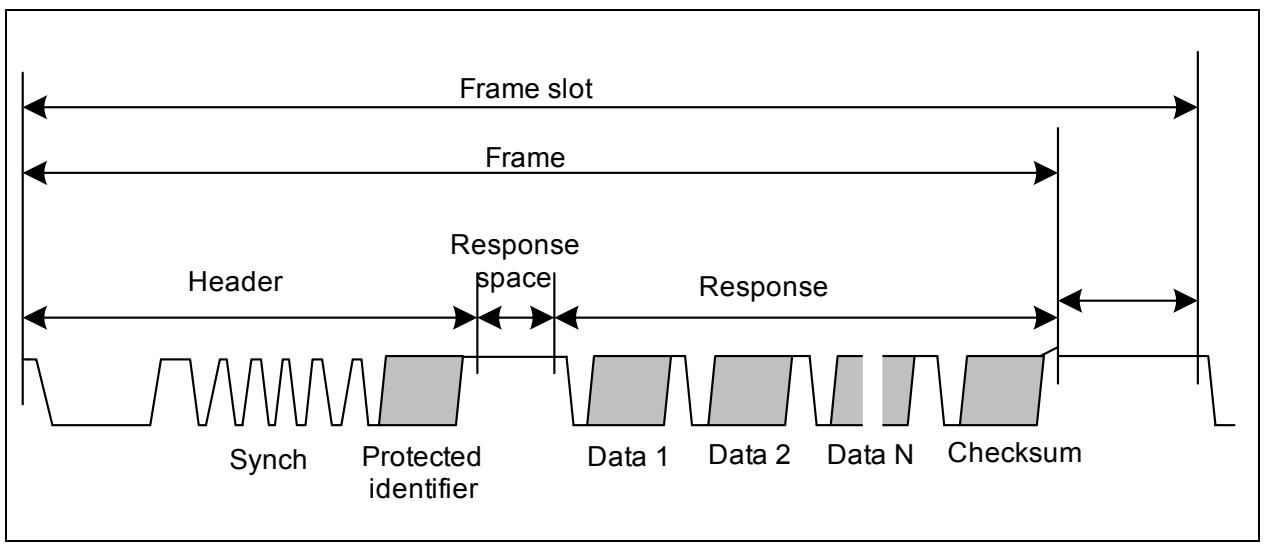


Figure 31 Structure of LIN Frame

3.15.1 LIN Header Transmission

LIN header transmission is only applicable in master mode. In the LIN communication, a master task decides when and which frame is to be transferred on the bus. It also identifies a slave task to provide the data transported by each frame. The information

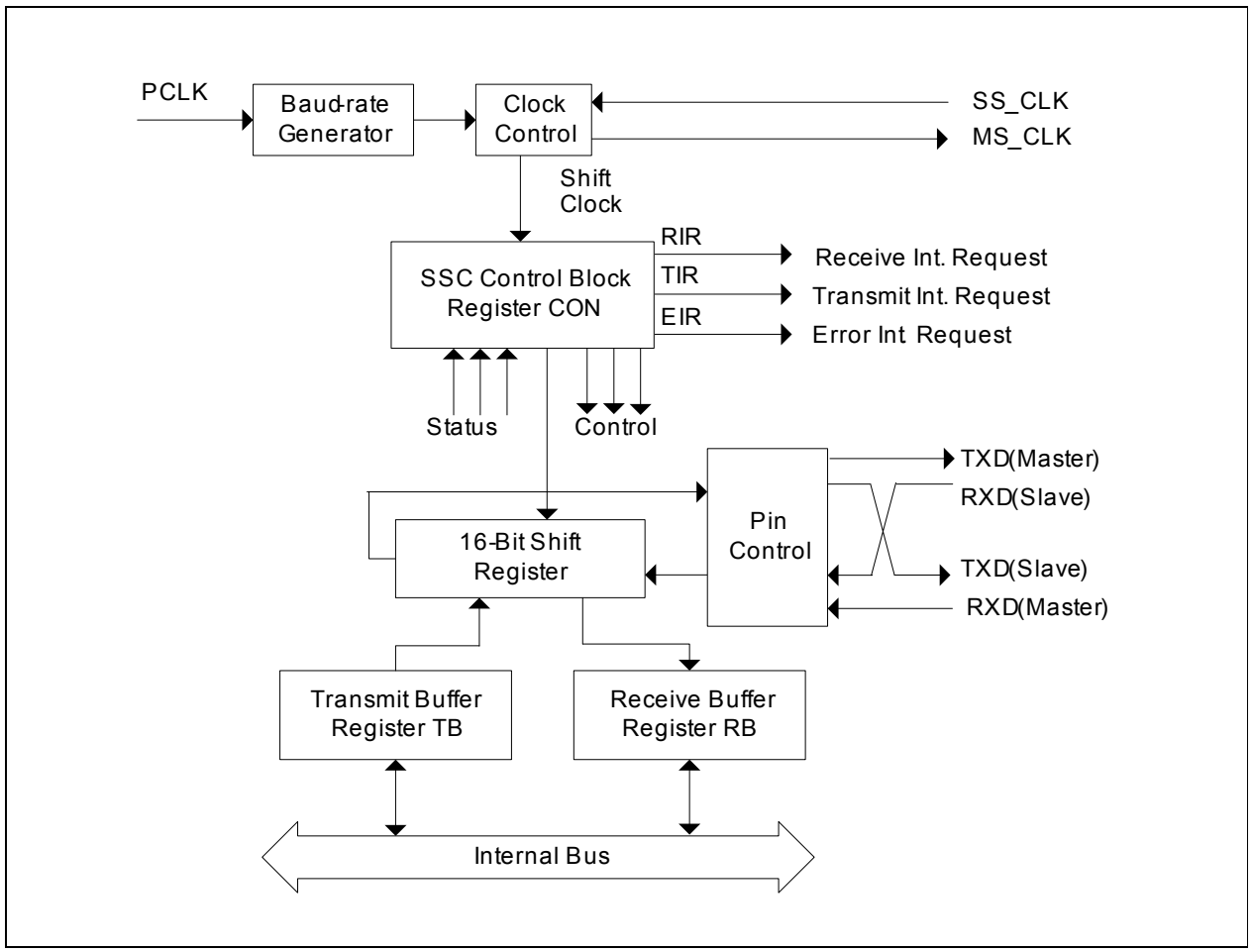


Figure 32 SSC Block Diagram

- CAN functionality according to CAN specification V2.0 B active.
- Dedicated control registers are provided for each CAN node.
- A data transfer rate up to 1 MBaud is supported.
- Flexible and powerful message transfer control and error handling capabilities are implemented.
- Advanced CAN bus bit timing analysis and baud rate detection can be performed for each CAN node via the frame counter.
- Full-CAN functionality: A set of 32 message objects can be individually
  - allocated (assigned) to any CAN node
  - configured as transmit or receive object
  - setup to handle frames with 11-bit or 29-bit identifier
  - counted or assigned a timestamp via a frame counter
  - configured to remote monitoring mode
- Advanced Acceptance Filtering:
  - Each message object provides an individual acceptance mask to filter incoming frames.
  - A message object can be configured to accept only standard or only extended frames or to accept both standard and extended frames.
  - Message objects can be grouped into 4 priority classes.
  - The selection of the message to be transmitted first can be performed on the basis of frame identifier, IDE bit and RTR bit according to CAN arbitration rules.
- Advanced Message Object Functionality:
  - Message Objects can be combined to build FIFO message buffers of arbitrary size, which is only limited by the total number of message objects.
  - Message objects can be linked to form a gateway to automatically transfer frames between 2 different CAN buses. A single gateway can link any two CAN nodes. An arbitrary number of gateways may be defined.
- Advanced Data Management:
  - The Message objects are organized in double chained lists.
  - List reorganizations may be performed any time, even during full operation of the CAN nodes.
  - A powerful, command driven list controller manages the organization of the list structure and ensures consistency of the list.
  - Message FIFOs are based on the list structure and can easily be scaled in size during CAN operation.
  - Static Allocation Commands offer compatibility with TwinCAN applications, which are not list based.
- Advanced Interrupt Handling:
  - Up to 8 interrupt output lines are available. Most interrupt requests can be individually routed to one of the 8 interrupt output lines.
  - Message postprocessing notifications can be flexibly aggregated into a dedicated register field of 64 notification bits.

3.22 On-Chip Debug Support

The On-Chip Debug Support (OCDS) provides the basic functionality required for the software development and debugging of XC800-based systems.

The OCDS design is based on these principles:

- Use the built-in debug functionality of the XC800 Core
- Add a minimum of hardware overhead
- Provide support for most of the operations by a Monitor Program
- Use standard interfaces to communicate with the Host (a Debugger)

Features

- Set breakpoints on instruction address and on address range within the Program Memory
- Set breakpoints on internal RAM address range
- Support unlimited software breakpoints in Flash/RAM code region
- Process external breaks via JTAG and upon activating a dedicated pin
- Step through the program code

The OCDS functional blocks are shown in **Figure 37**. The Monitor Mode Control (MMC) block at the center of OCDS system brings together control signals and supports the overall functionality. The MMC communicates with the XC800 Core, primarily via the Debug Interface, and also receives reset and clock signals.

After processing memory address and control signals from the core, the MMC provides proper access to the dedicated extra-memories: a Monitor ROM (holding the code) and a Monitor RAM (for work-data and Monitor-stack).

The OCDS system is accessed through the JTAG[1)], which is an interface dedicated exclusively for testing and debugging activities and is not normally used in an application. The dedicated MBC pin is used for external configuration and debugging control.

Note: All the debug functionality described here can normally be used only after XC886/888 has been started in OCDS mode.

1) The pins of the JTAG port can be assigned to either the primary port (Port 0) or either of the secondary ports (Ports 1 and 2/Port 5).
User must set the JTAG pins (TCK and TDI) as input during connection with the OCDS system.

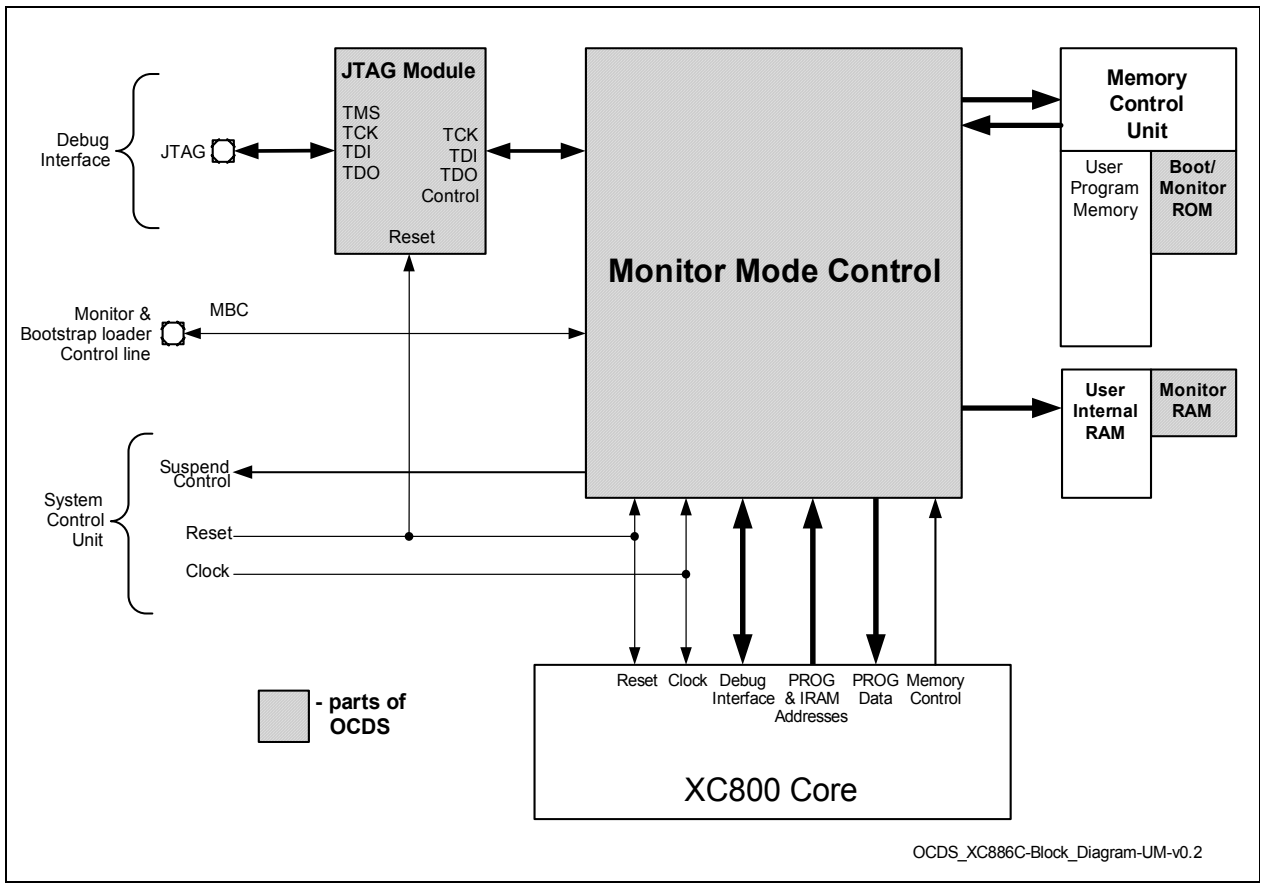


Figure 37 OCDS Block Diagram

3.22.1 JTAG ID Register

This is a read-only register located inside the JTAG module, and is used to recognize the device(s) connected to the JTAG interface. Its content is shifted out when INSTRUCTION register contains the IDCODE command (opcode 04_H), and the same is also true immediately after reset.

The JTAG ID register contents for the XC886/888 Flash devices are given in **Table 35**.

Table 35 JTAG ID Summary

Device Type	Device Name	JTAG ID
Flash	XC886/888*-8FF	$1012\ 0083_H$
	XC886/888*-6FF	$1012\ 5083_H$
ROM	XC886/888*-8RF	$1013\ C083_H$
	XC886/888*-6RF	$1013\ D083_H$

Note: The asterisk (*) above denotes all possible device configurations.

4.1.3 Operating Conditions

The following operating conditions must not be exceeded in order to ensure correct operation of the XC886/888. All parameters mentioned in the following table refer to these operating conditions, unless otherwise noted.

Table 37 Operating Condition Parameters

Parameter	Symbol	Limit Values		Unit	Notes/ Conditions
		min.	max.		
Digital power supply voltage	V_{DDP}	4.5	5.5	V	5V Device
Digital power supply voltage	V_{DDP}	3.0	3.6	V	3.3V Device
Digital ground voltage	V_{SS}	0		V	
Digital core supply voltage	V_{DDC}	2.3	2.7	V	
System Clock Frequency[1)]	f_{SYS}	88.8	103.2	MHz	
Ambient temperature	T_A	-40	85	°C	SAF-XC886/888...
		-40	125	°C	SAK-XC886/888...

1) f_{SYS} is the PLL output clock. During normal operating mode, CPU clock is f_{SYS} / 4. Please refer to **Figure 26** for detailed description.

4.2.3 ADC Characteristics

The values in the table below are given for an analog power supply between 4.5 V to 5.5 V. The ADC can be used with an analog power supply down to 3 V. But in this case, the analog parameters may show a reduced performance. All ground pins (V_{SS}) must be externally connected to one single star point in the system. The voltage difference between the ground pins must not exceed 200mV.

Table 40 ADC Characteristics (Operating Conditions apply; V_{DDP} = 5V Range)

Parameter	Symbol		Limit Values			Unit	Test Conditions/ Remarks
			min.	typ .	max.		
Analog reference voltage	V_{AREF}	SR	V_{AGND} + 1	V_{DDP}	V_{DDP} + 0.05	V	[1]
Analog reference ground	V_{AGND}	SR	V_{SS} - 0.05	V_{SS}	V_{AREF} - 1	V	[1]
Analog input voltage range	V_{AIN}	SR	V_{AGND}	–	V_{AREF}	V	
ADC clocks	f_{ADC}		–	24	25.8	MHz	module clock[1]
	f_{ADCI}		–	–	10	MHz	internal analog clock[1] See **Figure 35**
Sample time	t_S	CC	(2 + INPCR0.STC) × t_{ADCI}			µs	[1]
Conversion time	t_C	CC	See **Section 4.2.3.1**			µs	[1]
Total unadjusted error	|TUE|	CC	–	–	1	LSB	8-bit conversion[2]
			–	–	2	LSB	10-bit conversion[2]
Differential Nonlinearity	$\|EA_{DNL}\|$	CC	–	1	–	LSB	10-bit conversion[1]
Integral Nonlinearity	$\|EA_{INL}\|$	CC	–	1	–	LSB	10-bit conversion[1]
Offset	$\|EA_{OFF}\|$	CC	–	1	–	LSB	10-bit conversion[1]
Gain	$\|EA_{GAIN}\|$	CC	–	1	–	LSB	10-bit conversion[1]
Overload current coupling factor for analog inputs	K_{OVA}	CC	–	–	1.0 x 10^{-4}	–	$I_{OV} > 0$[1][3]
			–	–	1.5 x 10^{-3}	–	$I_{OV} < 0$[1][3]

4.3.3 Power-on Reset and PLL Timing

Table 49 provides the characteristics of the power-on reset and PLL timing in the XC886/888.

Table 46 Power-On Reset and PLL Timing (Operating Conditions apply)

Parameter	Symbol		Limit Values			Unit	Test Conditions
			min.	typ.	max.		
Pad operating voltage	V_{PAD}	CC	2.3	–	–	V	1)
On-Chip Oscillator start-up time	t_{OSCST}	CC	–	–	500	ns	1)
Flash initialization time	t_{FINIT}	CC	–	160	–	µs	1)
$\overline{\text{RESET}}$ hold time	t_{RST}	SR	–	500	–	µs	V_{DDP} rise time (10% – 90%) ≤ 500µs 1)2)
PLL lock-in in time	t_{LOCK}	CC	–	–	200	µs	1)
PLL accumulated jitter	D_P		–	–	0.7	ns	1)3)

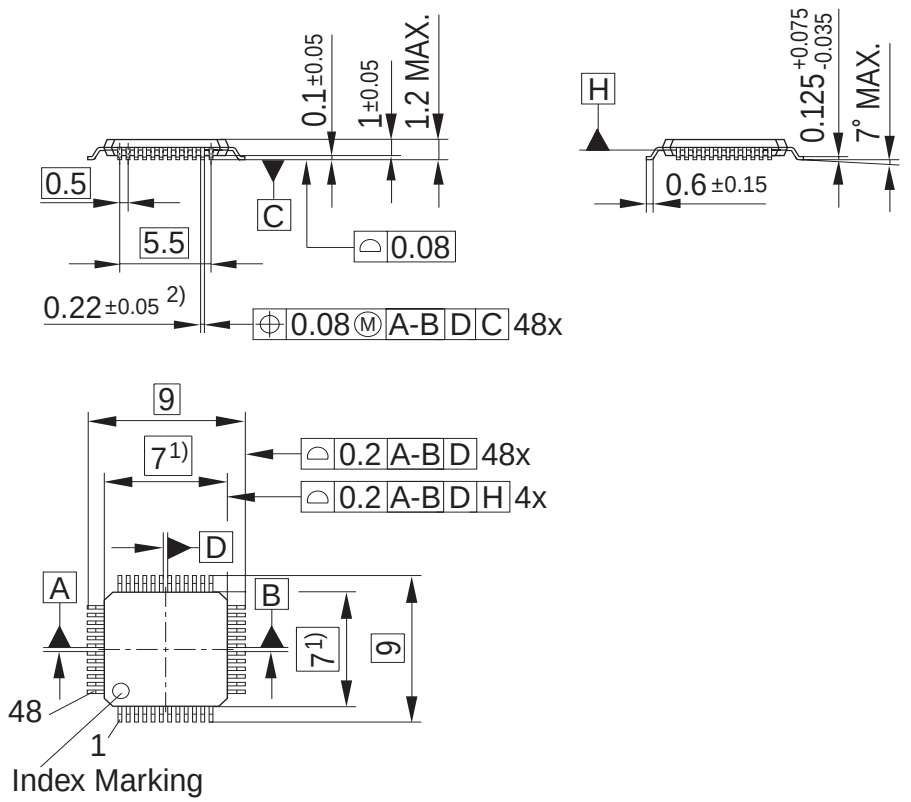
1) Not all parameters are 100% tested, but are verified by design/characterization and test correlation.

2) $\overline{\text{RESET}}$ signal has to be active (low) until V_{DDC} has reached 90% of its maximum value (typ. 2.5 V).

3) PLL lock at 96 MHz using a 4 MHz external oscillator. The PLL Divider settings are K = 2, N = 48 and P = 1.

5.2 Package Outline

Figure 48 shows the package outlines of the XC886.



1) Does not include plastic or metal protrusion of 0.25 max. per side
2) Does not include dambar protrusion of 0.08 max. per side

GPP09237

Figure 48 PG-TQFP-48 Package Outline