

Table of Contents

Introduction
Sample Sequencing in ACE
Design Example
Special Consideration During the Simultaneous Sampling
Conclusion
Appendix A: ACE Configuration and Connectivity
Appendix B: Special ACE Register for Design Example 12
Appendix C: Configuration for Simultaneous Sample Procedure
Appendix D: ACE Configuration and Procedure for Design Example 2
Appendix E: Design Files
List of Changes

Introduction

The mixed signal blocks found in the SmartFusion[®] customizable system-on-chip (cSoC) devices are controlled and connected to the rest of the system via a dedicated processor called the analog compute engine (ACE). The ACE is built to handle the sampling, sequencing, and post-processing of the ADCs, DACs, and signal conditioning blocks (SCBs). For a larger deterministic synchronization between the sampling and custom fabric logic, the sample sequencing in the ACE is controlled through the microprocessor subsystem (MSS) or FPGA fabric logic.

This application note provides the two design examples which allow deterministic synchronization between the sampling essential steps:

- "Design Example 1: ACE Sequencer Using Fabric Logic" on page 3
- "Design Example 2: Simultaneous Sampling Using the MSS" on page 5

This application note also assumes that you are familiar with the SmartFusion analog block. Refer to the *SmartFusion Programmable Analog User's Guide* for more information.

Sample Sequencing in ACE

The ACE in the SmartFusion cSoC is made up of two major blocks:

- Sample sequencing engine (SSE)
- Post processing engine (PPE)

The SSE offers a flexible configuration of the analog front end (AFE) resources (analog inputs and monitors, comparators, ADCs, and DACs), as well as a variety of simple and sophisticated sample sequencing. The PPE is a self sufficient block that allows the data processing such as linear transformation, filtering, and thresholds comparisions, etc. The ACE also has an interface with the AFE and the MSS/FPGA fabric.



It interfaces with the MSS and fabric is through the APB3 bus as depicted in Figure 1.

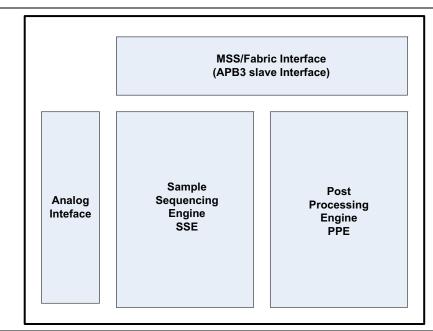


Figure 1 • Overview of ACE Architecture and Interfaces

Figure 2 provides an overview of the sample sequencing engine (SSE) block and its interfaces to PPE, APB3, ADCs, DACs, and ACB. The SSE block has the following:

- The sample sequencing instruction (SSI) unit multiplexes between APB3 and program instruction in SRAM, access to the analog fabric registers.
- Time division multiplexing finite state machine (TDM FSM): Creates the timeslots for each of the three ADCs, as well as a shared timeslot for APB3 or PPE accesses. The ADC timeslots are allocated during the ACE configuration in the SmartDesign.
- APB3/PPE interface: Arbitrates the SSE access between the APB3 and PPE.
- 24-bit phase accumulators: Front-end accumulators for the three SDDs.

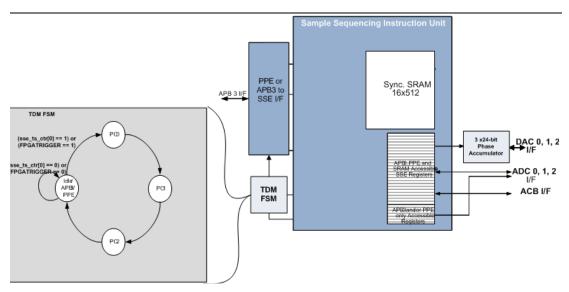


Figure 2 • Overview of ACE Architecture and Interfaces



The TDM FSM block (shown in Figure 2 on page 2) implements a mechanism to sequence through the ADCs. It allows the equal access by three separate program counters (PC0, PC1, and PC2) in the SSE and the APB3/PPE master. The SSE uses a separate program counters (PC) for sequencing and controlling each ADCs independently. When controlling the SSE from the APB3 master, the master needs to honor the wait-states generated by the *PREADY* signal during the normal operation of the SSE TDM timeslot counter.

Note: When the TDM timeslot counter is enabled for normal operation of the SSE block, there are always four timeslots, regardless of whether there are one, two, or three ADC instances in a specific device.

The TDM FSM is controlled by FABACETRIG signal or SSE_TS_CTRL register when bit 0 of

- SSE_TS_CTRL register or FABACETRIG is set to 1, it enables all 4 timeslots
- SSE_TS_CTRL or FABACETRIG set to 0, it only enables access to the APB3 master this fixes the PREADY signal high to allow the zero-wait state access to the APB3 master

The SmartFusion cSoC devices also have the capability of simultaneous sampling on different ADCs. Simultaneous synchronized ADC conversion control register (ADC_SYNC_CONV) allows for a single program to issue the synchronized start instructions for all ADCs.

Design Example

This section describes the design examples. This application note shows two design examples of controlling the ACE sequencer using the fabric logic and MSS.

The first design example shows a master in fabric, uses the *FABACETRIG* to access the APB3 interface in the ACE, and then control the SSE sampling. The second design example shows the simultaneous sampling on multiple ADCs using the MSS.

Note: The timeslots do not have any capability neither to block each other nor to signal each other. So, you need to use the special consideration during the simultaneous sampling. For more details, refer to "Special Consideration During the Simultaneous Sampling" on page 6.

Design Example 1: ACE Sequencer Using Fabric Logic

This design example shows a master in fabric controlling the SSE timeslot. It uses the *FABACETRIG* (refer to "Appendix A: ACE Configuration and Connectivity" on page 8) signal to access the APB3 interface in ACE and then control the SSE sampling. Figure 3 illustrates the concept of using the *FABACETRIG* to control the SSE sampling.

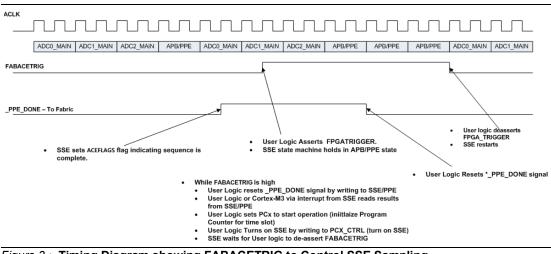


Figure 3 • Timing Diagram showing FABACETRIG to Control SSE Sampling



The fabric APB master is responsible for:

- Detecting that the ACE has reached the end of a sequence through the ACEFLAGS signal (ACEFLAGS are connected to ACE_TRIGGER in fabric).
- Assert the FABACETRIG signal to the ACE to halt the SSE time division multiplexing.
- Clear the *_PPE_DONE notification flag by writing to the ACE PPE_FLAGS0_IRQ_CLR registers (refer to "Appendix B: Special ACE Register for Design Example" on page 12).
- Notifying the rest of the fabric logic to begin a transaction. This is simulated in this design with the ENABLE_PWM signal that enable a CorePWM IP.
- Re-enabling the ACE sequencer by writing to the program counter enables bit and writing the next program counter address (refer to "Appendix B: Special ACE Register for Design Example" on page 12). The program counter address is found in the
 - <project>\firmware\drivers_config\mss_ace\ace_config.c file.

The starting loop address for a given procedure is specified in the *ace_procedure_dest_t* table. For example, in the structure below, "2" is the starting loop address for this procedure.

The block diagram of the design example is shown in Figure 4.

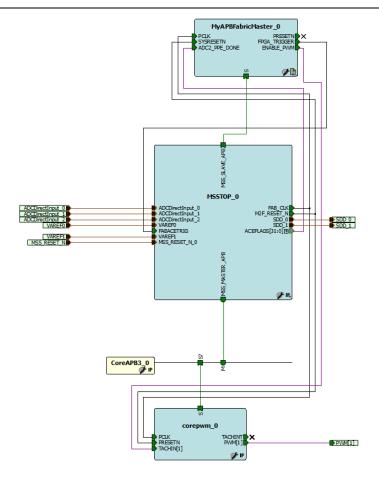


Figure 4 • Top Level Block Diagram of ACE Sequencer Using Fabric Logic



The design example is available for downloading at http://soc.microsemi.com/download/rsc/?f=A2F AC366 DF .

The following waveform (Figure 5) shows the ACE and fabric APB3 master interacting.

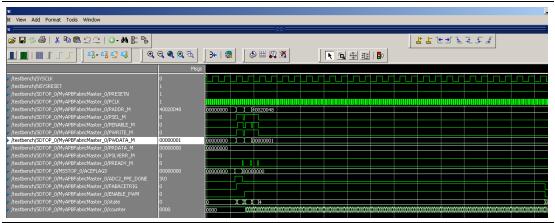


Figure 5 • ACE and Fabric APB3 Master Simulation

Design Example 2: Simultaneous Sampling Using the MSS

The second design example shows the simultaneous sampling on multiple ADCs using the MSS. Figure 6 shows the ACE configuration used in this design. "Appendix C: Configuration for Simultaneous Sample Procedure" on page 13 shows the option to enable simultaneous sampling in the ACE configurator.

	Controller Flags				
ADC Configuration					
ACLK: 80 MHz	ADC Clock: 10 MHz R	esolution: 10 - bits		Advanced Op	tions
					<i>a</i> l ~
Add service: Active Bipolar Prescaler Inpu			Sample		
ADC Direct Input	Service	Signal	time (us)	Package Pin	SCB
Current Input Differential Input	Active Bipolar Prescaler Inpu Volta		10.000 AA6		0(1)
Temperature Input	Active Bipolar Prescaler Inpu Volta		10.000 V8 (0(2)
LVTTL Input	Active Bipolar Prescaler Inpu Volta		10.000 AB7		1(6)
Analog Comparator	Active Bipolar Prescaler Inpu Volta		10.000 W9 10.000 W12		1(5)
Sigma Delta DAC	Active Bipolar Prescaler Inpu Volta Active Bipolar Prescaler Inpu Volta		10.000 W12		3(5) 3(6)
	Active Bipolar Prescaler Inpu Volta Active Bipolar Prescaler Inpu Volta		10.000 V13		2(2)
1	Active Bipolar Prescaler Input Volta Active Bipolar Prescaler Input Volta		10.000 V13		2(1)
	ADC Block 0				

Figure 6 • ACE Configuration for Simultaneous Sampling Design Example



The block diagram of the design example is shown in Figure 7. The design example Libero[®] System-on-Chip (cSoC) software projects are available for downloading at http://soc.microsemi.com/download/rsc/?f=A2F_AC366_DF

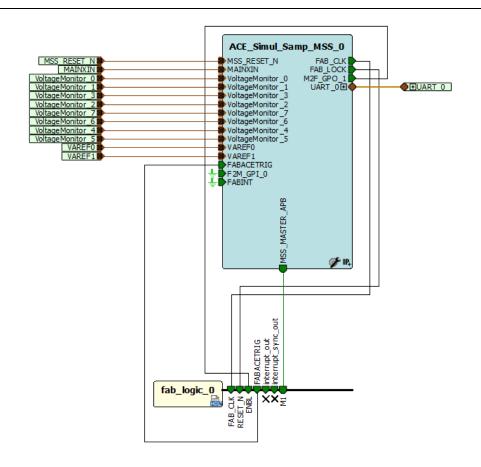


Figure 7 • Top Level Block Diagram of Simultaneous Sampling Design Example

Special Consideration During the Simultaneous Sampling

The ADC timeslots have the capability to access and control any of the ADCs. However, these timeslots do not have the capability to block or signal each other. Therefore, essentially they get two clock cycles each to execute the available microcode instruction, and then the SSE moves to the next timeslot. Due to this architectural advantage, you should take care of the following situation where the results might be wrong.

Assume that both the timeslot 0 and timeslot 1 are sampling the ADC0 channels. This means that the timeslot 0 executes its sample instructions and asserts the *ADCSTART* for the ADC0 on channel X. The ADC0 begins that process. Now the timeslot 1 is activated (after two clocks) and it also has an instruction that uses ADC0. There is no mechanism in the SSE to explicitly query whether the ADC is busy or not. Therefore, when that timeslot sees the sample microcode, it asserts another *ADCSTART* for the ADC0 for channel Y. Consequently, now you have the timeslot 0 waiting for a data valid from the ADC0 for channel X, and the timeslot 1 is waiting for a data valid from ADC0 for channel Y. When the ADC0 finally finishes and returns the data valid, both timeslots move onto their next respective microcode instructions. However, only one of them have the sample proper data. Only one piece of data is sent to the PPE, since it is the ADC data valid that is used to write into the *ADCFIFO* and only one ADC data valid signal allocated per ADC.



A technique is shown to solve this simultaneous sampling in "Design Example 2: Simultaneous Sampling Using the MSS" on page 5. Please refer to the technique below that solves the problem of simultaneous sampling on the same ADC.

Note: The two ADCs' procedure uses the simultaneous sample.

- 1. After power-on reset,
 - The MSS initializes the various peripherals.
 - The MSS loads the simultaneous sampling procedures into SSE to sample two channels per ADC (total four samples expected).
 - In the FPGA, FABACETRIG is set to 1.
 - The MSS sends an enable signal through a GPIO to FPGA (to convey that it can start the sampling sequence).
- 2. On seeing the GPIO set, FPGA starts 10 KHz counter.
- 3. On rising of 10 KHz, it brings FABACETRIG low after 1 μ s.
- 4. The ACE starts simultaneous sampling and interrupts M3 when samples are available. The procedure disables the timeslot (*PCx_EN* = 0).
- 5. The M3 gets the data from the ACE into variables and loads both the sequential sampling procedures (*seq_adc00* and *seq_adc01*). Sequential sampling starts.
- On completion, the M3 gets two interrupts for both the ADCs. The procedure disables both the timeslots (*PCx_EN* = 0). On serving both the interrupts (reading sample values), M3 does a write to fabric to signal end of sequence.
- 7. On detecting the write from M3, FPGA brings FABACETRIG high.
- 8. The M3 loads simultaneous sampling procedure (two ADCs).
- 9. Sequence repeats from step 3.

Conclusion

This application note provides the design examples of using the fabric logic or MSS to control the SSE sampling. This can be used to allow the deterministic synchronization.



Appendix A: ACE Configuration and Connectivity

Step1. MSS ACE Configuration for Design Example 1

1. Inside the ACE Configurator, click the **Advanced Options** dialog box. Select the **Expose FABACETRIG port** checkbox. This exposes the *FABACETRIG* port on the ACE block enabling the fabric logic to halt the ACE sequencing.

Advanced ACE Options
Use External Varef V
Expose FABACETRIG port Initiate ACE sampling from application code
OK Cancel

Figure 8 • Expose FABACETRIG Port Checkbox

2. Create the ACE design as normal. Configure the ACE services and threshold flags.



3. Sequence your analog services in the ACE configurator **Controller** tab.

Procedures ADCO_MAIN ADCO_MAIN ADCO_MAIN	<u>*1 / ×</u>	ADC Block 0	
Operating sequence entry: C Auto			
Details of procedure: ADC0_MAIN Available signals: S	ampling rate		
·> ·>>	ADCDirectInput_0 ADCDirectInput_1 ADCDirectInput_1	Actual Rate (ksps) 0.200 0.200 0.200	
<	(Total samplin		
Calculate Actua		* _ /× + *	
	nstruction	SSE Execution Time (us)	
Sample ADCDirectInput_0 Sample ADCDirectInput_1 Sample ADCDirectInput_2 Stop the sequence in this timeslot		11.660 11.660 11.660 0.000	

Figure 9 • ACE Configurator Controller

You can initiate a signal to the fabric when the end of this sequence is reached. You can accomplish this by reconfiguring the *ADCDirectInput_2* service to assert a signal on completing its post processing. Use this particular service, as it is the last one in this sequence.

4. Configure the *ADCDirectInput_2* service and select the **Assert flag when post processing completed** checkbox.

	Signal name: ADCDirectInput_2	_	Send raw ADC r	and he DAA	•	
	· -		Send raw AUC r	esuic co Dima		
Acquisition time: Digital filtering Filtering factor: Initial value:	10.000 us		Linear transform	nation		
Send filtered	0.000 V		Offset:	rmed result to D	MA	
	Tł	nreshold Detection			<u>*</u> 🗡	
Flag N	lame Flag Type	Threshold H (V)			assert Angles	
					~	

Figure 10 • Configure the ADCDirectInput_2 Service



Selecting this checkbox creates a flag signal named <*Signal name*>:*PPE_DONE* automatically.

5. Go into the **Flags** tab and assign it to the bit position of your choice. In the example, you have assigned the new *PPE_DONE* indication to bit 15.

onfigure ACE	Controller	Flag	s				
lect a register to view ACE	flag mappi <mark>ng:</mark> Avail	able I	bits of PPE_FLAG50 register:				
Fabric	В	it	Source	ACE	Cortex-M3	Lock	
Flag Registers		0	<none></none>	54	118	Γ	
PPE_FLAGS0 (0x4002	21450.)	1	<none></none>	55	119		
PPE_FLAGS1 (0x4002			<none></none>	56	120		
PPE_FLAGS2 (0x4002			<none></none>	57	121		
PPE_FLAGS3 (0x4002			<none></none>	58	122		
PPE_SFFLAGS (0x400			<none></none>	59	123		
			<none></none>	60	124		
			<none></none>	61 62	125		
			(none)	63	126 127		
			<none></none>	64	127		-
			(none)	65	128		
			(none)	66	130	-	-
			(none>	67	131	-	
			(none)	60	132	-	
			ADCDirectInput 2:PPE DONE	69	133	V	
	4	10	(none)	70	134		
			<none></none>	71	135	Г	
			<none></none>	72	136		
			<none></none>	73	137		
			<none></none>	74	138		
			<none></none>	75	139		
			<none></none>	76	140		
			<none></none>	77	141		
			(none)	78	142		
			(none)	79 80	143 144		
			<none></none>	80	144		
			<none></none>	81	145	-	
			<none></none>	83	145	-	
		20 1	(none)	0.4	147	-	~

Figure 11 • Assigning ADCDirectInput_2_PPE_DONE to Bit 15



6. Click *FPGA_FLAGS* in the left-hand treeview and select the **Expose these signals as ACEFLAGs** checkbox.

Configure ACE Controller	Flags	
ielect a register to view ACE flag mapping:	Available bits of FPGA_FLAGS register:	
	✓ Expose these signals as ACEFLAGs	
- Fabric	Bit Source	<u>^</u>
FPGA_FLAGS	0 THRESH_FLAGS 1 THRESH FLAGS	
Flag Registers	2 THRESH FLAGS	
	3 THRESH FLAGS	
	4 THRESH_FLAGS	
PPE_FLAG52 (0x40021458) PPE_FLAG53 (0x4002145c)	5 THRESH FLAGS	
PPE_FLAGS3 (0x40021450)	6 THRESH FLAGS	
PPE_SFFLAGS (0X40021460)	7 THRESH_FLAGS	
	8 THRESH_FLAGS	
	9 THRESH_FLAGS	
	10 THRESH_FLAGS	
	11 THRESH_FLAGS	
	12 THRESH_FLAGS	
	13 THRESH_FLAGS	
	14 THRESH_FLAGS 15 THRESH_FLAGS	: E
	16 THRESH_FLAGS	
	17 THRESH FLAGS	
	18 THRESH FLAGS	
	19 THRESH FLAGS	
	20 THRESH_FLAGS	
	21 THRESH_FLAGS	
	22 THRESH_FLAGS	
	23 THRESH_FLAGS	
	24 THRESH_FLAGS	
	25 THRESH_FLAGS	
	26 THRESH_FLAGS	
	27 THRESH_FLAGS	
	28 THRESH_FLAGS	
	29 THRESH_FLAGS	
	30 THRESH_FLAGS	

Figure 12 • Exposing ACEFLAGs

7. Click **OK**, the ACE instance in the MSS configurator window is displayed, as shown in Figure 13.

ACEFLAGS[31:0]
*

Figure 13 • ACE Instance in the MSS Configurator



Appendix B: Special ACE Register for Design Example

ADDR	Register Name	Description
0x4002122c	PPE_FLAGS0_ IRQ_CLR	These <i>write-only</i> bits are used to clear corresponding bits in the <i>PPE_FLAGS0_IRQ</i> register. Writing a 1 to any of the bits clears the corresponding bits in the <i>PPE_FLAGS0_IRQ</i> register, while writing a 0 to any of the bits does not have an effect. In the event that writing a 1 to these clear bits coincides with a set event in the <i>PPE_FLAGS0_IRQ</i> register, the set event shall have higher priority.
0x40020040	PC0_LO	Program Counter 0 points to address the next sample sequence instruction for ADC0. A write to this register causes the timeslot 0 to unconditionally jump to the SSE RAM address PC[7:0], which includes RAM addresses from 0 to 255 inclusive.
0x40020044	PC0_HI	Program Counter 0 points to address the next sample sequence instruction for ADC0. A write to this register causes timeslot 0 to unconditionally jump to the SSE RAM address 256+PC[7:0], which includes RAM addresses from 256 to 511 inclusive.
0x40020048	PC0_CTRL	Only bit 0 is used. When set to 1, Program Counter 0 is enabled and when set to 0, Program Counter 0 is disabled (stop).

Table 1 • Special ACE Register for Design Example

Note: Table 1 shows the register description for the Program Counter 0. The registers are duplicated for Program Counter 1 and Program Counter 2.



Appendix C: Configuration for Simultaneous Sample Procedure

- 1. In the Configurator ACE window, click the Controller tab.
- 2. From the **Insert Operating Sequence Slot** drop-down list, select SIMULTANEOUS SAMPLE. The Configure 'SIMULTANEOUS SAMPLE' window is displayed.

figure ACE
Configure ACE Controller Flags
Procedures
ADC Block 0 ADC Z_MAIN ADC Block 1 ADC Block 2
Operating sequence entry: C Auto Manual Details of procedure: Simultaneous_sample Available signals: Sampling rate Sampling rate Signal Actual Rate Sec. < <
Calculate Actual Rate Total sampling rate: 0.400 ksps Operating sequence • • • •
Instruction NOP SAMPLE STRULTANEOUS SAMPLE SET INTERRUPT WAIT TIME RESTART SEQUENCE CALIBRATE ADC POWERDOWN ADC STOP ADC
Help OK Cancel

Figure 14 • Adding Simultaneous Sample Procedure

3. Select the services you want to sample simultaneously and click OK.

nfigure 'SIMULTANEOUS SAMPLE'	
Analog Pad on ADCO: VoltageMonitor_0	_
Analog Pad on ADC1: VoltageMonitor_4	•
Analog Pad on ADC2:	•
Description nultaneously Sample Analog Peripherals on multiple ADCs	
	OK Cancel

Figure 15 • Selecting Services for Simultaneous Sample



Appendix D: ACE Configuration and Procedure for Design Example 2

The ACE controller procedures are:

ADC0_MAIN:

• Stop the sequence in this time slot

ADC1_MAIN:

Stop the sequence in this time slot

ADC2_MAIN:

- Stop the sequence in this time slot
- seq_adc00:
 - Sample VoltageMonitor_2
 - Sample VoltageMonitor_3
 - Assert Interrupt on GP1
 - Stop the sequence in this time slot

seq_adc10:

- Sample VoltageMonitor_6
- Sample VoltageMonitor_7
- Assert Interrupt on GP0
- Stop the sequence in this time slot

two ADCs:

- Simultaneously Sample VoltageMonitor_0 VoltageMonitor_4
- Assert Interrupt on GP0
- Simultaneously Sample VoltageMonitor_1 VoltageMonitor_5
- Stop the sequence in this time slot

Appendix E: Design Files

You can download the design files from the Microsemi SoC Products Group website: $\label{eq:http://soc.microsemi.com/download/rsc/?f=A2F_AC366_DF \ .$



List of Changes

Revision*	Changes	Page
Revision 1	Updated Figure 4 (SAR 35795).	4
(January 2012)	Modified the section "Design Example" (SAR 35795).	3
	Updated Figure 7 (SAR 35795).	6
	Modified the section "Appendix E: Design Files" (SAR 35795).	14
Revision 2	Modified the sections:	5 and 6
(May 2019)	Design Example 1: ACE Sequencer Using Fabric Logic	
	Design Example 2: Simultaneous Sampling Using the MSS (SAR 51242).	

The following table lists critical changes that were made in each revision of the document.

Note: *The revision number is located in the part number after the hyphen. The part number is displayed at the bottom of the last page of the document. The digits following the slash indicate the month and year of publication.



а 🥂 Міскоснір company

Microsemi Headquarters One Enterprise, Aliso Viejo, CA 92656 USA Within the USA: +1 (800) 713-4113 Outside the USA: +1 (949) 380-6100 Sales: +1 (949) 380-6136

Fax: +1 (949) 215-4996 Email: sales.support@microsemi.com www.microsemi.com

©2019 Microsemi, a wholly owned subsidiary of Microchip Technology Inc. All rights reserved. Microsemi and the Microsemi logo are registered trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners. Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document or to any products and services at any time without notice.

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.