W65C265i1M16SA Microcontroller Datasheet

WDC reserves the right to make changes at any time without notice in order to improve design and supply the best possible product. Information contained herein is provided gratuitously and without liability, to any user. Reasonable efforts have been made to verify the accuracy of the information but no guarantee whatsoever is given as to the accuracy or as to its applicability to particular uses. In every instance, it must be the responsibility of the user to determine the suitability of the products for each application. WDC products are not authorized for use as critical components in life support devices or systems. Nothing contained herein shall be construed as a recommendation to use any product in violation of existing patents or other rights of third parties. The sale of any WDC product is subject to all WDC Terms and Conditions of Sales and Sales Policies, copies of which are available upon request.

This page intentionally left blank.

Table of Contents

DOCUMENT REVISION HISTORY

Version	Date	Authors	Description
1.0	05/13/2019	Bill Mensch, David Gray	Initial Document Entry
1.0	07/01/2019	Bill Mensch, David Gray	Fixed minor typos and added UFM with EKP
1.0	07/10/2019	David Gray	Updated MAX10 Device Resource Table

1 INTRODUCTION

The W65C265i1M16SA Microcontroller Datasheet is for MyMENSCH™ Rev-B. MyMENSCH™ Rev-B and -C replaces the five JTAG pins on connector J4 on Rev-A with AGND, dedicated ADC input ADC_IN0, 3v3REF, 5vBAT, and VIA_B_PB6. In addition to the connector J4 changes Rev-C with replace the GPIO driven FT240x on Rev-B parallel-to-USB chip with a CH340C ACIA_A UART-to-USB chip. Rev-A used the FT245 parallel-to-USB chip.

The microprocessor unit (MPU) is the W65C816RTL microprocessor. The W65Cx65 Microcontrollers have interfaces for connected Things for sensing, processing, communicating and actuating (SPCA). The W65Cx65 Microcontrollers are described with the Verilog HDL for use with both FPGAs and ASIC design and manufacturing flow.

WDC creates application specific versions of the W65Cx65 for variants of the MAX10 technology. Refer to Section 1.2 below for MAX10 169 BGA versions available from Intel that could be evaluated with custom versions of MyMENSCH™. The MAX10M16SA is first available on MyMENSCH™ Rev-B evaluation and developer board.

The W65Cx65 Microcontrollers have a CFM MyMENSCH Monitor for boot loading and debugging user code with WDCTools Assembly and C language support, Hardware Breakpoint Module (HBM), multipliers, divider, Unique Chip ID, ACIAs/UARTs, VIAs, User GPIO pins, SPI Master, I2C Master, initialized and memory protected RAM for User code from the USB developer port or User FLASH Memory (UFM), and varying amounts of data RAM.

OEMs now have plenty of flexible choices for customization of application specific features, contract manufacturing, lowering cost, improving ease of use, future proofing, lowering power/energy, and enabling higher levels of security for use with Cloud services connected through the Internet of Things (IoT).

This product description assumes that the reader is familiar with the W65C816S CPU hardware and programming capabilities. Refer to documentation on the WDC65xx.com website, ***Programming the 65816 Including the 6502, 65C02 and 65802*** Manual, WDC Datasheets for the W65C02S MPU, W65C816S MPU, W65C22 VIA, W65C51 ACIA/UART, Industry documentation for SPI, I2C, GPIO, JTAG, Intel PSG MAX10 FPGAs, Quartus and Mentor Modelsim design tools for more information.

1.1 Key Features of the W65C265i1M16SA Microcontroller

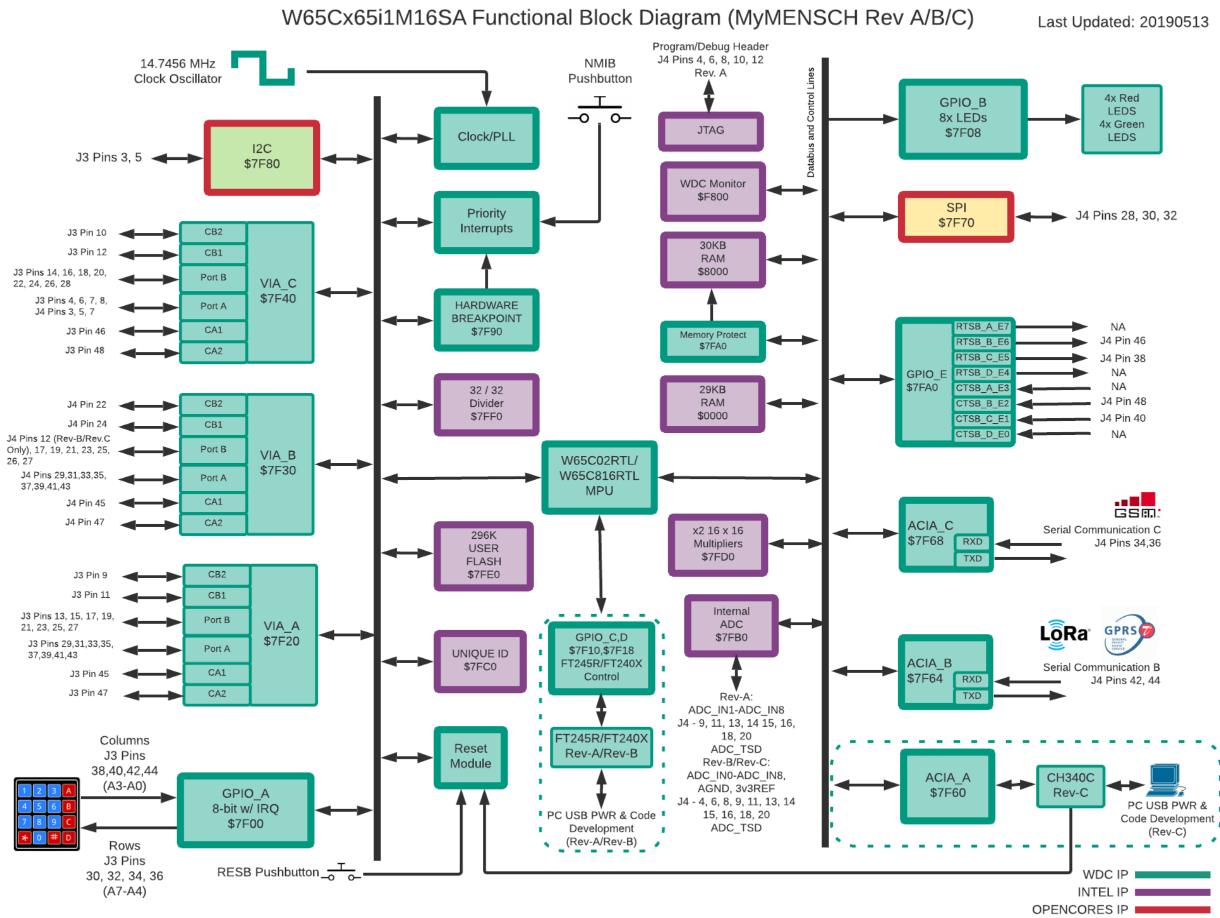
- Intel PSG MAX10M16 SA FPGA
- ~16,000 Logic Elements Available
- Operating Voltage – 3.3V
- System Operation Speed – 14.7456 MHz
- W65C816RTL MPU
- W65C22RTL VIA (x3)
- W65C51RTL ACIA (x2) on Rev-A and -B (x3) on Rev-C
- ACIA XTLI Operation Speed – 1.8432 MHz
- W65CGPIO 5 register and 2 register
- De-bounced Keypad GPIO_A
- W65CHBM Hardware Breakpoint Module
- SPI Master
- I2C Master
- WDC 2K byte for 2048 bytes of CFM MyMENSCH™ Monitor for boot loading and debugging code
- 2K bytes 2048 bytes reserved for ADC
- 30K bytes for a total of 30,720 bytes for User code SRAM boot loaded from USB or copied from UFM
- 29K bytes for a total of 29,696 bytes for data SRAM
- JTAG available on MyMENSCH™ Rev-A on J4
- 16x16 Hardware multiplier (x2 – Signed and Unsigned)
- 32/32-bit Hardware Divider
- 12-bit ADC (1x), 1 dedicated pin, 8 dual function (digital IO or ADC) pins, Temperature Diode
- 296K bytes for 303,104 bytes of User FLASH Memory (UFM)
- 64-bit Unique Chip ID/serial number programmed in the Intel MAX10 factory
- 18,446,744,073,709,551,616 Unique IDs

1.2 Intel MAX 10 Device Maximum Resources

Maximum Resource Counts for Intel MAX 10 Devices			
Resource Device	**M08SC**	**M08SA**	**M16SA**
Logic Elements (LE) (K)	8	8	16
M9K Memory Blocks (1 KB)	42	42	61
User Flash Memory Max (KB) with EKP	32	114	184
User Flash Memory Max (KB) without EKP	90	172	296
18 × 18 Multiplier	24	24	45
PLL (Max)	1	1	1
169 BGA Package	Yes	Yes	Yes
GPIO (Max IO Count 169 BGA Package)	130	130	130
Internal Configuration Image	1	1	1
ADC	0	1	1

1.3 Functional Block Diagram

The following block diagram is for the W65C265i1M16SA.



1.4 Controller Function Description

The W65C816RTL Core is the central processor for all function control. The 2048 bytes of CFM RAM contains a Monitor for boot loading and debugging code. There is 61K bytes for a total of 62,464 bytes of SRAM available for application code, ADC and data.

2 MODULE DESCRIPTIONS

Following are descriptions of the basic modules.

2.1 CLOCK MODULE

There are 4 clocks used in this design one for PHI2 for the MPU and system timing; one for the ADC; one for the embedded memory blocks; and one for the XTLI on the ACIAs. All clocks are derived from the 14.7456 MHz Oscillator on MyMENSCH™.

2.2 RESET MODULE

There are no Reset Module Registers and therefore no definitions. This is a basic module to handle the reset logic for the system.

2.3 W65C816RTL Programming Model

Refer to the W65C816S Datasheet for the Microprocessor Programming Model, Status Register Coding and complete information. More information is found in ***Programming the 65816: Including the 6502, 65C02 and 65802*** Manual available through Amazon.

2.4 Priority Interrupt Controller Module Information

The Interrupt Control Module controls the priority and memory map for interrupts. Each interrupt is connected to the Interrupt Control Module for prioritizing.

Interrupt Enable Registers for the various interrupts are the interrupt enable by the various enable bits. Reading the various IER and IFR bits determines the interrupt that occurred. By prioritizing the interrupts one can determine which interrupt occurred in the associated interrupt handler routine. Notice that any of the 8 interrupts for a GPIO 8-bit port will cause a GPIO vectored interrupt to occur.

2.4.1 Priority Encoded Interrupt Vector Module

Vector Address	Label	Function
0xFFFE,F	IRQBRK	BRK – Software Interrupt
0xFFFC,D	IRQRES	RES – "REStart" Interrupt
0xFFFA,B	IRQNMI	Non-Maskable Interrupt/Hardware Breakpoint (HBP)
0xFFF8,9	IRQGPIO_E	GPIO_E Interrupt for all Eight Input Edge interrupts
0xFFF6,7	IRQGPIO_A	GPIO_A Interrupt for all Eight Input Edge interrupts
0xFFF4,5	IRQVIA_A	VIA_A Interrupt
0xFFF2,3	IRQVIA_B	VIA_B Interrupt
0xFFF0,1	IRQVIA_C	VIA_C Interrupt
0xFFEE,F	IRQVIA_D	Reserved
0xFFEC,D	IRQSPI	SPI Interrupt
0xFFEA,B	IRQI2C	I2C Interrupt
0xFFE8,9	IRQADC	ADC Interrupt
0xFFE6,7	IRQACIA_A	ACIA_A Interrupt (not available on Rev-A and -B)
0xFFE4,5	IRQACIA_B	ACIA_B Interrupt
0xFFE2,3	IRQACIA_C	ACIA_C Interrupt
0xFFE0,1	IRQACIA_D	Reserved

2.5 Memory Map

Start	End	Size	Description
		2048 B	2048 Byte Block RAM Reserved for ADC
0xF800	0xFFFF	2048 B	2048 Byte CFM Monitor
0x8000	0xF7FF	30720 B	30K Byte Protected RAM (Loadable from UFM)
0x7FF0	0x7FFF	16 B	32/32-bit Divider
0x7FE0	0x7FFC	12 B	User FLASH
0x7FD0	0x7FDF	16 B	16x16 Multipliers
0x7FC0	0x7FC8	9 B	Unique Chip ID
0x7FB0	0x7FB5	6 B	ADC
0x7FA8		1 B	Memory Protect
0x7FA0	0x7FA4	5 B	GPIOE
0x7F90	0x7F9F	16 B	HBM
0x7F88	0x7F8F	8 B	Reserved
0x7F80	0x7F85	6 B	I2C
0x7F78	0x7F7F	8 B	Reserved
0x7F70	0x7F75	6 B	SPI
0x7F6C	0x7F6F	4 B	Reserved
0x7F68	0x7F6B	4 B	ACIA_C
0x7F64	0x7F67	4 B	ACIA_B
0x7F60	0x7F63	4 B	ACIA_A (not available on Rev-A and –B)
0x7F50	0x7F5F	16 B	Reserved
0x7F40	0x7F4F	16 B	VIA_C
0x7F30	0x7F3F	16 B	VIA_B
0x7F20	0x7F2F	16 B	VIA_A
0x7F18	0x7F19	2 B	GPIO_D for FTDI245 (not available on Rev-C)
0x7F10	0x7F11	2 B	GPIO_C for FTDI245 (not available on Rev-C)
0x7F08	0x7F09	2 B	GPIO_B 8 LEDs
0x7F00	0x7F04	5 B	GPIO_A
0x0000	0x73FF	29696 B	29K Byte SRAM

2.6 VIA Port Module

The W65C265i1M16SA features three Versatile Interface Adapters (VIA) based on the W65C22S. See Memory Map for base addresses. See W65C22S Datasheet for full register descriptions.

2.7 GPIO Port Modules

GPIO Port Modules are included on this design. There are two different GPIO IP blocks used depending on the need for Interrupt Logic. The 2 Register (8-bit) version has only a PIO Register (PIOx) and Data Direction Register (DDRx). GPIO B, C, and D use the 2 Register modules. The 5 Register (8-bit) version supports edge sense interrupts and has a PIO Register (PIOx), Data Direction Register (DDRx), Interrupt Flag Register (IFRx), Interrupt Enable Register (IERx), and Edge Sense Register (ESRx). GPIOA and GPIOE use the 5 Register version. Note that GPIOA has a special GPIO function for the W65C265i1. GPIOA is used for a 4x4 keypad interface. Bits 4-7 are intended to be inputs and have pullup resistors implemented within the FPGA. GPIOE was intended to be used as handshake logic for the 3 ACIA modules. GPIO E0 and E4 can be used as GPIO. See Memory Map for base addresses.

2.7.1 GPIO Module Registers Description - 5 Register Version

Address = Base + 4	GPIO_ESR: GPIO Edge Sense Register					Reset Value = 0x00	
7:0->	ESR7	ESR6	ESR5	ESR4	ESR3	ESR2	ESR1	ESR0
Bit	Name	Access	Description					
7 – 0	ESR[7:0]	R/W	1 = Positive Edge Sense for PIO7-0					
			0 = Negative Edge Sense for PIO7-0					

Address = Base + 3	GPIO_IER: GPIO Interrupt Enable Register					Reset Value = 0x00	
7:0->	IER7	IER6	IER5	IER4	IER3	IER2	IER1	IER0
Bit	Name	Access	Description					
7 – 0	IER[7:0]	R/W	1 = Enable Interrupt on inputs for PIO7-0					
			0 = Disable Interrupts on inputs for PIO7-0					

Address = Base + 2	GPIO_IFR: GPIO Interrupt Flag Register					Reset Value = 0x00	
7:0->	IFR7	IFR6	IFR5	IFR4	IFR3	IFR2	IFR1	IFR0
Bit	Name	Access	Description					
7 – 0	IFR[7:0]	R/W	1 = Interrupt Occurred on inputs for PIO7-0					
			0 = Interrupts did not occur on inputs for PIO7-0					

Address = Base + 1	GPIO_DDR: GPIO Data Direction Register					Reset Value = 0x00	
7:0->	DDR7	DDR6	DDR5	DDR4	DDR3	DDR2	DDR1	DDR0
Bit	Name	Access	Description					
7 – 0	DDR[7:0]	R/W	1 = PIO data direction set to Output PIO7-0					
			0 = PIO data direction set to Input PIO7-0					

Address = Base	GPIO_DATA: GPIO Data Register					Reset Value = 0x00	
7:0->	PIO7	PIO6	PIO5	PIO4	PIO3	PIO2	PIO1	PIO0
Bit	Name	Access	Description					
7 – 0	PIO[7:0]	R/W	1 = PIO line is logic 1 value read and sets a 1 value on write for PIO7-0					
			0 = PIO line is logic 0 value read and sets a 0 value on write for PIO7-0					

2.7.2 GPIO Module Registers Description - 2 Register Version

Address = Base + 1	GPIO_DDR: GPIO Data Direction Register					Reset Value = 0x00	
7:0->	DDR7	DDR6	DDR5	DDR4	DDR3	DDR2	DDR1	DDR0
Bit	Name	Access	Description				
7 – 0	DDR[7:0]	R/W	1 = PIO data direction set to Output PIO7-0				
			0 = PIO data direction set to Input PIO7-0				

Address = Base	GPIO_DATA: GPIO Data Register					Reset Value = 0x00		
7:0->	PIO7	PIO6	PIO5	PIO4	PIO3	PIO2	PIO1	PIO0
Bit	Name	Access	Description					
7 – 0	PIO[7:0]	R/W	1 = PIO line is logic 1 value read and sets a 1 value on write for PIO7-0					
			0 = PIO line is logic 0 value read and sets a 0 value on write for PIO7-0					

2.8 Hardware Breakpoint Module (HBM)

The Hardware Breakpoint Module pulls NMIB low during a match condition. HBM addresses 00, 01 are for the low and high byte of 16-bit address bus matching registers. HBM addresses 02, 03 are RESERVED for future 32 bit address bus matching registers. HBM address 04 is the data bus matching register. HBM addresses 05, 06, 07 are RESERVED for future 32-bit data bus matching registers. HBM addresses 08-0E are RESERVED for future use. HBM address 0F is the HBM Control Register. The monitor needs to write a "0" into the Control Register after a breakpoint has been read to clear it. Writing a "1" to Bit 7 will cause a manual NMI if the breakpoint is enabled. See Memory Map for Base address.

2.8.1 Hardware Breakpoint Match (HBM) Registers Description

Address = Base + F	HBM_ICDCTRL: Hardware Breakpoint Control						Reset Value =0x00	
7:0->	BRK	0	0	0	MATCH	DATAEN	RWSEL	BRKEN
Bit	**Name**	**Access**	**Description**					
7	BRK	R/W	1 = Hardware Break occurred					
			0 = No Hardware Break occurred					
3	MATCH	R/W	1 = Data breakpoint if DATAREG value matches bus value					
			0 = Data breakpoint if DATAREG value doesn't match bus value					
2	DATAEN	R/W	1 = Enable breakpoint on data bus and DATAREG match (or mismatch as selected by bit 3)					
			0 = Disable breakpoint on Data					
1	RWSEL	R/W	1 = Data breakpoint on Read data (in to MPU)					
			0 = Data breakpoint on Write data (out from MPU)					
0	BRKEN	R/W	1 = Enable breakpoint on match with Address in BRKREG register					
			0 = Disable Address breakpoint					
Address = Base + 4	**HBM_DATAREG: Hardware Data Match**							
7:0->	DVAL7	DVAL6	DVAL5	DVAL4	DVAL3	DVAL2	DVAL1	DVAL0
Bit	**Name**	**Access**	**Description**					
7 - 0	DVAL[7:0]	R/W	Value of Data bus to match or mismatch with (as selected by ICDCTRL register)					
			bits 7-0 correspond to MPU data bus signals 7-0 for matching or not-matching					
Address = Base + 1	**HBM_BRKREG_H: Hardware Breakpoint Address (High Byte)**						**Reset Value =0x00**	
7:0->	BADR15	BADR14	BADR13	BADR12	BADR11	BADR10	BADR9	BADR8
Bit	**Name**	**Access**	**Description**					
7 - 0	BADR[15:8]	R/W	Value of Address bus to match with					
			bits 15-0 correspond to MPU address bus signals 15-0 for matching					
Address = Base	**HBM_BRKREG_L: Hardware Breakpoint Address (Low Byte)**						**Reset Value =0x00**	
7:0->	BADR7	BADR6	BADR5	BADR4	BADR3	BADR2	BADR1	BADR0
Bit	**Name**	**Access**	**Description**					
7 – 0	BADR[7:0]	R/W	Value of Address bus to match with					
			bits 15-0 correspond to MPU address bus signals 15-0 for matching					

2.9 ACIA Modules

The W65C265i1M16SA has two Asynchronous Communications Interface Adapter (ACIA) modules on MyMENSCH™ Rev-A and -B and three available on Rev-C used to transfer information to and from various communications modules such as LoRa, GSM, Bluetooth, Wi-Fi radio modules and UART enabled devices. See the Memory Map for base addresses. The baud rates are derived from 1.8432MHz XTLI input. Rev-C has a third ACIA for use with the PC USB developer port.

2.9.1 ACIA Control Register Description

Address = Base + 3	ACIA_CTRL: ACIA Control Register						Reset Value = 0x00	
7:0->	SBN	WL1	WL0	RSC	SBR3	SBR2	SBR1	SBR0
HWRES	0	0	0	1	0	0	0	0
SWRES	-	-	-	1	-	-	-	-

Bit	Name	Access	Description
7	SBN	R/W	1 = 2 Stop bits, 1 ½ Stop bits for WL = 5, 1 Stop bit for WL = 8 and parity
			0 = 1 Stop bit
6	WL1	R/W	11 = 5 bits
			10 = 6 bits
5	WL0	R/W	01 = 7 bits
			00 = 8 bits
4	RSC	R/W	1 = Baud rate
			0 = RSC clock source
3	SBR3	R/W	1110 = 9600, 1111 = 19200
			1100 = 4800, 1101 = 7200
2	SBR2	R/W	1010 = 2400, 1011 = 3600
			1000 = 1200, 1001 = 1800
1	SBR1	R/W	0110 = 300, 0111 = 600
			0100 = 134.58, 0101 = 150
0	SBR0	R/W	0010 = 75, 0011 = 109.92
			0000 = 115.2K, 0001 = 50

2.9.2 ACIA Command Register Description

Address = Base + 2	ACIA_CMR: ACIA Command Register						Reset Value = 0x00	
7:0->	PCM1	PCM0	PME	REM	TIC1	TIC0	IRD	DTR
HWRES	0	0	0	0	0	0	0	0
SWRES	-	-	-	0	0	0	0	0

Bit	Name	Access	Description
7	**PCM1**	R/W	11 = Space parity
			10 = Mark parity
6	**PCM0**	R/W	01 = Odd parity
			00 = Even parity
5	**PME**	R/W	1 = Parity enabled
			0 = Parity disabled
4	**REM**	R/W	1 = Receiver Echo Mode not available
			0 = Receiver Echo Mode not available
3	**TIC1**	R/W	11 = RTSB = low, Transmitter interrupt disabled, Transmit Break
			10 = RTSB = low, Transmitter interrupt disabled
2	**TIC0**	R/W	01 = RTSB = low, Transmitter interrupt enabled
			00 = RTSB = high, Transmitter interrupt disabled
1	**IRD**	R/W	1 = Receiver Interrupt Disabled
			0 = Receiver Interrupt Enabled
0	**DTR**	R/W	1 = Data Terminal Ready
			0 = Data Terminal Transmitter Not Ready

2.9.3 ACIA Status Register Description

Address = Base + 1			ACIA_STR: ACIA Status Register				Reset Value = 0x10	
7:0->	IRQ	DSRB	DCDB	TDRE	RDRF	OVRN	FE	PE
HWRES	0	0	0	1	0	0	0	0
SWRES	-	-	-	1	-	-	-	-

Bit	Name	Access	Description
7	IRQ	R/O	1 = Interrupt has occurred
			0 = No Interrupt
6	DSRB	R/O	1 = Not ready and not clear to send data
			0 = Ready and clear to send data
5	DCDB	R/O	1 = DCD Not Detected
			0 = DCD Detected
4	TDRE	R/O	1 = Empty
			0 = Not Empty
3	RDRF	R/O	1 = Full
			0 = Not Full
2	OVRN	R/O	1 = Overrun has occurred
			0 = No overrun
1	FE	R/O	1 = Framing error detected
			0 = No framing error
0	PE	R/O	1 = Parity error detected
			0 = No parity error
Address = Base + 1		W/O	Program Reset aka SWRES

2.9.4 ACIA Data Register Description

Address = Base		ACIA_DR: ACIA Data Register					Reset Value = 0x00	
7:0->	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

Bit	Name	Access	Description
7 - 0	DR[7-0]	R/W	R = Read Receiver Data Register
			W = Write Transmitter Data Register

2.10 I2C Interface Module

For the I2C register descriptions and I2C Operation, refer to the “I2C Design Specification”.

2.10.1 I2C Status Register Description

Address = Base + 5		SR: I2C Status Register					Reset Value = 0x00	
7:0->	RxACK	WCOL	BUSY	RESERVED	WFFULL	WFEMPTY	RFFULL	RFEMPTY

Bit	Name	Access	Description
7	RxACK	R/O	1 = No acknowledge received
			0 = Acknowledge received
6	BUSY	R/W	1 = After start I2C bus busy signal detected
			0 = After stop I2C bus busy signal detected
5	RSRVD	R/O	1 = Never
			0 = Always
4	RSRVD	R/O	1 = Never
			0 = Always
3	RSRVD	R/O	1 = Never
			0 = Always
2	RSRVD	R/O	1 = Never
			0 = Always
1	TIP	R/O	1 = Transfer in progress when transferring data
			0 = When transfer complete
0	IF	R/O	1 = Interrupt is set when one byte is transferred, processor interrupt request if IEN bit is set.
			0 = No interrupt

2.10.2 I2C Command Register Description

Address = Base + 4	CR: I2C Command Register						Reset Value = 0x00	
7:0->	STA	STO	RD	WR	ACK	RSVRD	RSVRD	IACK
Bit	Name	Access	Description					
7	STA	R/W	1 = Generate start condition 0 = Do not generate start condition					
6	STO	R/W	1 = Generate stop condition 0 = Do not generate stop condition					
5	RD	R/W	1 = Read from slaver 0 = Do not read from slave					
4	WR	R/W	1 = Write slave 0 = Do not write slave					
3	ACK	R/W	1 = NACK 0 = ACK					
2	RSVRD	R/W	1 = Never 0 = Always					
1	RSVRD	R/W	1 = Never 0 = Always					
0	IACK	R/W	1 = Clear a pending interrupt 0 = Don't clear a pending interrupt					

2.10.3 I2C Receive Register Description

Last byte received via I2C.

Address = Base + 3	RXR: I2C Receive Register						Reset Value = 0x00	
7:0->	RXR7	RXR6	RXR5	RXR4	RXR3	RXR2	RXR1	RXR0
Bit	Name	Access	Description					
7 - 0	RXR[7-0]	R/O	**R = Read Receiver Data Register** W = no operation					

2.10.4 I2C Transmit Register Description

7:1 RW Next byte to transmit via I2C 0 RW In case of a data transfer this bit represent the data's LSB. In case of a slave address transfer this bit represents the RW bit. '1' for reading from slave '0' for writing to slave.

Address = Base + 2		TXR: I2C Transmit Register					Reset Value = 0x00	
7:0->	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
Bit	**Name**	**Access**	**Description**					
7 - 0	TXR[7-0]	R/W	R = Read Receiver Data Register					
			W = Write Transmitter Data Register					

2.10.5 I2C Control Register Description

The core responds to new commands only when the 'EN' bit is set. Pending commands are finished. Clear the 'EN' bit only when no transfer is in progress, i.e. after a STOP command, or when the command register has the STO bit set. When halted during a transfer, the core can hang the I2C bus.

Address = Base +1		CTR: I2C Control Register					Reset Value = 0x10	
7:0->	EN	IEN	RSRVD	RSRVD	RSRVD	RSRVD	RSRVD	RSRVD
Bit	**Name**	**Access**	**Description**					
7	EN	R/W	1 = I2C Core enabled					
			0 = I2C Core disabled					
6	IEN	R/W	1 = I2C Core interrupt enabled					
			0 = I2C Core interrupt disabled					
5	RSRVD	R/W	1 = Never					
			0 = Always					
4	RSRVD	R/W	1 = Never					
			0 = Always					
3	RSRVD	R/W	1 = Never					
			0 = Always					
2	RSRVD	R/W	1 = Never					
			0 = Always					
1	RSRVD	R/W	1 = Never					
			0 = Always					
0	RSRVD	R/W	1 = Never					
			0 = Always					

2.10.6 I2C Clock Prescale Register Description

For the I2C register descriptions, refer to the “I2C Design Specification”.

This register is used to prescale the SCL clock line. Due to the structure of the I2C interface, the core uses a 4*SCL clock internally. The prescale register must be programmed to this 4*SCL bitrate. Change the value of the prescale register only when the ‘EN’ bit is cleared.
Example: CLK_I = 32MHz, desired SCL = 100 KHz
Prescale = 32MHZ = 80 (dec) = 50 (hex) 4 * 100 KHz
Reset value: 0xFFFF

Address = Base		PRER: I2C Clock Prescale Register					Reset Value = 0xFF	
7:0->	PRER7	PRER6	PRER5	PRER4	PRER3	PRER2	PRER1	PRER0
Bit	**Name**	**Access**	**Description**					
7 - 0	PRER[7-0]	R/W	R = Read Receiver Data Register					
			W = Write Transmitter Data Register					

2.11 SPI Module

The SPI module described in the standard SPI Specification found in this link.

2.11.1 SPI Extension Register Description

Address = Base + 3	SPER: SPI Extension Register						Reset Value = 0x00	
7:0->	ICNT1	ICNT0	RESERVED	RESERVED	RESERVED	RESERVED	ESPR1	ESPR0
Bit	**Name**	**Access**	**Description**					
7	ICNT1	R/W	11 = SPIF is set after every four completed transfers					
			10 = SPIF is set after every three completed transfers					
6	ICNT0	R/W	01 = SPIF is set after every two completed transfers					
			00 = SPIF is set after every completed transfer					
5	RESERVED	R/W	1 = Never					
			0 = Always					
4	RESERVED	R/W	1 = Never					
			0 = Always					
3	RESERVED	R/W	1 = Never					
			0 = Always					
2	RESERVED	R/W	1 = Never					
			0 = Always					
1	ESPR1	R/W	11 = Reserved, do not use					
			10 = Add these two bits to the SPI Clock Rate 0=512, 01=1024, 02=2048, 03=4096					
0	ESPR0	R/W	01 = Add these two bits to the SPI Clock Rate 0=8, 01=64, 02=128, 03=256					
			00 = Add these two bits to the SPI Clock Rate 00=2, 01=4, 02=16, 03=32					

2.11.2 SPI Data Register Description

Address = Base + 2	SPDR: SPI Data Register					Not Initialized on Reset	
7:0->	SPDR7	SPDR6	SPDR5	SPDR4	SPDR3	SPDR2	SPDR1	SPDR0
Bit	**Name**	**Access**	**Description**					
7 - 0	SPDR[7-0]	R/W	**R = Read SPI Data buffer**					
			W = Write SPI Data buffer					

2.11.3 SPI Status Register Description

Address = Base + 1	SPSR: SPI Status Register						Reset Value = 0x05	
7:0->	SPIF	WCOL	RESERVED	RESERVED	WFFULL	WFEMPTY	RFFULL	RFEMPTY

Bit	Name	Access	Description
7	SPIF	R/W	1 = SPI Interrupt Flag is set on completion of a transfer block
			0 = SPI not interrupting
6	WCOL	R/W	1 = SPI Core write collision when SPI data register when Write FIFO is full
			0 = SPI Core disabled
5	RESERVED	R/O	1 = Never
			0 = Always
4	RESERVED	R/O	1 = Never
			0 = Always
3	WFFULL	R/O	1 = Write FIFO full
			0 = Write FIFO not full
2	WFEMPTY	R/O	1 = Write FIFO empty
			0 = Write FIFO not empty
1	RFFULL	R/O	1 = Read FIFO full
			0 = Read FIFO not full
0	RFEMPTY	R/O	1 = Read FIFO empty
			0 = Read FIFO not empty

2.11.4 SPI Control Register Description

Address = Base		SPCR: SPI Control Register					Reset Value = 0x10	
7:0->	SPIE	SPE	RESERVED	MSTR	CPOL	CPHA	SPR1	SPR0

Bit	Name	Access	Description
7	SPIE	R/W	1 = SPI Interrupt Enabled 0 = SPI Interrupt Disabled
6	SPE	R/W	1 = SPI Core enabled 0 = SPI Core disabled
5	RESERVED	R/W	
4	MSTR	R/W	1 = Master 0 = Slave
3	CPOL	R/W	1 = Negative Clock Polarity 0 = Positive Clock Polarity
2	CPHA	R/W	1 = Clock Phase Not Shifted 0 = Clock Phase Shifted
1	SPR1	R/W	These values are used with the ESPR bits to determine the extended clock rate. Refer to the SPI Datasheet for detailed selection information.
0	SPR0	R/W	These values are used with the ESPR bits to determine the extended clock rate. Refer to the SPI Datasheet for detailed selection information.

2.12 Multipliers

The MAX10M16 has embedded multiplier cascading blocks. WDC has created a 16x16 multiplier with 32-bit result module to perform signed and unsigned multiplication of 16-bit numbers. There is one dedicated 16x16 bit multiplier for unsigned values and one for signed. Once the multiplier and multiplicand values are written to their respective registers, the product registers are immediately available to read. All registers have a reset value of 0x00.

2.12.1 Unsigned Multiplication Equation and Register Description

Unsigned Multiplication:
Unsigned Data A (16-bits) x Unsigned Data B (16-bits) = Result C (32-bits)

Unsigned Registers ($7FD0-$7FD7):

Register	Address Offset	Description
UDATAAL	0	Unsigned Data A Lo Register(Bits 7-0)
UDATAAH	1	Unsigned Data A Hi Register (Bits 15-8)
UDATABL	2	Unsigned Data B Lo Register(Bits 7-0)
UDATABH	3	Unsigned Data B Hi Register (Bits 15-8)
URESULTLL	4	Unsigned Result C Lo-Lo Register (Bits 7-0)
URESULTLH	5	Unsigned Result C Lo-Hi Register (Bits 15-8)
URESULTHL	6	Unsigned Result C Hi-Lo Register (Bits 23-16)
URESULTHH	7	Unsigned Result C Hi-Hi Register (Bits 31-24)

2.12.2 Signed Multiplication Equation and Register Description

Signed Multiplication:
Signed Data A (16-bits) x Signed Data B (16-bits) = Result C (32-bits)

Signed Registers ($7FD8-$7FDF):

Register	Address Offset	Description
SDATAAL	8	Signed Data A Lo Register(Bits 7-0)
SDATAAH	9	Signed Data A Hi Register (Bits 15-8)
SDATABL	A	Signed Data B Lo Register(Bits 7-0)
SDATABH	B	Signed Data B Hi Register (Bits 15-8)
SRESULTLL	C	Signed Result C Lo-Lo Register (Bits 7-0)
SRESULTLH	D	Signed Result C Lo-Hi Register (Bits 15-8)
SRESULTHL	E	Signed Result C Hi-Lo Register (Bits 23-16)
SRESULTHH	F	Signed Result C Hi-Hi Register (Bits 31-24)

2.12.3 Multiplication Code Example

```
; Multiplies two UNSIGNED 16 bit values and returns a 32-bit UNSIGNED result
; Use MUL_BASE equ $7FD8 for SIGNED multiplication

      MUL_BASE:         equ   $7FD0       ; base address of Mulitiplier
      MUL_DATAALO:      equ   MUL_BASE+0
      MUL_DATAAHI:      equ   MUL_BASE+1
      MUL_DATABLO:      equ   MUL_BASE+3
      MUL_DATABHI:      equ   MUL_BASE+2
      MUL_RESULTLOLO:   equ   MUL_BASE+3
      MUL_RESULTLOHI:   equ   MUL_BASE+5
      MUL_RESULTHILO:   equ   MUL_BASE+6
      MUL_RESULTHIHI:   equ   MUL_BASE+7

MULTIPLIER:                   ;#$01FF x #$01FF
      lda #$01
      sta MUL_DATAAHI
      sta MUL_DATABHI
      lda #$FF
      sta MUL_DATAALO
      sta MUL_DATABLO
      lda MUL_RESULTHIHI
      jsr LCD_Write_Hex       ;Routine to print hex values to LCD
      lda MUL_RESULTHILO
      jsr LCD_Write_Hex
      lda MUL_RESULTLOHI
      jsr LCD_Write_Hex
      lda MUL_RESULTLOLO
      jsr LCD_Write_Hex
HOME:
      bra HOME
```

2.13 Hardware Divider

Intel provides Hardware Divide IP for their MAX10 FPGA families. WDC's module provides 32/32 Divide with 32-bit result and 32-bit remainder for hardware divide function of 32-bit numbers for unsigned values. Once the dividend and divisor are written to their respective registers, the quotient and remainder is immediately available to read. All registers have a reset value of 0x00.

2.13.1 Unsigned Divide Equation and Register Description

Unsigned Division:

Numerator (32-bits) / Denominator (32-bits) = Quotient (32-bits) with Remainder (32-bits)

Unsigned Registers ($7FF0-$7FFF):

Register	Address Offset	Description
NUMLOLO	0	Unsigned Numerator LoLo Register (Bits 7-0)
NUMLOHI	1	Unsigned Numerator LoHi Register (Bits 15-8)
NUMHILO	2	Unsigned Numerator HiLo Register (Bits 23-16)
NUMHIHI	3	Unsigned Numerator HiHi Register (Bits 31-24)
DENLOLO	4	Unsigned Denominator LoLo Register (Bits 7-0)
DENLOHI	5	Unsigned Denominator LoHi Register (Bits 15-8)
DENHILO	6	Unsigned Denominator HiLo Register (Bits 23-16)
DENHIHI	7	Unsigned Denominator HiHi Register (Bits 31-24)
QUOTLOLO	8	Unsigned Quotient LoLo Register (Bits 7-0)
QUOTLOHI	9	Unsigned Quotient LoHi Register (Bits 15-8)
QUOTHILO	A	Unsigned Quotient HiLo Register (Bits 23-16)
QUOTHIHI	B	Unsigned Quotient HiHi Register (Bits 31-24)
REMLOLO	C	Unsigned Remainder LoLo Register (Bits 7-0)
REMLOHI	D	Unsigned Remainder LoHi Register (Bits 15-8)
REMHILO	E	Unsigned Remainder HiLo Register (Bits 23-16)
REMHIHI	F	Unsigned Remainder HiHi Register (Bits 31-24)

2.14 Analog to Digital Converter Module

The MAX10M16SA has one built in 12-bit resolution Analog-to-Digital Converter (ADC) with one dedicated ADC input ADC_IN0 available on MyMENSCH™ Rev-C but not available on Rev-A, 8 selectable digital IO or analog input pins, plus a temperature sense diode for a total possible ten (10) ADC registered value channels. The ADC has selectable sampling rates.

2.14.1 ADC Module Register Overview

Register	Address Offset	Description
SAMPLEDATA0	0	Sample Store Data Bits 7-0
SAMPLEDATA1	1	Sample Store Data Bits 15-8
SAMPLEADDR	2	Sample Store Address 6-0 (Only 7 bits used)
SEQDATA0	3	Sequencer Data Bits 7-0
CTRL	4	Control Register Bits 7-0

2.14.2 ADC Control Register

Address = Base + 4		CTRL: Control Register					Reset Value = 0x00	
7:0->	IRQ	IRQ_EN	SAMP_RD	SAMP_WR	SEQ_ADR	SEQ_RD	SEQ_WR	AUTORUN

Bit	Name	Access	Description
7	IRQ	R/W	1 = ADC Sample Store Register is ready to be read / Write with a 1 to reset
			0 = ADC Sample Store Register is NOT ready to be read
6	IRQENABLE	R/W	1 = ADC Interrupt is Enabled
			0 = ADC Interrupt is Disabled
5	SAMP_RD	R/W	1 = Sets SAMPLE STORE READ pin HIGH, to READ the Sample Store Data Register
			0 = Sets SAMPLE STORE READ pin LOW.
4	SAMP_WR	R/W	1 = Sets SAMPLE STORE WRITE pin HIGH, to WRITE the Sample Store Data Register
			0 = Sets SAMPLE STORE WRITE pin LOW.
3	SEQ_ADR	R/W	1 = Set the ADC Sequencer Address Line HIGH
			0 = Set the ADC Sequencer Address Line LOW
2	SEQ_RD	R/W	1 = Sets SEQ_READ signal to the ADC HIGH. Toggle initiates a READ from the SEQ
			0 = Sets SEQ _READ signal to the ADC HIGH.
1	SEQ_WR	R/W	1 = Sets SEQ _WRITE signal to the ADC HIGH. Toggle initiates a WRITE to the SEQ
			0 = Sets SEQ _WRITE signal to the ADC HIGH.
0	AUTORUN	W	1 = Autorun of the ADC. Causes a write of the Sample Data Register and Sequencer
			0 = Does nothing

2.14.3 ADC Sequencer Data Register

Address = Base + 3	SEQD: Sequencer Data Register					Reset Value = 0x00	
7:0->	SEQD7	SEQD6	SEQD5	SEQD4	SEQD3	SEQD2	SEQD1	SEQD0
Bit	**Name**	**Access**	**Description**					
7-0	SEQD[7:0]	R/W	R = Read Sequencer Data Register Bits 7-0 (Bit 7 is never used)					
			W = Write Sequencer Data Register Bits 7-0 (Bit 7 is never used)					

2.14.4 ADC Sample Store Address Register

Address = Base + 2	SSADDR: Sample Store Address Register						Reset Value = 0x00	
7:0->	SSADDR7	SSADDR6	SSADDR5	SSADDR4	SSADDR3	SSADDR2	SSADDR1	SSADDR0
Bit	**Name**	**Access**	**Description**					
7-0	SSADDR[7:0]	R/W	R = Read Sample Store Address Register Bits 7-0					
			W = Write Sample Store Address Register Bits 7-0					

2.14.5 ADC Sample Store Data Registers

Address = Base + 1	SAMPDH: Sample Store Data Register Hi						Reset Value = 0x00	
7:0->	SAMPD15	SAMPD14	SAMPD13	SAMPD12	SAMPD11	SAMPD10	SAMPD9	SAMPD8
Bit	**Name**	**Access**	**Description**					
15-8	SAMPD[15:0]	R/W	R = Read Sequencer Data Register Bits 15-8					
			W = Write Sample Data Register Bits 15-8					

Address = Base + 0	SAMPDL: Sample Store Data Register Lo						Reset Value = 0x00	
7:0->	SAMPD7	SAMPD6	SAMPD5	SAMPD4	SAMPD3	SAMPD2	SAMPD1	SAMPD0
Bit	**Name**	**Access**	**Description**					
7-0	SAMPD[7:0]	R/W	R = Read Sequencer Data Register Bits 7-0					
			W = Write Sample Data Register Bits 7-0					

2.14.6 ADC_TSD Temperature Conversion Chart in °C

Temp	Code	Temp	Code	Temp	Code	Temp	Code	Temp	Code
-40	3798	-6	3738	28	3670	62	3593	96	3510
-39	3796	-5	3736	29	3667	63	3592	97	3507
-38	3795	-4	3733	30	3666	64	3591	98	3504
-37	3793	-3	3732	31	3664	65	3590	99	3501
-36	3792	-2	3731	32	3662	66	3589	100	3500
-35	3790	-1	3730	33	3660	67	3585	101	3498
-34	3788	0	3727	34	3658	68	3582	102	3496
-33	3786	1	3725	35	3656	69	3579	103	3494
-32	3785	2	3721	36	3654	70	3576	104	3492
-31	3782	3	3720	37	3651	71	3573	105	3490
-30	3781	4	3719	38	3648	72	3570	106	3489
-29	3780	5	3717	39	3645	73	3567	107	3486
-28	3779	6	3715	40	3643	74	3564	108	3483
-27	3777	7	3713	41	3642	75	3561	109	3480
-26	3775	8	3711	42	3641	76	3558	110	3477
-25	3773	9	3709	43	3640	77	3555	111	3474
-24	3771	10	3707	44	3638	78	3552	112	3471
-23	3770	11	3704	45	3636	79	3551	113	3468
-22	3768	12	3703	46	3634	80	3550	114	3465
-21	3766	13	3702	47	3632	81	3549	115	3461
-20	3765	14	3700	48	3630	82	3548	116	3460
-19	3764	15	3699	49	3628	83	3547	117	3459
-18	3762	16	3698	50	3625	84	3546	118	3456
-17	3759	17	3697	51	3622	85	3542	119	3451
-16	3756	18	3696	52	3619	86	3538	120	3450
-15	3754	19	3695	53	3616	87	3534	121	3449
-14	3752	20	3688	54	3613	88	3530	122	3445
-13	3751	21	3684	55	3610	89	3526	123	3440
-12	3750	22	3682	56	3607	90	3525	124	3432
-11	3748	23	3680	57	3604	91	3524	125	3431
-10	3746	24	3678	58	3601	92	3522	—	—
-9	3744	25	3677	59	3598	93	3519	—	—
-8	3742	26	3676	60	3595	94	3516	—	—
-7	3740	27	3673	61	3594	95	3513	—	—

The table above provides conversion data for the built-in temperature sensor. The Code is the decimal equivalent of the data read from the Sample Store Data Registers within the ADC.

2.14.7 ADC Code Example

```
      ADC_BASE:         equ $7FB0         ;ADC Base Address
      ADC_SAMDATA0:     equ ADC_BASE
      ADC_SAMDATA1:     equ ADC_BASE+1
      ADC_SAMADDR:      equ ADC_BASE+2
      ADC_SEQDATA0:     equ ADC_BASE+3
      ADC_CTRL:         equ ADC_BASE+4

      CHIP  65C02
      LONGI OFF
      LONGA OFF
      org   $8000       ; Our program code starting point
;****************************************************************************
;****************************************************************************
;                     ADC Sample Code
;****************************************************************************
;****************************************************************************
START:
      sei               ; Turn on interrupts
      cld
      ldx #$ff          ; Initialize the stack pointer
      txs

;Setup the ADC
;WRITE A $03 to SEQ for Single Cycle, $01 for Continuous
      lda #$03
      sta ADC_SEQDATA0  ; Set Sequencer Data to RUN

      lda #$01                ;AUTORUN, will WR SAMPLE and SEQUENCE
      sta ADC_CTRL            ;

;Poll for End of Packet Interrupt
EOPIRQ:
      lda ADC_CTRL
      bit #$80
      beq EOPIRQ

;Read Register Address 8; Channel 9 (Temperature Sensor Diode)
      lda #$08
      sta ADC_SAMADDR         ; Sample Register 9 (TSD)
      lda ADC_SAMDATA0
      sta $0620               ;Just a location in memory
      lda ADC_SAMDATA1
      sta $0621

; We finished the first sample, let's reset the EOP IRQ line
      lda #$01
      sta ADC_SAMDATA0  ; (0x01) will reset the End of Packet (EOP) IRQ
      lda #$41                ; (0x41) address of the Interrupt Sample Register
      sta ADC_SAMADDR         ; Write Value for ISR (0x41) in the Sample Storage Core
      lda #$10                ; (0x10) Control Reg Command to Write Sample Store Core
      sta ADC_CTRL            ; Write command into the Control Register
```

2.15 Chip ID Module

Each MAX10M16 Intel® FPGA has a unique 64-bit chip ID for 18,446,744,073,709,551,616 Unique IDs. Chip ID Intel FPGA IP cores allow you to read out this chip ID for device identification. A simple interface allows reading of the ID out in 8x 8-bit registers ($7FC7-$7FC0). A Data Valid register ($7FC8) can be read ($FF value indicates valid; $00 invalid) to check that the Chip ID can be read from the FPGA. All registers are READ ONLY.

2.15.1 Chip ID Register Description

Chip ID Registers ($7FC0-$7FC8):

Register	Address Offset	Description
CHIPID0	0	Chip ID(Bits 7-0)
CHIPID0	1	Chip ID (Bits 15-8)
CHIPID0	2	Chip ID (Bits 23-16)
CHIPID0	3	Chip ID (Bits 31-24)
CHIPID0	4	Chip ID (Bits 39-32)
CHIPID0	5	Chip ID (Bits 47-40)
CHIPID0	6	Chip ID (Bits 55-48)
CHIPID7	7	Chip ID (Bits 63-56)
DVALID	8	ID Valid – 0xFF = VALID; 0x00 = INVALID

2.16 User FLASH Module

The User FLASH Module (UFM) is an interface to the Intel FPGA On-Chip 296K byte User FLASH Memory (also UFM). The UFM is configured as a 303,104 byte FLASH module for initializing RAM as ROM for boot-loading code on start-up into protected SRAM. The UFM has a 32-bit parallel data bus Avalon interface adapted to the WDC standard 8-bit data and 16-bit address bus.

2.16.1 User FLASH Module Register Overview

Register	Address	Description
DATA0	0	FLASH Data Bits 7-0
DATA1	1	FLASH Data Bits 15-8
DATA2	2	FLASH Data Bits 23-16
DATA3	3	FLASH Data Bits 31-24
ADDR0	4	FLASH Address Bits 7-0
ADDR1	5	FLASH Address Bits 15-8
ADDR2	6	FLASH Address Bits 23-16
PER0	7	Page Erase Register 7-0
PER1	8	Page Erase Register 15-8
PER2	9	Page Erase Register 21-16
SER	A	Sector Erase Register (Only uses bits 2-0)
WPR	B	Write Protect Register (Only uses bits 4:0)
CMD	C	Command Register - Used to initiate reads and writes to the FLASH.

2.16.2 UFM Command Register Description

Address = Base + C		CMD: Command Register					Reset Value = 0x00	
7:0->	WAITREQ	DRD	DWR	-	CSA	CSR	CSW	-
Bit	**Name**	**Access**	**Description**					
7	WAITREQ	R/W	1 = On-Chip FLASH is BUSY with a Read/Write operation 0 = On-Chip FLASH is not BUSY					
6	DATA_READ	R/W	1 = Sets DATA_READ signal to the UFM HIGH. Used to READ the UFM Data Register 0 = Sets DATA_READ signal to the UFM LOW.					
5	DATA_WRITE	R/W	1 = Sets DATA_WRITE signal to the UFM HIGH. Used to WRITE the UFM Data Register 0 = Sets DATA_WRITE signal to the UFM LOW.					
4	RESERVED	R/W	Reserved for future use					
3	CSR_ADDR	R/W	1 = Selects the UFM Control Register 0 = Selects the UFM Status Register					
2	CSR_READ	R/W	1 = Sets CSR_READ signal to the UFM HIGH. Toggle initiates a READ from the CSR 0 = Sets CSR_READ signal to the UFM HIGH.					
1	CSR_WRITE	R/W	1 = Sets CSR_WRITE signal to the UFM HIGH. Toggle initiates a WRITE to the CSR 0 = Sets CSR_WRITE signal to the UFM HIGH.					
0	RESERVED	R/W	Reserved for future use					

2.16.3 UFM Write Protect Register Description

Address = Base + B		CMD: Command Register					Reset Value = 0x1F	
7:0->	-	-	-	WPSID5	WPSID4	WPSID3	WPSID2	WPSID1
Bit	**Name**	**Access**	**Description**					
7-5	-	NA	Bits 7-5 are not used					
4	WPSID5	R/W	1 = DEFAULT – Sector 5 is Write Protected 0 = Sector 5 is NOT Write Protected					
3	WPSID4	R/W	1 = DEFAULT – Sector 4 is Write Protected 0 = Sector 4 is NOT Write Protected					
2	WPSID3	R/W	1 = DEFAULT – Sector 3 is Write Protected 0 = Sector 3 is NOT Write Protected					
1	WPSID2	R/W	1 = DEFAULT – Sector 2 is Write Protected 0 = Sector 2 is NOT Write Protected					
0	WPSID1	R/W	1 = DEFAULT – Sector 1 is Write Protected 0 = Sector 1 is NOT Write Protected					

2.16.4 UFM Sector Erase Register Description

Address = Base + A		CMD: Command Register					Reset Value = 0x00	
7:0->	-	-	-	-	-	SE2	SE1	SE0
Bit	**Name**	**Access**	**Description**					
7-3	-	NA	Bits 7-3 are not used					
2-0	SE[2:0]	R/W	3b'101 = Sector 5 Erased ; 3b'100 = Sector 4 Erased; 3b'011 = Sector 3 Address					
			3b'010 = Sector 2 Erased ; 3b'001 = Sector 1 Erased					

2.16.5 UFM Page Erase Register Description

Address = Base + 9		UFM_PE2: Page Erase Register 2					Reset Value = 0x00	
7:0->	PE23	PE22	PE21	PE20	PE19	PE18	PE17	PE16
Bit	**Name**	**Access**	**Description**					
7 – 0	PE[23:16]	R/W	R = READ Page Erase Address					
			W = WRITE Page Erase Address					
Address = Base + 8		**UFM_DW0: UFM Write Data Register 1**					**Reset Value = 0x00**	
7:0->	PE15	PE14	PE13	PE12	PE11	PE10	PE9	PE8
Bit	**Name**	**Access**	**Description**					
7 – 0	PE[15:8]	R/W	R = READ Page Erase Address					
			W = WRITE Page Erase Address					
Address = Base + 7		**UFM_DW0: UFM Write Data Register 0**					**Reset Value = 0x00**	
7:0->	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
Bit	**Name**	**Access**	**Description**					
7 – 0	PE[7:0]	R/W	R = READ Page Erase Address					
			W = WRITE Page Erase Address					

2.16.6 UFM FLASH Address Registers Description

Address = Base + 6		UFM_ADDR2: FLASH Address Register 2					Reset Value = 0x00	
7:0->	ADDR23	ADDR22	ADDR21	ADD20	ADDR19	ADDR18	ADDR17	ADDR16
Bit	Name	Access	Description					
7 – 0	ADDR[23:16]	R/W	R = READ FLASH Address W = WRITE FLASH Address					
Address = Base + 5		UFM_ADDR1: FLASH Address Register 1					Reset Value = 0x00	
7:0->	ADDR15	ADDR14	ADDR13	ADDR12	ADDR11	ADDR10	ADDR9	ADDR8
Bit	Name	Access	Description					
7 – 0	ADDR 15:8]	R/W	R = READ FLASH Address W = WRITE FLASH Address					
Address = Base + 4		UFM_ADDR2: FLASH Address Register 0					Reset Value = 0x00	
7:0->	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
Bit	Name	Access	Description					
7 – 0	ADDR[7:0]	R/W	R = READ FLASH Address W = WRITE FLASH Address					

2.16.7 UFM FLASH Data Registers Description

Address = Base + 3	UFM_DW3: UFM Data Register 3					Reset Value = 0x00	
7:0->	DW31	DW30	DW29	DW28	DW27	DW26	DW25	DW24
Bit	**Name**	**Access**	**Description**					
7 – 0	DW[31:24]	R/W	R = Read UFM Data Register 0					
			W = Read UFM Data Register; Values written to internal USER FLASH					
Address = Base + 2	**UFM_DW2: UFM Data Register 2**					**Reset Value = 0x00**		
7:0->	DW23	DW22	DW21	DW20	DW19	DW18	DW17	DW16
Bit	**Name**	**Access**	**Description**					
7 – 0	DW[23:16]	R/W	R = Read UFM Data Register 0					
			W = Read UFM Data Register; Values written to internal USER FLASH					
Address = Base + 1	**UFM_DW1: UFM Write Data Register 1**					**Reset Value = 0x00**		
7:0->	DW15	DW14	DW13	DW12	DW11	DW10	DW9	DW8
Bit	**Name**	**Access**	**Description**					
7 – 0	DW[15:8]	R/W	R = Read UFM Data Register 0					
			W = Read UFM Data Register; Values written to internal USER FLASH					
Address = Base + 0	**UFM_DW0: UFM Write Data Register 0**					**Reset Value = 0x00**		
7:0->	DW7	DW6	DW5	DW4	DW3	DW2	DW1	DW0
Bit	**Name**	**Access**	**Description**					
7 – 0	DW[7:0]	R/W	R = Read UFM Data Register 0					
			W = Read UFM Data Register; Values written to internal USER FLASH					

2.17 MyMENSCH™ Monitor

MyMENSCH™ Monitor interfaces MyMENSCH™ to a PC that has a Python terminal to enable code development functions. The following functions are featured in the Monitor:

Function Name	Description
Sync	Sends 0x00 to the Terminal
Get_Info	Sends Board model and Monitor Version information to the Terminal.
Write_Data_To_Memory	Write Data From the Terminal to Memory
Read_Data_From_Memory	Read Data Memory to Terminal
Execute	Execute program from Memory (Not USER FLASH)
Write_Data_To_Flash	Write Data from the Terminal, into Memory and then to USER FLASH
Read_Data_From_Flash	Read Data from USER FLASH to Terminal
Clear_Flash	Clears all Data from the entire USER FLASH
Check_Flash	Checks to make sure USER FLASH is not busy and is not cleared
Execute_From_Flash	Execute program from USER FLASH

This page intentionally left blank.

3 IO Connectors with Ball Assignments on MyMENSCH™

3.1 Left IO Connector J3 on MyMENSCH™

The J3 left connector has 46 IO, 2x 3v3 power and 2x VSS pins. Ball assignments labeled NA are Non-Assigned pins. Note that VIA_C_PA3 is not available, all other IO pins of VIA A, B, and C are available.

J3 – Left Expansion Connector					
Pin	Signal Name	FPGA Ball	Pin	Signal Name	FPGA Ball
1	VSS	-	2	VDD	-
3	I2C_SCL	L4	4	VIA_C_PA6	L3
5	I2C_SDA	K6	6	VIA_C_PA5	K5
7	VIA_C_PA7	H4	8	VIA_C_PA4	N2
9	VIA_A_CB2	M4	10	VIA_C_CB2	N3
11	VIA_A_CB1	M5	12	VIA_C_CB1	N4
13	VIA_A_PB7	L5	14	VIA_C_PB7	N5
15	VIA_A_PB6	N7	16	VIA_C_PB6	N6
17	VIA_A_PB5	N8	18	VIA_C_PB5	M7
19	VIA_A_PB4	M9	20	VIA_C_PB4	M8
21	VIA_A_PB3	M10	22	VIA_C_PB3	N9
23	VIA_A_PB2	M11	24	VIA_C_PB2	N10
25	VIA_A_PB1	N12	26	VIA_C_PB1	N11
27	VIA_A_PB0	M13	28	VIA_C_PB0	M12
29	VIA_A_PA7	L13	30	GPIO_A7	L12
31	VIA_A_PA6	K13	32	GPIO_A6	K12
33	VIA_A_PA5	K8	34	GPIO_A5	J8
35	VIA_A_PA4	J9	36	GPIO_A4	L10
37	VIA_A_PA3	K10	38	GPIO_A3	L11
39	VIA_A_PA2	K11	40	GPIO_A2	J10
41	VIA_A_PA1	H9	42	GPIO_A1	H10
43	VIA_A_PA0	J12	44	GPIO_A0	J13
45	VIA_A_CA1	H13	46	VIA_C_CA1	G12
47	VIA_A_CA2	G13	48	VIA_C_CA2	F12
49	VDD	-	50	VSS	-

3.2 Right IO Connector J4 on MyMENSCH™

The J4 right connector has 46 IO, 2x 3v3 power and 2x VSS pins. Ball assignments labeled NA are Non-Assigned pins. Note that VIA_C_PA3 and VIA_B_PB6 are not available on Rev-A, all other IO pins of VIA A, B, and C are available. MyMENSCH™ Rev-B and -C will replace the five JTAG pins with AGND, dedicated ADC input ADC_IN0, 3v3REF, 5vBAT, and VIA_B_PB6; however, VIA_C_PA3 is not available on either Rev-A, -B, or -C.

J4 – Right Expansion Connector					
Pin	***Signal Name***	***FPGA Ball***	***Pin***	(Rev-A) ***Signal Name*** (Rev-B,C)	***(Rev-A) FPGA Ball (Rev-B,C)***
1	VDD	-	2	VSS	-
3	VIA_C_PA2	H3	4	(JTAG_TCK) or (AGND)	(G2) or (E2)
5	VIA_C_PA1	H1	6	(JTAG_TDI) or (ADC_IN0)	(F5) or (D2)
7	VIA_C_PA0	H2	8	(JTAG_TDO) or (3v3REF)	(F6) or (D3)
9	ADC_IN5	F1	10	(JTAG_TMS) or (5vBAT)	(G1) or (-)
11	ADC_IN6	E1	12	(JTAG_EN) or (VIA_B_PB6)	(E5) or (F4)
13	ADC_IN7	C1	14	ADC_IN1	D1
15	ADC_IN8	B1	16	ADC_IN2	C2
17	VIA_B_PB5	B2	18	ADC_IN3	E3
19	VIA_B_PB4	A2	20	ADC_IN4	E4
21	VIA_B_PB3	B3	22	VIA_B_CB2	E6
23	VIA_B_PB2	B4	24	VIA_B_CB1	A3
25	VIA_B_PB1	B5	26	VIA_B_PB7	A4
27	VIA_B_PB0	B6	28	SPI_SDI_A	A5
29	VIA_B_PA7	B7	30	SPI_SDO_A	A6
31	VIA_B_PA6	A7	32	SPI_SCLK_A	D9
33	VIA_B_PA5	A8	34	TXD_C	E8
35	VIA_B_PA4	C9	36	RXD_C	F8
37	VIA_B_PA3	C10	38	RTSB_C_E5	A9
39	VIA_B_PA2	B10	40	CTSB_C_E1	A10
41	VIA_B_PA1	B11	42	TXD_B	A11
43	VIA_B_PA0	B12	44	RXD_B	A12
45	VIA_B_CA1	B13	46	RTSB_B_E6	C11
47	VIA_B_CA2	C13	48	CTSB_B_E2	C12
49	VSS	-	50	VDD	-

4 FCC Compliance

The Western Design Center, Inc. (WDC) provides the enclosed product under the following conditions: This board is intended for use for Engineering Development or Evaluation Purposes ONLY and is not considered by WDC to be a finished consumer product. This board should be handled with caution using good electronics handling practices. This board is compliant per RoHS/Green directives. It does not fall within the scope of directives such as FCC, CE, and UL. It generates uses and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to part 15 of FCC rules.

5 Ordering Information

The W65C265i1M16SA is available from WDC Direct and our distribution partners. For information please visit: http://wdc65xx.com/where-to-buy/